TALLINN UNIVERSITY OF TECHNOLOGY
DOCTORAL THESIS
28/2020

Mediated Interactions for Collection and Exchange of Situational Information in Smart Environments

JAANUS KAUGERAND

TALTECH PRESS

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies
Department of Software Science

The dissertation was accepted for the defence of the degree of Doctor of Philosophy in Computer Science on 3 August 2020

Supervisor: Jürgo-Sören Preden,
Department of Software Science, School of Information Technologies
Tallinn University of Technology
Tallinn, Estonia

Opponents: Professor Michael Henshaw,
Loughborough University,
Loughborough, Leicestershire, United Kingdom

Dr Gabriel Jakobson,
CyberGem Consulting, USA,

Defence of the thesis: 18 August 2020, Tallinn

Declaration:
Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology, has not been submitted for any academic degree elsewhere.

Jaanus Kaugerand ______________________
signature

European Union
European Regional
Development Fund

Investing
in your future

ISSN 2585-6898 (publication)
ISBN 978-9949-83-582-9 (publication)
ISSN 2585-6901 (PDF)
ISBN 978-9949-83-583-6 (PDF)
Printed by Koopia Niini & Rauam

TALLINNA TEHNIKAÜLIKOOL
DOKTORITÖÖ
28/2020

Vahendatud interaktsioonid olukorrateadlikkuse informatsiooni kogumiseks ja vahetamiseks arukates keskkondades

JAANUS KAUGERAND

KIRJASTUS

Contents

List of Publications

The present Ph.D. thesis is based on the following publications that are referred to in the text by Roman numbers.

I **J. Kaugerand**, J. Ehala, L. Mõtus, and J.-S. Preden. Time-selective data fusion for in-network processing in ad hoc wireless sensor networks. *International Journal of Distributed Sensor Networks*, 14(11):1–17, 2018

II J. Ehala, **J. Kaugerand**, R. Pahtma, S. Astapov, A. Riid, T. Tomson, J.-S. Preden, and L. Mõtus. Situation awareness via internet of things and in-network data processing. *International Journal of Distributed Sensor Networks*, 13(1):1–21, 2017

III **J. Kaugerand**, J.-S. Preden, E. Suurjaak, S. Astopov, L. Motus, and R. Pahtma. A system of systems solution for perimeter control: Combining unmanned aerial system with unattended ground sensor network. In *Systems Conference (SysCon), 2015 9th Annual IEEE International*, pages 317–323. IEEE, 2015

IV J.-S. Preden, **J. Kaugerand**, E. Suurjaak, S. Astapov, L. Motus, and R. Pahtma. Data to decision: pushing situational information needs to the edge of the network. In *Proceedings of the 2015 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision*, pages 158–164. New York: IEEE, Orlando, FL, USA, 9-12 March 2015

V S. Astapov, J. Berdnikova, J. Ehala, **J. Kaugerand**, and J.-S. Preden. Gunshot acoustic event identification and shooter localization in a wsn of asynchronous multichannel acoustic ground sensors. *Multidimensional Systems and Signal Processing*, 29(2):563–595, 2018

VI A. Riid, **J. Kaugerand**, J. Ehala, M. Jaanus, and J.-S. Preden. An application of a low-cost microwave radar to traffic monitoring. In *2018 16th Biennial Baltic Electronics Conference (BEC)*, pages 1–4. IEEE, 2018

VII L. Motus, M. Teichmann, T. Kangilaski, J. Priisalu, and **J. Kaugerand**. Some issues in modelling comprehensive situation awareness. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 540–545. IEEE, 2019

Author’s Contributions to the Publications

I In Publication I, I was the main author, designed and implemented the algorithm for temporal data alignment and selection of suitable elements from streams for the fusion algorithm, designed the network for the experiment, carried out the experiment and the analysis of the results, prepared the figures and wrote the manuscript.

II In Publication II, the work described in this paper was shared between two main contributors, Ehala and author of this thesis. I focused on identification of military requirements and the design of the network according to these requirements. Work with data processing algorithms and choosing optimal parameters for validity and timing and the analysis of the experiment results was equally shared between me and Ehala. The workload sharing during manuscript preparation fell more on Ehala than author of the this thesis.

III In Publication III, I was the main author. I designed the architecture of the system, the autonomy and message handling components for the unmanned aerial system. I also described how to apply the concept of System of Systems to integrate heterogeneous systems such as autonomous vehicles with Smart Environments. I wrote the paper.

IV In Publication IV, I Created the example application to demonstrate the usefulness of pushing the computation capability required for the decision making to the edge and carried out the experiments.

V In Publication V, I prepared the computational hardware and software of the sensor nodes for the experiment. I managed synchronous start of the WSN nodes during the experiments. I helped with the timing analysis of the event analysis at fusion level. I carried out the experiments to test the simultaneity interval of the synchronous start.

VI In Publication VI, I worked with sensor design at all stages. I assembled equipment for the experiments and together with colleagues from ProLab performed the field work during the experiments including raw data collection and its analysis.

VII In Publication VII, I contributed with an insight and understanding how data validity together with spatial and temporal consistency of data are necessary for situational information processing.

Abbreviations

AODV	Ad hoc On-demand Distance Vector
AoA	Angle of Arrival
API	Application Programming Interface
CPS	Cyber-Physical Systems
CPU	Central Processing Unit
CMSIS	Cortex Microcontroller Software Interface Standard
DAQ	Data Acquisition
DIL	Disrupted, Intermittent and Limited
DSA	Distributed Situation Awareness
DYMO	Dynamic MANET On-demand
HVAC	Heating Ventilation and Air Conditioning
IEEE	Institute of Electrical and Electronics Engineers
INDP	In-network data processing
IoT	Internet of Things
ISR	Intelligence, Surveillance and Reconnaissance
JDL	Joint Directors of Laboratories
MAC	Media Access Control
MAS	Multi Agent System
MANET	Mobile Ad hoc Network
OLSR	Optimised Link State Routing
OODA	Observe, Orient, Decide, Act
RF	Radio Frequency
RTOS	Real Time Operating System
SA	Situation Awareness
SE	Smart Environment
SI	Situational Information
SNR	Signal-to-Noise Ratio
SoS	System of Systems
SP	Situation Parameter
TDMA	Time Division Media Access
UAV	Unmanned Aerial Vehicle
UAS	Unmanned Autonomous System
UGS	Unattended Ground Sensors
UML	Universal Modelling Language
UTC	Universal Time Coordinated
WSN	Wireless Sensor Network
XML	Extensible Markup Language

1 Introduction

Tens of billions of devices connected by the Internet of Things (IoT), which have been deployed already and tens of billions more that will be deployed in the coming decades, will operate in our environment, enhancing our capability for acquiring real-time data for decision making and automating mundane tasks. The lowest layer of these devices will operate on low-power, low-bandwidth embedded networks, which conventionally have been called wireless sensor networks (WSNs). Such low-bandwidth networks are ideal for collecting data from thousands of low-cost devices that are deployed in different environments. For example, during the project SMENETE2, Tallinn University of Technology deployed a wireless sensor network (WSN) in Tallinn city consisting of ca 900 sensors prove the readiness of WSN technology for Smart Environments (SE). This network of smart sensors is just an example of a first wave of what is coming in the future. There are various forecasts for the number of electronically connected devices that form IoT, for example according to a forecast from International Data Corporation (IDC), the number of connected IoT devices will reach 41.6 billion by 2025 [86].

The opportunity arising from embedding sensors in physical things and environments surrounding us is the creation of Smart Environments (SE). The concept of SE suggests that through embedding the sensors and capability of computation deeply to physical world surrounding us, the environment itself becomes smart. It is here, where today's challenges of interest start. What does it mean for the environment to be smart? Are we talking about a house which regulates its temperature to a convenient level for inhabitants or a situation aware street which helps a vehicle to reach its destination both safely and in a timely fashion or a smart city which provides a comprehensive situational overview to help city official's in decision making? Are we talking about an environment that is enhanced with a capability to provide real-time situational awareness for its users inhabiting the environment? These are questions that cannot be addressed without understanding the limitations and opportunities that technology imposes and opens up for these solutions.

The devices in sensor networks, which are the first examples of the Internet of Things have been mostly connected using wired connections and/or fixed to the monitored object. However, in many situations, the hard-wiring of sensors incurs higher costs and is unscalable. To mitigate this, radio frequency (RF) has been used to communicate the sensed data. Cost-effective solutions are needed to make IoT financially viable, to improve the return of investment of IoT solutions.

IoT brings features and functionalities we expect from the internet to the physical world and the one value of IoT solutions is that these systems help to provide situational information to humans and artificial systems. While most people do not realise it, the main purpose of IoT solutions is collecting data for generating situational information for humans and machines for decision-making. These solutions are used in all industries and by a broad range of people.

Embedded computation and communication technologies for building large wireless sensor networks are becoming both capable and cheap enough for performing complex computational tasks for data acquisition for generating situational information. Solutions, which have practical applications, are emerging. Examples of sensor networks can already be seen more and more surrounding us in our everyday life. Sensors are no longer only in factories, vehicles and other artificial environments, but are moving into our everyday environments to track situations which are related to our daily life. Sensors are deployed in smart buildings to monitor heating, ventilation, noise, to count people occupying the rooms, etc. Sensors are deployed as wearables for monitoring peoples' health condition.

Sensors are also deployed into urban environments to monitor an abounding number of unfolding situations in streets, squares, parks etc. Sensors are placed into natural environments to monitor both wild life and weather phenomena. And sensors are used in defence and security applications to detect or warn against unwanted situations. The applications are multitudinous. It is clear that networks of such sensors have a genuine potential of producing huge amounts of information about the world evolving around us. Having things connected to traditional internet, sensors in things and things that also compute something is called the Internet of Things.

Internet of Things is a new types of Cyber-Physical System (CPS) that requires novel approaches to systems engineering to be successfully adopted and deployed [28]. Often the networks of IoT devices also form Systems of Systems (SoS). SoS aspects of networks of CPS devices were analysed by author of the thesis in Publication III. As IoT devices are CPS and when networked they can form SoSs and also the fact that the amount of data collected from these devices is potentially huge, means that the use conventional methods for data collection or processing may not be sufficient of even possible. Instead, the processing has to be performed at the network edge, as much as possible and only higher-level situation assessments can be communicated to other devices or to the Cloud. In this way the thesis focuses not so much on individual algorithms, but on a system level view of data collection and exchange for SA via Smart Environments.

Combining IoT devices and/or embedded devices connected into SoSs has the potential of creating large scale heterogeneous sensor networks. Using such SoSs to create Smart Environments can be challenging because of the issues listed above. As stated above, in large-scale sensor networks (which includes IoT), not all collected raw data can be transmitted to the single end-point (e.g. an internet gateway), nor is it not always necessary to transmit all data to central cloud servers, where it would be stored for further processing. Essentially, this is because the available communication channels reach their limits for the data transport with large number of nodes producing the data. One solution for this is to process the data on-line by local nodes, either in the nodes which collect the data - this is called in-sensor data processing (ISDP) or in distributed fashion in several nodes by exchanging the results of local computations and processing the data in the network - this is called in-network data processing (INDP). Together both methods fit under a concept called Edge Computing (EC). In the case of distributed mesh networks of low power devices (e.g. WSNs) the combination of concepts of in-network data processing and in-sensor data processing have also been called Mist Computing (MC) [112]. MC is a new concept according to which, the computation takes place in the very edge nodes of the network, but at the same time, in order to complete the computation, requires input both from the surrounding physical environment and from other distributed network nodes.

Another concept, similar to the IoT and SE, to which the phenomenon of integrating computational and networking capabilities with both physical and often also social processes has led to, is cyber-physical-social systems (CPSS). Imre Horvath describes these systems in [53] as structurally and functionally open, context-sensitive, intelligent and self-managing engineered systems in which the physical and the cyber constituents evolve co-operatively, and which gradually penetrate into the social world, as well as into the mental world of humans. This is what a Smart Environment does, it moves the (computational) intelligence closer to the sources of data, one could even say that EC in Smart Environments is a step closer towards collective intelligence [87]. This thesis does not use the notion of CPSS, but utilises concept of Situation Awareness (SA) that assumes cognitive processes behind forming of SA. The concept of SA allows to use models of ongoing situations on a local scale to hierarchically build up or compose larger scale or more abstract or more complex models of situations. The nodes which hold the models of situations need constant streams of situational information in order to keep the models updated. The challenge is that components of these systems (e.g. IoT or WSN devices) are often physically distant, but the interactions between them are often expected and assumed by designers in near real time.

Making use of the lessons learned from the field of SA (this topic is handled in chapter 2.4), it is known that it is considerably more advantageous to create (both distributed and) shared models of situations as early as possible in the SA hierarchy to avoid the stovepipe effects later as described in Publication IV. In this view, in order to achieve a common understanding of a specific situation in a Smart Environment (a system of large scale distributed sensor networks which monitor their surrounding physical phenomena), it can be extremely useful to exchange and share the learned understandings of situations between the system nodes as early as possible already on the local scale, before the situational information is passed upwards in the SA hierarchy. Another important lesson from the field of SA is that in order to acquire correct data for providing situational information, both data acquisition and processing should be driven by the SA needs, i.e., be goal-driven.

Classical sensor systems have been used to collect all data to a central server, where logical rules can be used to detect events from historical data. SA-driven data collection requires pushing data processing rules to the edge nodes, closer to the physical phenomena, where the events of interest actually occur, closer to the edge - leading to EC. EC includes methods such as in-sensor data processing (ISDP) and in-network data processing (INDP), combining these with ad hoc mesh networks leads to Mist Computing (MC), enabling novel paradigms for Smart Environments such as data to decision (D2D). In Publication IV, authors describe that according to the D2D only these data are collected that are needed to make situation assessment for current decisions. This thesis suggests that both D2D and Mist Computing can be exploited when searching for methods to manage the large data flows.

However, the large networks of sensors already do produce huge amounts of data. The sensors in those networks can be both heterogeneous and are often autonomous. This leads to a range of technical issues that arise when assembling usable information from the data produced by these sensors into e.g., using the data as an input for a model of ongoing situation assessment. Especially when situations are handled more globally than just a single application domain, e.g. couple of more extreme examples of data acquisition applications that one could think of would be either a digital twin of a city [123] or even an application for comprehensive Situation Awareness of a nation as described in Publication VII.

For these reasons this thesis focuses on the assessment of situations already at the edge of the networks by taking advantage of concepts such as MC. The individual distributed networked sensor nodes have become capable of processing raw data both locally and in-network, described in Publication II. INDP consists of two main methods, sensor data aggregation and sensor data fusion. The first combines similar data either into a bundle or to a higher abstraction level and the second integrates data, possibly with different modalities, to create new data types at a higher level of abstraction. Both methods can contribute considerably to both bandwidth reduction and enabling of generation of SA information already in the sensor networks for Smart Environments.

This thesis focuses on the part of Smart Environments that is based on ad hoc mesh wireless sensor networks (WSNs), where the reality is that the communication channels for interactions between the network nodes, to other networks and to the general internet are often Disrupted Intermittent and Limited (DIL) [83]. Data transport over such networks is with time-variable delay. This must be taken into consideration when using data from such heterogeneous and distributed sources. As otherwise the INDP is performed on data from distributed sources with erroneous temporal and spatial context. This thesis uses the concept of mediated interactions to mitigate this issue. A smart mediator agent is used to filter out data that is not contextually valid and for selecting the correct data for SA applications from the sensor data streams.

This thesis presents examples of Smart Environments that follow the above described paradigms, bringing computation to the edge of the network where local situation assessment is made in the network nodes, applying mediated interaction concept to enable and manage the in-network interactions and information flows and makes results available to users directly, without being dependent on the cloud. This approach of course does not rule out cloud-based information collection and analysis. The paradigms of Mist Computing, Fog Computing and Cloud Computing are all complementary to each other and are all needed to successfully implement future Smart Environments.

Furthermore, constituent components in such networks are Cyber-Physical Systems (CPS) and are both with interaction with each-other and in substantial interaction with their respective environments. These aspects can lead to a case where not all possible behaviours can be described in advance and unexpected or novel (also known as emergent) behaviours that were not described during design time, can emerge during on-line interactions dynamically. The challenge is that it is impossible to address these the emergent behaviour by offline methods (e.g., by offline validation). Methods to to detect new behaviours and mitigate unwanted behaviours while they are forming i.e. online in due time are required. Although this thesis addresses this problem briefly, this falls out of the scope of this thesis and remains an open topic for future research.

1.1 Motivation and problem formulation

In order to take advantage of the opportunities offered by the emerging Internet of Things technologies there is a need to use novel computing and communication paradigms as it may not be possible to build systems consisting of hundreds of thousands of autonomous sensors by using existing technological approaches. A new approach is needed for the design and implementation of these systems from the application or utility angle - instead of providing raw data, these systems should be able to actively assess information and aid decision-making by humans and machines with the generated situational information.

Thanks to technological developments in the field of low-cost sensors, embedded hardware and software, it is today possible to place a fabric of hundreds and thousands of sensors and actuators (also called IoT devices) to different outdoor and indoor environments, with the capability to produce massive amounts of data. Some examples of such environments can be urban, agricultural, forestry, large constructions, bodies of water, regions of crisis, military theatres etc. Most of these environments can be called non-benign, as they are either radio congested (especially if sensor networks for SE use non licensed frequencies), exhibit natural or non natural and mobile blockages in radio paths, are under influence of natural forces, fail due to hardware or software errors (especially if being based on low-cost technologies), etc. However, the same technological developments also provide a challenging opportunity to build novel SA applications. For example, moving the intelligence closer to sensors deployed at the periphery. The computational intelligence to make use of the sensed data at the edge, in turn, creates Smart Environments. Smart Environments consisting of massive distributed sensor networks have a potential to considerably improve the SA of the agents (that can be biological or artificial) who carry out their tasks within the environments. The information which is required for improving the SA of the users is called situational information. The situational information contains information about detected situations, where some situations at higher level of abstraction have been derived in hierarchical manner by combination of lower level situations. However, the new types of Smart Environments, which often comprise ad hoc, multi hop sensor networks, deployed in non-benign environments, introduce several additional challenges. In these networks, the individual sensing nodes are: distributed, autonomous, working asynchronously, operate on different time scales, their operation is sometimes disrupted by events in environment and the availability of communication can sometimes be only intermittently available and with limited bandwidth (i.e. Disrupted, Intermittent and limited - DIL communication).

The classical SA models have omitted the stage of primary information sensing (and human information acquisition). The main interest of most researchers working on different SA models has been perception and comprehension, explained in detail in 2.4. The available SA models simply assume both the existence of situational information and validity and consistency of generated information by the respective sources. In case of the ad hoc sensor networks, there are many technical aspects (e.g. end-to-end-delay, processing constraints, limited bandwidth, context awareness, etc) that must be taken into account. In addition, the sensor networks use different technologies to transport the data to the agents who need it. These technical difficulties regarding validity and consistency of situational information must be solved in order to hierarchically combine situational information online and convey it to the user in a timely manner. The main problem in that is to guarantee the spatial and temporal consistency of the generated and exchanged situational information as described also in Publication I. This thesis focuses on the problem of improving validity and contextual (e.g. spatial and temporal) consistency of situational information from disparate sources that is used for deriving situations at a higher level of abstraction. The thesis combines several theories in order to take one step closer to a general solution where the concept of mediated interactions is used for collecting and exchange of situational information. The main novelty of applying the mediated interaction concept lies in creating usable situational information already at the edge, where Smart Environments with ad hoc, multi hop sensor networks with imperfect and dynamically changing communication topology are used. These network nodes are Cyber-Physical Systems (e.g. WSN nodes), which often have high level of autonomy and must interact actively with both physical and cyber worlds. The resulting Smart Environment is also considered Systems of Systems which exhibits emergent behaviour. One may conclude that providing situational information in such Smart Environments is a challenge.

1.2 Methodology

This chapter outlines the general concept and design choices of our solution for a Smart Environment based on distributed sensor networks for SA applications. In general, the research problem is approached by utilising the concept of mediated interactions by extending an existing middleware solution used in each node in a distributed sensor network. For this a proactive middleware called ProWare [111] is utilised. This allows to enhance the concept of mediation with new functionality which supports checking temporal and spatial consistency of the situational information from disparate sources. The solution allows to influence and manage the interactions and information flows between the distributed network nodes. The irrelevant, invalid and inconsistent data is filtered out and only validated, contextually relevant and timely situational information is conveyed to the user. The sensor nodes and their interactions within the network are described by applying the concept of a System of Systems (SoS). In a SoS the individual components (i.e., network nodes) are geographically dispersed and independent, the individual nodes can exhibit high levels of both operational and managerial autonomy, their design can include very heterogeneous components and the whole network (system) functionality and performance is more than just a sum of constituent components. Designing the sensor network for Smart Environment with the properties of SoS allows viewing each node as an independent autonomous system and also leads to emergent behaviour. Considering the concept of emergent behaviour and understanding its concept helps potentially to take advantage of the novel behaviours and functions that emerge during the operations.

The chosen solution provides situation awareness to users on different hierarchical levels (from in situ SA to cloud level post-operation analysis), but focuses more on in-the-field units that operate in the monitored environment (e.g. in-the-field military units, autonomous network nodes such as unmanned aerial vehicle (UAV) or even self-driving cars). As such, users task the Smart Environment directly and subscribe to data of their interest. There is no requirement to collect all raw data to a central database and access the data from this database, although a database can still be created and used (e.g. for post-operation analysis). This greatly contrasts the common approach to environment monitoring sensor networks, where users access data as clients of a central database. The service-oriented architecture and publish–subscribe principles fit well with sensor networks designed to provide SA information in Smart Environments under non-benign conditions (e.g. either military or radio congested urban environments), considering the unreliable communication links and persistent shortage of bandwidth that these networks encounter. Allowing users to directly interact with sensors constrains the network less than general network-wide data collection and distribution via a central database.

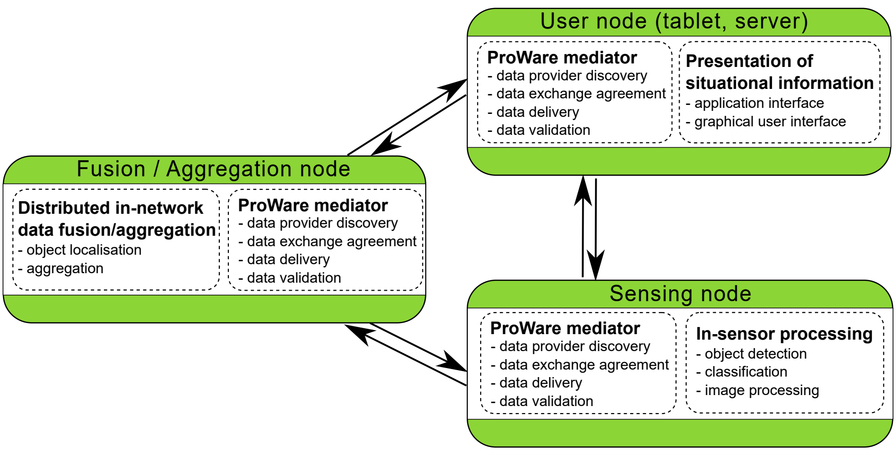


Figure 1 – General overview of the ProWare mediator functionalities.

Figure 1 presents a general overview of the different functionalities offered by ProWare mediator and Smart Environments. Signal acquisition and initial signal processing are conducted in the sensor nodes. The produced situational information is then transmitted to interim fusion or aggregation nodes which combine the distributed lower level situational information into new situational information of higher level of abstraction (having new data types and/or structures). It is possible to have several fusion or aggregation levels before the results are presented to the end user. The thesis discusses four different in-sensor signal processing cases. First, sensor nodes capable of audio signal acquisition compute the angle of arrival (AoA) of measured sound waves based on the time difference of arrival (TDOA) method. Second, some of these sensors are also capable of performing in situ fuzzy classification of the measured sound source (The objective is to classify vehicles based on the sounds they emit). Third, a camera-equipped sensor node performs video analysis with the goal of locating and counting mobile foreground objects (personnel) captured on the video. And fourth, a sensor equipped with a microwave radar capable of detecting passing vehicles' speeds and direction.

The use of mediated interactions allows to build hierarchical combination of situations, carried out via two INDP methods - in-network data fusion and aggregation. Both methods are demonstrated by special network nodes (fusion or aggregation nodes). For example fusion nodes collect AoA estimates from sensor nodes and calculate the location of the sound source from the intersection of beams formed from the AoA estimates. This thesis considers this to be in-network data fusion, since a new data type (location coordinate estimate) was created from another, different type of input data (AoA estimates). Location coordinate estimates can then be combined with fuzzy classification results to form new meaningful data structures (e.g. a data bundle comprising the classification result and location of a detected object). This thesis considers this to be data aggregation, since data of different types are meaningfully grouped together and presented in a human understandable form.

In order to ensure and enhance the quality of in-network data processing the thesis utilises data validity and consistency checking techniques, the validity checking technique allows checking the validity of input data for in-network data processing on-line [111] and consistency checking technique ensures that data from distributed sources used for INDP are mutually consistent i.e. they describe the same situation. For the latter, the thesis develops new methods for data alignment, checking the consistency of data and for selecting the contextually suitable data from input streams. Formally the data exchange and selection of suitable elements from streams during mediation is modelled by Q-model [96]. Q-model is used as a possible candidate for a model of interactive distributed computation. The results of the influence of this technique are experimentally demonstrated.

The capability of the technology solution is validated in three main experiments. The experiments are carried out by using WSNs, which network nodes utilise 8-bit Atmel AVR-based platforms (utilising TinyOS operating system). For more resource demanding computation, some Atmel nodes are supplemented with more powerful platforms such as Rasperry Pi or BeagleBone Black, these platforms are interfaced with Atmel nodes by a serial interface. During the experiments and use cases an unlicensed frequency band for the communication between the nodes is used. (A drawback to unlicensed radio frequency communications is that communication may be susceptible to high levels of interference.) For communication protocol, the IEEE 802.15.4 standard is used. The first main experiment is conducted in a military context. This experiment describes a use case for military application of a wireless ad hoc sensor network and demonstrates its applicability in real world scenario. The second main experiment investigates bandwidth usage and its dependence on contextual constraints used for the mediator. The third main experiment demonstrates that applying selective mediation of interactions for sensor fusion increases its quality considerably. The raw input data for the last two main experiments is recorded with the same deployed sensor network from the real world environment. The same sensor network is then set up in the laboratory conditions where the nodes instead of monitoring live signals, now read the previously recorded raw data and treat it as if it were directly received from real world. The setup is described in more detail in Chapter 5. Using this setup enables repeated replay of the same set of situations with different network and data validity and data mediation configurations. In addition to experiments, the thesis also examines various future use cases of smart environments. At the time of writing this thesis, the Smart Environment use cases are currently being implemented. These use cases require already considerably more powerful hardware platforms for sensor nodes based on energy efficient 32 bit SiLabs Mighty Gecko Systems on Chip.

1.3 Contributions of thesis

This thesis summarises the research and the main results achieved by the author in the framework of ad hoc sensor networks for Situation Awareness applications in Smart Environments. The main focus of the thesis is collection and exchange of consistent and valid situational information via Smart Environments.

The main contributions of this thesis are as follows:

- Application of a concept of mediated interactions that allows to influence and manage the collection and exchange of situational information between distributed network nodes in Smart Environments.
- A novel data alignment and selection technique for allowing time-selective in-network data processing that fosters an INDP node to select the appropriate data from the streams, increases the consistency of detected situations and considerably enhances the performance of INDP algorithms.
- A solution implementation and demonstration, which showed that manipulating with sensor data validity intervals has a considerable effect on quality of INDP. The results from Publication II show that time-selective data processing and on-line checking of validity by the proactive middleware ProWare based on temporal and spatial validity constraints on data has a considerable effect on quality of data produced by INDP algorithms. The difference in successful fusion operations between having very loose or very strict temporal and spatial constraints was around 20 times.
- A demonstration of considerable reduction in bandwidth requirements by using distributed in-network data processing techniques. A demonstration by experiment showed reduction of 10 times in bandwidth.
- A demonstration that time-selective communication approach improves INDP quality. Demonstrated by an experiment in urban environment by improving the precision of vehicle tracking by 23%.
- A revised architectural design for the ProWare middleware to support Smart Environments.

Also, this thesis contributes with Implementation of a set of new in-sensor and in-network data processing algorithms: implementation of audio signal acquisition and computing the angle of arrival (AoA) of measured sound waves based on the time difference of arrival (TDOA) method, implementation of sensors capable of performing in situ fuzzy classification of the measured sound source and implementation of a microwave radar signal acquisition and processing capable of detecting passing vehicles' speeds and direction. Additionally, the thesis demonstrates that embedded computation and communication technologies for Smart Environments have become capable enough to perform complex computational tasks required to collect and process situational information in a distributed manner. The thesis also explores new and innovative ways to use this technology for practical applications. Demonstrated in Publications I, II, III,, IV and V. Author's contributions are discussed in more detail in the Chapters 4 and 5 and 6.

1.3.1 Demonstration of a state of the art wireless sensor network at Purple Nectar

During November 2016 author together with ProLab's team presented a distributed wireless sensor network solution at Dutch military exhibition Purple Nectar. Purple Nectar is an annual experiments- and demonstration environment, organised by the Dutch Ministry of Defence, with the objective to show decision makers and end users the possibilities of state of the art technological solutions. Prolab presented a WSN for military SA solution, capable of detecting military vehicles and personnel. The solution was developed for European Defence Agency (EDA) project - IN4STARS 2.0 (Information Interoperability and Intelligence Interoperability by Statistics, Agents, Reasoning).

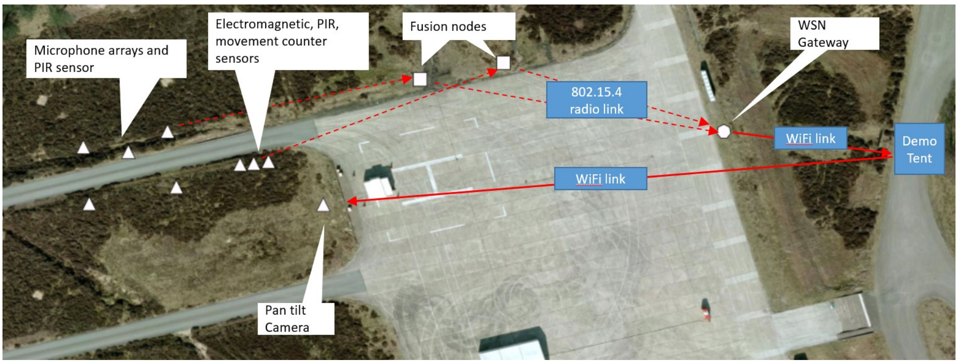


Figure 2 – WSN layout at Purple Nectar. The white triangles depict sensors, squares depict fusion nodes and hexagon depicts the gateway. The WSN communication is depicted with dashed lines and WiFi communication with solid line.

The WSN presented by Prolab team at "Purple Nectar" consisted of 8 sensor nodes, 2 fusion nodes, 1 relay node a single gateway which also acts as a subscription server and a pan-tilt camera. The 8 sensor nodes included 4 linear microphones arrays for noise source detection, 1 magnetic sensor, 1 video based people counter, 2 passive infrared movement detectors and a pan-tilt camera (which emulated a UAV, as an actual one was not allowed to fly). WSN layout can is depicted on figure 2.

The first of the fusion nodes uses angles to the noise source produced by four distributed acoustic array sensors for input and computes a position of the noise source. The acoustic arrays were also capable of classification of the noise source, so in case the object detected is classified, the information is also aggregated with the fusion result. The image of an acoustic array sensor is depicted on Figure 3. An acoustical sensor consists of 6 microphones, placed in an aluminium square tube. The sensor computation and communication components are placed in a box just below sensor array.

Figure 3 – WSN microphone array sensor. The linear array itself consists of 6 microphones, the distance between each microphone is 10 centimetres.

Figure 4 – WSN sensor set consisting of a motion detector, magnetic field change detector and a video based moving object counter.

The second fusion node uses inputs from motion detector, magnetic field sensor and moving objects counter and estimates whether a vehicle or human was detected and the number of the objects detected. The three sensors for motion detection, magnetic field change detection and counting moving objects can be seen on Figure 4.

The pan-tilt camera sensor included to the network was able to respond to subscriptions from both the fusion nodes inside the local wireless network and from external users in order to produce images from subscribed locations. During the Purple Nectar the area where wireless sensor network was deployed was also covered by Thales Squire infantry radar, which upon detection of contacts, subscribed to the images from pan-tilt camera. During the exhibition, the network was deployed and its operation was demonstrated in harsh weather conditions for a period of 8 hours. During the exhibition, from time to time Netherlands military vehicles and personnel moved through the area monitored by the WSN, their detection was demonstrated in THALES exhibition tent on simple graphical user interfaces, see Figure 5. The user interface was designed to run on any computational platform, provided it supported up-to-date web browsing. The user interface had two main windows, one for a map with sensors and events and another for log with description of recent events. On the figure the sensor nodes are depicted with blue stars and events with red pictograms. When an event was detected by sensors, the pan-tilt camera imitating an UAV would capture an image of the event. On the user interface the image thumbnail is depicted near the event and a reference to the image is stored in the log. It was also possible to add different map layers to the user interface, however, for better overview of what was happening in the field of sensor networks view, this option was not used in practice.

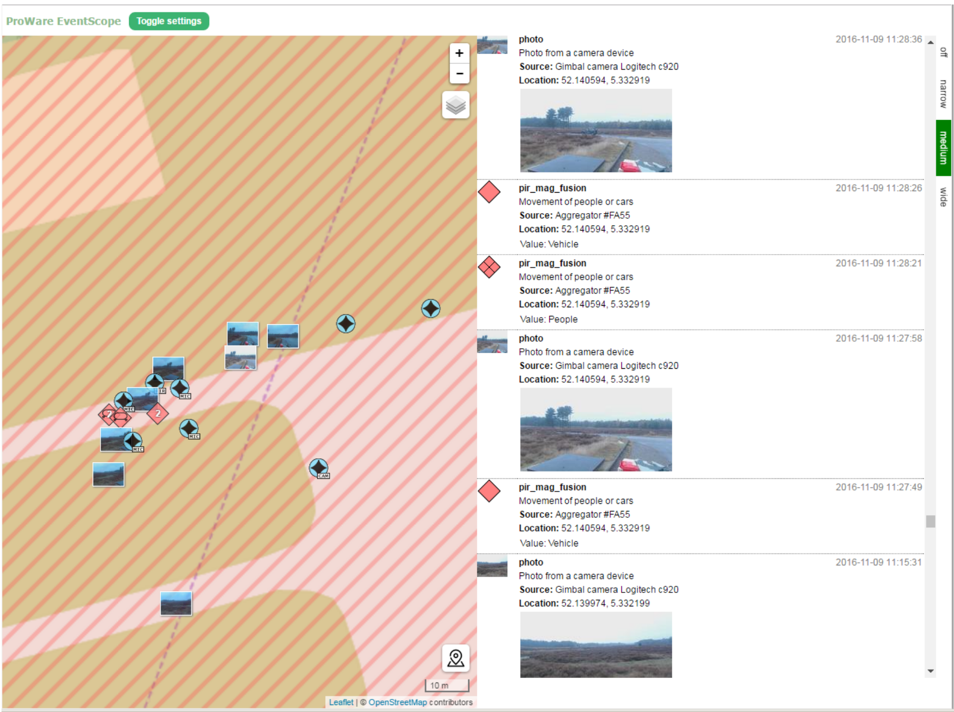


Figure 5 – Graphical user interface for ProWare. User interface has two main windows, on the left is a map over observed area and the logged events are listed on the right. Sensors are depicted with blue stars. Events are depicted with different red pictograms. The images captured by pan-tilt camera are depicted with small thumbnails.

The sensor network presented at the Purple Nectar made use of most of the novel concepts discussed in this thesis. The ISDP algorithms were custom designed for each sensor.

Among INDP techniques, there were two different fusion algorithms presented and two aggregation algorithms. First fusion algorithm received data from microphone-array sensors. For this fusion algorithm also time-alignment algorithm and selection algorithm of temporally suitable elements from streams were used. The second fusion algorithm combined data from movement detection sensor, magnetic anomaly detection sensor and video based object counter sensor. This fusion algorithm required only that requirements for validity of sensor readings and overlap of validity intervals were satisfied. For both fusion algorithms the validity intervals for input data and on-line validity checking was used. Altogether, even if the sensor network presented at Purple Nectar was not very large, it could definitely be described as a SoS composed of CPS, as it contained both different sensor technologies, computational platforms for sensor nodes and also Fog Computing and Cloud level support. By having a mesh topology and both ISDP and INDP examples present the sensor network part was a good example of MC for SE where D2D paradigm is exploited in order to obtain good SA. However, this all would not have worked without the application of the concept of mediated interactions. The concept of mediated interactions was implemented in the middleware solution ProWare. This allowed the production of situational information from heterogeneous wireless sensor network via online validity and consistency checks.

1.4 Statement of novelty

The main novelty of this thesis is the application of the concept of mediated interactions in Smart Environments. In contrast to existing methods, described in Chapter 2, which outlines state of the art of related work, this thesis applies the concept of mediated interactions for managing collection and exchange of situational information in Smart Environments.

Other novel aspects of this PhD thesis are:

1. The alignment and selection algorithm for improving consistency of situational information in distributed ad hoc WSN.
2. Application of on-line validation of situational parameters in distributed systems.
3. Application of several edge computing examples and distributed algorithms presented in Publications I, II and V, helping to demonstrate the usage of mediated interactions for improving the quality of whole system. All examples of sensor data processing are specifically designed for operation in WSNs.
4. Demonstration that advanced concepts used in IoT for Smart Environments can be applied in and can significantly advance both Situation Awareness and Intelligence Surveillance and Reconnaissance systems.

1.5 Organisation of the thesis

Chapter 2 starts with providing a definition for SE and then gives an overview over state of the art communication technologies for enabling SE. The technology overview includes different low power technologies for both wide area, short range and personal area networks, also a brief overview of military wireless sensor networks is provided. Next, this chapter will introduce several problems related with wireless distributed ad hoc sensor networks and gives state of the art overview how these problems are tackled in scientific literature. The chapter ends with a section about SA, explaining what is SA and what are special requirements are imposed by SE for SA applications.

Chapter 3 describes necessary building blocks required for building up SA by utilising Smart Environments. The concept of Smart Environment is tightly related to the field of Systems of Systems, which consists of cyber-Physical Systems. Hence, the next two consecutive sub-chapters are dedicated for explaining both concepts. In next sub-chapter, the concept of mediated interactions is explained, as it is used for the management of interactions and information flows within the networks of sensing nodes. The mediation process is controlled by publish subscribe method, which is explained in next sub-chapter. The cognitive agents subscribe to information services provided by Smart Environments. The information collection and exchange for SA between distributed nodes in turn is based on situation parameters. This chapter also provides definitions for the concepts of situation parameters and their validity and relevance and how situation parameters are used for ISDP and INDP algorithms. Finally both ISDP and INDP are explained.

Chapter 4 gives an overview of the necessary theoretical concepts for modelling mediated interactions for SE applications required for SA. The chapter starts with a short section describing the previous work and then continues in next section with short overview of the requirements for modelling mediated interactions. Then the next section continues with a description of several technical terms (e.g. temporal and spatial validity, relative consistency, overlap of validity intervals, simultaneity and end-to-end delays) necessary for achieving contextual consistency of mediated data in Smart Environments. Finally this chapter ends with sub-chapter for describing how network nodes and mediated interactions are modelled. The latter sub-chapter explains the concept of time-selective communication an provides description of the alignment and selection algorithm.

Chapter 5 describes three main experiments and a number of future possible use cases for Smart Environments. The first sub-chapter describes an experiment conducted in a military context. The experiment describes a use case for military application of a wireless ad hoc sensor network and demonstrates its applicability in a real world scenario. The second sub-chapter describes an experiment, which investigates bandwidth usage and its dependence on contextual constraints used for the mediator. Then, the third experiments demonstrates that applying selective mediation of interactions for sensor fusion increases its quality considerably. Finally an entire sub-chapter is dedicated for describing project SmENeTe2 and its future outlooks. SmENeTe2 (Smart Environment Networking Technologies) project was a project funded by Archimedes foundation established by Estonian Government. The Section 5.4.2 provides a short descriptions of 6 possible future use cases of wireless sensor networks for SE.

Chapter 6 gives and overview of a middleware design provided for SE. While preparing this thesis, the author worked with a Smart Environment networking technologies design project (SmENeTe2). Both the experiments described in Section 5, the theoretical work presented in the thesis and the work done during SmENeTe2 project has led to these suggestions for the middleware design.

Finally, the Conclusion sums up how the different concepts defined and techniques introduced in this thesis are used for building up Situation Awareness in Smart Environments.

2 State of the art and beyond

This chapter starts with providing a definition for SE and then gives an overview over state of the art communication technologies for enabling SE. The technology overview includes different low power technologies for both wide area, short range and personal area networks, also a brief overview of military wireless sensor networks is provided. Next, this chapter will introduce several problems related with wireless distributed ad hoc sensor networks and gives state of the art overview how these problems are tackled in scientific literature. The chapter ends with a section about SA, explaining what is SA and what special requirements are imposed by SE for SA applications. Sensor networks are the main source for acquiring situational information in SE. Contemporary state of the art sensor-, embedded- and wireless technologies allow to design networks consisting of a large number of heterogeneous sensor nodes, which are rapidly deployed in many different environments. There are many types of sensor nodes and networks, each type with its own limitations, as more often than not, the requirements of information collection for SA cannot be realised by only one type of sensor or networks.

2.1 Smart Environments at the edge

A smart environment is a concept of *a physical world that is richly and invisibly interwoven with sensors, actuators, displays, and computational elements, embedded seamlessly in the everyday objects of our lives* [49]. It is an ecosystem of interacting objects that have the capability to self-organise, to provide services and to manipulate and publish complex data in order to provide the users with better situation awareness about their surrounding environment. The examples of Smart Environments typically include smart homes, buildings, offices, factories, hospitals, cities etc. Some significant scientific works regarding sensor networks for Smart Environments are listed in papers [2] and [47]. Other concepts related to Smart Environments are Ambient Intelligence [25] and Federated Embedded Systems [68]. The concept of Ambient Intelligence is about *a digital environment that proactively, but sensibly, supports people in their daily lives* [25]. The concept of Federated Embedded Systems is *a constellation of devices that are part of and control different products, and that exchange data with each other and with external servers to the benefit of all, in such a way that no individual device is in control over the others* [68]. Although these concepts are somewhat similar, this thesis uses the concept of Smart Environments as its definition is broader and more general, thus implicitly relating better to the concept of SA.

In order for an environment to be smart it should firstly be able to sense the different physical phenomena of interest happening inside the environment and secondly to provide some useful service on sensed data. For example in the case of a smart city [106], the smart environment could be concerned with hundreds of sensors carrying out the monitoring of environmental factors such as air quality, noise in green areas and near streets, density of vehicles and pedestrians in streets etc. On the other hand, examples of Smart Environments can be found with very few sensors, for example [80] applies a single wearable device, a smart phone and Bluetooth beacons to detect a current situation of a user. This is a good example to demonstrate that in case of simple applications only a few sensors might be required to create smart applications which are able to react to some simple stimuli. These sensors often include simple individual Internet of Things devices which are directly connected to the general internet. Although, it may seem that the definition of Smart Environment implicitly implies an expectation of a large number of distributed embedded devices deployed over a significant geographical dimension. The significance of geographical dimension can have different meanings. For example, for low power and low range communication solutions even few tens of meters can already be considered a significant distance. On the other hand, one could easily also imagine a case where tens and tens of thousands of sensor nodes are deployed that span over tens of kilometres. In this case one has to assume a communication solution with longer transmission range, e.g. hundreds of meters to several kilometres. However, regardless of distances and total number of sensors, it is the number of sensors that each node in the network can communicate with that is important. A larger number of interacting sensor nodes requires scalable solutions to organise and manage interactions between the network nodes and in-network data processing. By definition, the applications of the Smart Environment, cited in the beginning of this section, require that sensors also provide local services for the information consumers within the Smart Environments. This can lead to the need to exchange data among the Smart Environment devices themselves while they are detecting and identifying the physical phenomena of interest. This is especially useful when many nodes monitor the same physical phenomena, which can easily happen in Smart Environments. How the information about the detected phenomena is combined and built up with data exchange is explained in section 2.4.

2.1.1 State of the art sensing capabilities

The commercially available selection and variety of sensors available for monitoring applications is huge. The sensors are able to monitor a range of various physical modalities. There are both passive and active sensors. Examples of more prominent passive sensors can under favourable conditions detect even both humans and vehicles via heat capacity [32], seismic vibrations [11], structural health monitoring [59], magnetic field changes, etc. Newer active sensor technologies include camera based sensors for activity detection [57], traffic radars for urban traffic estimation [27], acoustic arrays and even hyper-spectral cameras. This thesis also considers military applications for Smart Environments. Some examples in this domain include sniper detection (described in Publication V) or large projectile positioning, perimeter control (described in Publication III), soldier medical information [134], different weather and environmental monitoring sensors for real-time weather forecasting [69]. It is also possible to use Smart Environment solutions for positioning of friendly forces [60] to distinguish them from opponents and to better coordinate resources [61]. The list of useful applications could go on. However, the existing solutions are traditionally expensive and sometimes only designed for very specific use cases and are not compatible with systems developed by competitors.

2.1.2 Sensor network architectures for Smart Environments

Situation awareness applications can make use of three different computing architectures: Cloud, Fog and Mist and/or combinations of them.This section will give an overview of the definitions of Cloud, Fog and Mist Computing (MC) used in scientific literature.

In cloud computing architecture, the data from sensor networks is uploaded to so called "cloud". Cloud computing opens up possibilities to apply large data warehouses and computing resources for storing, analysing and carrying out resource demanding data processing. The data are uploaded to cloud servers where complex analytical algorithms can be used to detect events of interest by for example data mining from historic data. Typically the result is then conveyed and displayed to users by a separate application which can be on a personal computer or a smart phone. The user then decides what to do with the presented result. However, despite clear benefits, cloud computing still has scalability issues, when connecting IoT devices directly to the cloud. The number of devices directly connected (e.g., via 5G or other long range communication protocols that usually apply a star topology) increases substantially, the energy management, high bandwidth consumption and computing power can become limiting factors. In addition, the traditional Ethernet based networking stack, which is designed for cloud computing can be too complex for small sensor devices required in Smart Environments.

In order to solve several above-mentioned issues Cisco introduced a revolutionary concept of fog computing [18]. Fog computing is defined as a distributed computing infrastructure that is able to handle billions of Internet-connected devices [18]. In Fog computing architecture, the application logic has been moved from cloud to devices such as routers, smart switches and gateways. These devices, when used to process the data collected from IoT/SE end devices, are called the fog nodes. The general architecture of Fog computing is depicted on Figure 6.

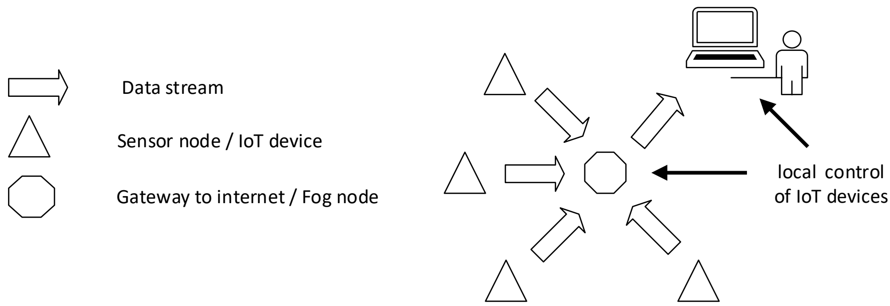


Figure 6 – General architecture of Fog computing.

The Fog computing architecture considerably reduces latency in case IoT devices require feedback from application logic, and also lessens considerably the requirements for bandwidth regarding connectivity to the cloud. Fog nodes orchestrate and decide when and how IoT applications are executed, reporting intervals, etc. Fog architecture is especially efficient when local control of several devices is required. Some good examples are heating ventilation and air conditioning (HVAC) and lighting systems in smart buildings where it would be inefficient to upload user requirements and control rules to the cloud.

One of the underlying principles of the fog architecture is based on Edge Computing (EC) in which the services are hosted within the edge devices inclusive of the gateways, routers, and access points. However, most of the Fog Computing architectures still rely on a star topology, which ultimately, with increase of the IoT devices, leads to a bandwidth congestion. The next logical step is to push the intelligence even further to the edge - to the embedded devices themselves. This has led to a concept of MC [112].

MC complements the Fog but also develops its concepts even further. As two of the most important motivations behind fog and MC are firstly the reduced bandwidth, when communicating data from Smart Environments up to Fog and Cloud nodes and secondly the latency, when feedback between either local nodes or between different levels is required. How MC nodes will start to exchange data among each other is depicted on Figure 7.

Often the SA applications require several IoT devices to operate in collaboration in order to achieve a specific goal. In Fog computing architecture this is achieved by a central fog node tasking several edge nodes and receiving feedback from each of them. Moving over to the MC architecture, the role of more powerful central fog node is reduced and mist nodes must start exchanging data and information in feedback loops among them-

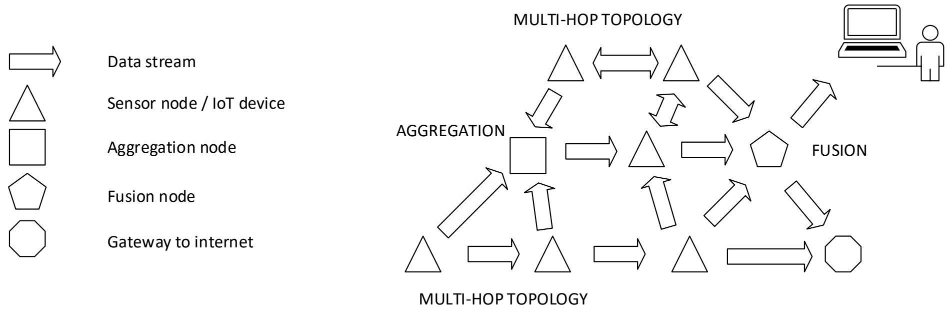


Figure 7 – General architecture of Mist Computing.

selves. The mesh topology is now used in order to solve the same collaborative tasks instead of fog nodes using star topology. One motivation behind this is even more reduced latency when edge nodes can directly communicate with each other. Hence, as depicted in Figure 7 with MC architecture, also the user or information consumer could be directly connected to any node (provided availability of architectural compatibility). Another motivation is removing a central single point of failure. On the other hand, the concept of MC also introduces several new challenges. The edge devices can no longer simply by automated, but become autonomous. The embedded devices need to become aware of their local context (location, neighbouring nodes, surrounding Smart Environment, time). Autonomy in turn requires both self-awareness and situation awareness.

In Publication IV the authors explain that in distributed computing it is more efficient to push the computation closer to the physical phenomena of interest. The data to decision (D2D) paradigm is applied. This thesis also demonstrates this by an example and proves that it is considerably more effective to process data locally in collaboration at the edge than in cloud (somewhere at the other end of the world). Pushing the needs (and instructions) closer to the situation which needs to be managed holds many benefits (e.g., low latency and reduced bandwidth usage). Of course this requires the processing entities to be able to adapt to the changed needs/instructions.

The concept of MC makes it even possible to combine the sensor networks and autonomous vehicles into systems of collaborating agents. This approach was briefly analysed by author of this thesis in Publication III. In cited work the author viewed an autonomous vehicle and sensor network as a unified system of system, where the sensor network could task or draw attention of the autonomous vehicle to certain aspects of which only the sensor network is aware. The autonomous vehicle could take this into consideration and recalculate its future trajectory.

Concept of MC is also related to the EC computing paradigm. EC is a distributed computing paradigm in which substantial computation and data storage resources are placed at the Internet's edge, either in or in close proximity to IoT devices, embedded systems, sensors or mobile devices to improve the response time and to save the bandwidth [125]. It is a more generic term as it can refer both to Fog as well as MC paradigms, in this sense, MC can be considered a part of EC. However, while MC also occurs at the edge it is more specific and refers to very low power embedded and networked devices in ad hoc and mesh networks, it brings intelligence to the edge through both smart algorithms (run on low computationally capable nodes) and collaboration via mesh topology between the network nodes. The EC in turn brings computationally more power to the IoT nodes and sensor networks (both through more computationally powerful nodes and by distributed computing). In Publication IV, the authors describe that both computing paradigms push the decision capability closer to the data production - leading to D2D paradigm. The D2D paradigm in turn reduces both the latency between situation detection and decision making and bandwidth usage between the edge devices and cloud. The situational information provided by the edge nodes is prerequisite for the SA and decision capability. Computing more abstract situations or higher level situational information already at the edge of the network where the same network nodes also carry out the sensing tasks is one of the main incentives of this thesis. It does not have to be a single node that constitutes the EC - i.e. the edge node. The higher level situations may be computed in collaborative fashion at the edge of the internet by several edge nodes.

2.2 State of the art overview of enabling communication technologies

Contemporary IoT and Smart Environments applications make use of a variety of communication technologies for sensor data collection. This thesis focuses on wireless communication between distributed sensor nodes. Such wireless sensor networks have also been known as "smart dust", which is why it is often referred that such sensor networks could consist of hundreds and even thousands of miniature smart nodes that can be scattered over large areas e.g., from an aircraft. While state-of-the-art technology can provide very small-scale systems, in practice it is difficult to use them in such ways. The main advantage of wireless sensor networks is the idea that there is no need for almost any infrastructure and central control. They lack a single point of failure, can be quickly installed, forgotten and used when needed. The information flow through the wireless sensor network is dynamically created according to the available resources, which are the deployed sensor nodes themselves or the connections to other networks such as the Internet. Each sensor node is a stand-alone system that monitors, processes and analyses analogue signals from the outside world.
However, in addition to WSN, there are many other technologies out there. The communication technologies that can be applied for IoT and Smart Environments are divided into classes according their transmission range. The different ranges and data rates of some example technologies are depicted on figure 8.

2.2.1 Low Power Wide Area Networks Technologies

In recent years, the low-power wide area network (LP-WAN) technologies have been gaining significant attention and are emerging as enablers to support massive machine to machine connectivity for the Internet of Things. These technologies can operate in coexistence with the traditional cellular and short-range wireless technologies to enable connectivity for low power and low data rate devices. The LP-WAN data rate ranges from 0.3 kbit/s to 50 kbit/s per channel [1]. LP-WAN technologies are applicable to a large range of IoT scenarios such as smart metering, smart parking, smart homes, smart tracking, smart logistics, e-health, industrial automation, etc. LP-WAN technologies include both licensed and unlicensed technologies. Some examples of licensed LP-WAN technologies are narrowband Internet-of-Things (NB-IoT), extended coverage GSM (EC-GSM) and LTE Category M1 (LTEM1). These technologies are also included to the 5G standard. For example, NB-IoT was standardised in 2016 by the Third Generation Partnership (3GPP) and is now one of the emerging technologies in the LPWAN area. At the same time, there are several unlicensed technologies, such as Ingenu, LoRa, SigFox, Weightless-SIG, Telensa, etc. Each of these technology examples are employing various techniques to achieve long-range, low power operation, and high scalability, e.g., spread-spectrum technology with a wideband and data rates using encoded packets (e.g. LoRa) or (ultra)-narrowband technology

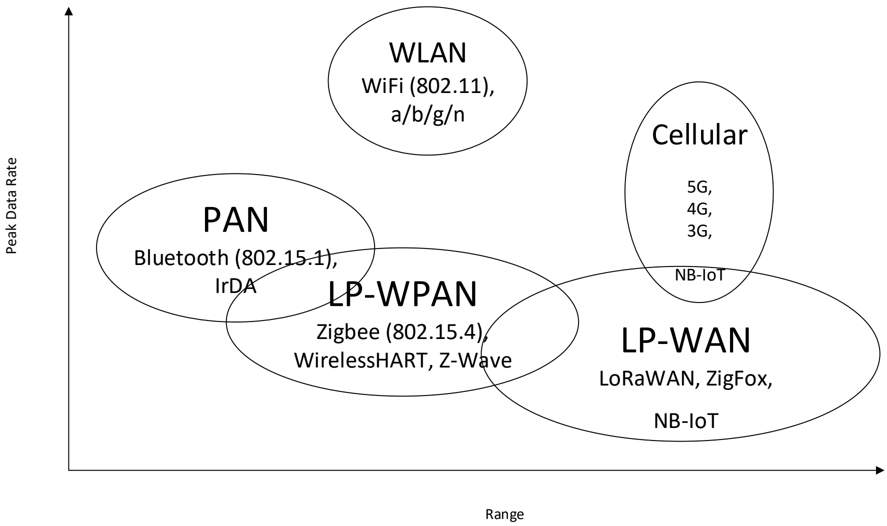


Figure 8 – Range vs datarate of different networking technologies for smart Environments. PAN - Personal Area Networks. WLAN - Wireless Local Area Networks. LP-WPAN - Low Power Wireless Personal Area Networks. LP-WAN - Low Power Wide Area Networks.

and slow modulation rate for extended range (e.g. SigFox). However, LP-WAN technologies are still considered in their early stage, needing on one hand, practical deployments and measurements, and on the other hand, deep theoretical investigation for modelling and optimising system performance. Also, emerging applications that can be enabled by LP-WAN technologies and implementation challenges therein need further exploration. Another aspect is that LP-WAN networks assume an existing infrastructure, while there are also networks which can operate without existing infrastructure.

2.2.2 Low Power Short Range Networks Technologies

The range of low power short area network technologies are more difficult to classify. Broadly, they can be divided into two classes, the wireless networks based on IEEE 802.15.4 standard and other short range personal area networks (especially suitable for wearables), such as Bluetooth, NFC or IrDA. The low power short range wireless communication technologies based on IEEE 802.15.4 standard includes technologies such as Zigbee, WirelessHART, ISA100.11a and several other technologies that are based on IEEE802.15.4 standard. The networks based on IEEE 802.15.4 standard, make use of unlicensed frequency bands.

2.2.3 Smart Environments based on IEEE 802.15.4 standard protocol

Smart Environments, based on IEEE 802.15.4 standard use wireless communication for connecting individual devices. The standard defines operation for Low-Rate Wireless Personal Area Networks (LR-WPANs), also classically called Wireless Sensor Networks. There are many different types of wireless communication solutions to connect sensor systems today. Typically the networks based on IEEE 802.15.4 standard are not based on IP communication protocols (However, there are exceptions such as 6LoWPAN (IPv6 over Low -Power Wireless Personal Area Networks [129]), which differs from 802.15.4 networks by including encapsulation and header compression mechanisms that allow IPv6 packets to be sent and received over IEEE 802.15.4 based networks.). The low bandwidth and non-deterministic delays, exhibited by most of the protocols regulated by IEEE 802.15.4 standard, mean that methods for reliable data transport used in internet are not applicable. The hardware, software, communication protocols for WSNs are especially designed for conserving energy, so the nodes in WSN run on low energy processors and possess very little memory. For example a typical embedded system such as 8-bit Atmega128 has 128KB of program memory. A typical packet size for WSN communications is between 17 and 256 bytes. The IEEE 802.15.4 protocol allows sensor systems to communicate over a relatively short distance width low on power consumption. The distances for 802.15.4 span over from few tens of meters to few hundreds of meters. WSNs can be considered as one example to construct systems of connected things in Smart Environments.

IEEE 802.15.4 standard defines operation for WSN which is ad hoc and mesh, the former means that network nodes can join and leave spontaneously an existing network and the latter means that all nodes in communication range can potentially communicate with each other. This means that a wireless ad hoc mesh network consists of a group of nodes communicating with each other with no need for an access point or a central coordinator. In case the data needs to be conveyed to the cloud server (connected to general internet), then at least one of the nodes should be connected to a sink node (Gateway). In military parlance such network nodes are usually considered unattended. (often called UGS - Unattended Ground Sensors). The ad hoc property also means that these networks do not rely on existing infrastructure to establish the network. No routers nor access points are needed for an ad-hoc network. Instead, nodes are dynamically assigned and reassigned based on some dynamic routing protocol (e.g. Dynamic MANET On-demand (DYMO) Routing protocol). The ad hoc mesh network allows real time communication between the nodes as shown in Figure 9. The figure also demonstrates that not all nodes can communicate with each other, only those which are within radio coverage.

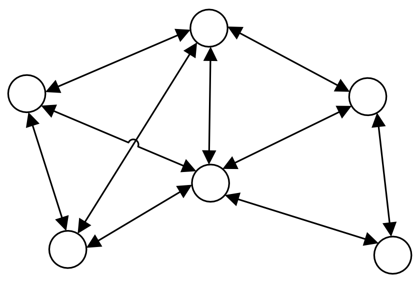


Figure 9 – An architecture of a mesh network.

The infrastructure less architecture also makes it possible to introduce distributed in-network data processing (INDP) (or distributed decision making) as otherwise in classical infrastructure-based networks the decision engine is usually infrastructure based (as in Fog computing, explained in Section 2.1.2). This also provides redundancy-by-design, as the entire network is not dependent upon an infrastructure to carry out their tasks. This is a primary advantage of the ad hoc mesh sensor network and also enables to introduce the concept of Mist Computing (MC). The MC concept pushes/distributes application logic to the end devices. This holds many advantages compared to both Cloud and Fog computing (explained in Section 2.1.2): local network adaption according to traffic load, shorter time delays, lower bandwidth requirements etc. In this way, Mist computation becomes the foundation for Smart Environment applications. The main drawbacks of WSNs are related to energy management and power considerations. However, this falls out of the scope of this thesis. The energy efficiency in WSNs is very well researched area and the reader is referred to surveys conducted by [4, 73].

The thesis also provides some sample case studies. The cases are described in Section 5.4.2. The use cases demonstrate Smart Environment applications in the context of the project SmENeTE2. All cases assume computation within the network. A consumer node in the network is assigned a task to either aggregate or fuse data from several other spatially distributed sensors. Mist Computing concept allows multiple roles for network nodes, i.e. data providers can at the same time be a relay node or consumer node (aggregation node, fusion node or actor node). However, for the successful integration of data (e.g. sensor data fusion) from several data providers nodes, the methods for checking data validity and consistency are required.

2.2.4 Military wireless sensor networks

While sensors as metering devices can be encountered everywhere, at home in smart devices and as part of heating or security systems, in industrial machinery and robots, and also in self-driving cars around the world, the term "wireless sensor networks" has been more commonly known in the military. Like other autonomous or unattended systems, they have been used in places where human labour is either too expensive, drab or dangerous. One illustrative scenario of using a wireless sensor network could be a perimeter protection. Unattended sensor-nodes connected to the wireless network, equipped with different sensors, are temporarily mounted around a camp or a Forward Operating Base (FOB) to detect the movement of the adversary forces or vehicles. The sensors are combined so that, after the initial detection of a hazard by low-power sensors, more sophisticated sensors are activated, which perform classification and possibly even video and image processing identification as described in Publication IV. Where possible, the sensor system may even order observation on an unmanned drone as described in Publication III. An overview of the resulting situation, both sensor information and photos, is transmitted via a low-power and encrypted wireless channel to a patrol team who, being aware of the overall situation and context, will be better placed to make the right decisions. The military application field also sets some important requirements that differ from usual civilian use and which are often contradictory for high-tech solutions. Sensor systems must remain undetected and operate autonomously over long periods of time, meaning that their physical dimensions are limited and their power consumption must be well optimised. In order to perform sophisticated signal analysis and classification, the sensor systems must be computationally capable. The information collected and generated by the sensor network must reach the user at the right time, which means that both sensor systems and network link duty cycles and communication sessions must be dynamically self-organising and adapting to a dynamic situation. Such requirements are contradictory and finding a compromise may not be easy. As a result, the power consumption of both the computational part and the radio part is usually minimised. There are different ways to achieve low energy usage. For example, sleep periods are used during which most functionality is disabled and the node is in a standby mode. Most of the power that comes with a sensor node nowadays is very often comparable to a standard two to three AA batteries. If needed and if possible in an application also different renewable energy solutions could be employed - solar energy, vibration, electromagnetic radiation.

2.3 State of the art approaches to ensure timeliness of sensor data in Smart Environments

Detecting and monitoring complex time-variable (real-time) situations in SEs requires methodical consideration of temporal aspects, especially when the network consists of ad hoc wireless sensor nodes which are distributed, asynchronous and autonomous. The situational information used for building up SA for comprehending and predicting specific situations in time and space must be temporally and spatially consistent, as described in Chapter 4. For example, combining speed and position data of a moving vehicle to project the next position at a certain time instant requires that time instants of measured data are consistent, meaning within the specified simultaneity interval. When using IoT devices i.e. nodes in a distributed sensor network to collect and transport the data to the user, the number of possible sources for time delays and variations in delays is large. Signal processing, data queuing, radio transmission related delays etc. On top of this, the ad hoc mesh network architecture adds time variable end-to-end communication delays originating from the changing network topology. In case the cloud infrastructure is used to transport the data to the users who do not have direct access to the IoT networks then additional delays must be considered. This is despite computing in cloud infrastructure, utilises synchronised UTC (Universal Time Coordinated), which allows precise time stamping to avoid the ambiguity with the delays at cloud level. Instead of transporting all sensor data to a central cloud server, this thesis applies the paradigms of Edge Computation and INDP as described in Publication II. The former is about performing as much computation close to the on-going situation as possible (either in sensor nodes themselves or in a distributed fashion close to them) and the latter is about carrying out data fusion and aggregation within the network to mitigate bandwidth and energy scarcity. Both are necessary for handling of situational information in a timely manner. Both methods assume that data collected by a distributed sensor network is processed already within a network by means of distributed data fusion and aggregation by network nodes themselves. Carrying out computation within the network reduces considerably latency and increases solution resilience and the reliability of situation detection.

However, in many contemporary applications, the issue of timing correctness is reduced to performance requirements. On top of this, in most sensor networks, the data acquired by sensors are processed in the same sequence as they arrive. Often they are even timestamped at the arrival according to the same order as they arrive. This could be acceptable if the overall structure of the network is known at design time - and the communication paths for all sensors is fixed and similar end-to-end transport times in order to avoid the out of order arrival. However, considering communication delays and jitter in the communication delays in communicating sensor data is critical, there are also scientific references that do take delay of sensor information into account. For example, Izadi et al. [54] present a data fusion approach which distinguishes low-quality input data from good-quality input data by assigning weights on sensor readings. The network delay is considered as one of the factors in the computation of weights, such that sensor readings with longer delays have lower influence on fusion result. This approach favours the freshest data and discards the opportunity to use delayed data that may be of high quality and better suitable for multisensor fusion. Other examples of prioritising data freshness can be found in the papers that analyse quality-of-service (QoS) aspects in WSNs. A good survey of the state-of-the-art QoS techniques for delay handling and reliability mechanisms is provided in Al-Anbagi et al. [3]. Similar overviews of WSN solutions for manufacturing and industrial control are given by Zhao [141] and Diallo et al. [33]. All solutions described in these works demonstrate reasonably good time-aware performance in handling time-critical data and time-sensitive communication. These works consider the QoS aspects, such as real-time constraints and data freshness are considered most important in these surveys. Another approach to ensuring timeliness in conventional systems of sensor networks is to design them so that the end-to-end delays caused by disparate communication paths meet the specified deadlines [52]. The authors in [52] use CSMA/CA based MAC delay analysis to construct communication paths to sinks that are less congested. However, such strategy assumes existence of several sink nodes for load balancing, which also must be connected in order to work out the central strategy. Such an approach is not applicable in this thesis, as DIL and ad hoc WSN, where INDP nodes and sink nodes might not always be able to share their current load data. Another possibility is to modify the network structure to optimise the delays. One such method is proposed by Cheng et al in [21]. In their paper, they present a delay-aware network structure, which organises into clusters according to the existing information flows within the network with the goal of minimising the delays in clusters. However, the arrangement of information flows in ad hoc networks is inherently difficult to control, hence the method suggested in Cheng et al. [21] may not applicable in real-life wireless sensor networks.

For the sake of providing better understanding of timeliness capabilities of sensor networks [136, 99], it may be beneficial to consider probabilistic methods for analysing traffic flow aspects, such as end-to-end delay, jitter and throughput. However, both works [136] and [99] also consider it necessary to highlight that in most practical cases, the worst-case bounds for end-to-end delay in WSNs are not applicable, due to their self-organising nature. This thesis emphasises that in ad hoc sensor networks and especially in networks for collecting SA information, the data consumer must be able to analyse the timeliness of the data online [107] and to determine how long the data are usable. For this current thesis uses augmenting situation parameters (explained in section 3.5.1), computed by sensor nodes, with validity metadata (explained in section 3.5.3) and time selective strategy (explained in 4.4.1) for in-network processing, which can handle more variability in end-to-end delays, but requires a means to compute the delays accumulated (explained in 4.3.6) during the data transport through the network.

The primary sources for non-determinism in timing and delays in contemporary sensor network communication systems include transmission delays, packet losses, queuing for transmission, nodes contest for radio frequency medium and clock drifts and jitters in individual nodes of the network. Delays that are related to radio transmission originate from send time, access time, propagation time and receive time. These delays are a well-researched area in traditional communication networks [71]. In wireless ad hoc sensor network solutions, where global time synchronisation is not feasible (or accurate), the radio transmission related non-determinism can be alleviated for low number of hops by applying state-of-the-art transceivers (e.g. using IEEE 802.15.4 protocol), which allow for modifying the contents of a packet after packet transmission is started by utilising delay computation method described by Maroti and Sallai in [88]. Timing challenges in wireless sensor networks also include packet losses, which can happen due to dynamically changing network structure and unreliable wireless links [3]. The nodes may spontaneously join or leave the network, radio frequency interference from other sources may influence the wireless links (which may force the dynamic routing protocol to find different paths) and also mobile nodes must be considered. A good overview of different routing protocols for wireless sensor networks is given by [63], where authors demonstrate with simulations that in terms of average throughput, average energy consumption, and total packet received at sink Dynamic MANET On-demand routing (DYMO) protocol performs best in comparison to other popular routing algorithms. The other popular algorithms

DYMO was compared against, were Ad-hoc On-demand Distance Vector routing (AODV), Bellman–Ford and Optimized Link State Routing (OLSR) protocols. In the networks and experiments used in this thesis a patented mesh networking technology for a MAC layer is used [113] that enables creation of large scale mesh networks. It enables direct device-to-device communications with an underlying routing algorithm based on the Dynamic MANET On-demand (DYMO) routing protocol [91]. High robustness is achieved via its self-healing ability and no collisions TDMA based on a clustering algorithm [113]. It is a modified MAC layer protocol for self-forming robust mesh networking, called BeatStack. The BeatStack algorithm decomposes the network into multiple smaller clusters where each node is given a time-slot for transmitting its packets. The clusters are formed based on link quality (RSSI level). The maximum number of nodes in a cluster can vary, ranging from 5 to 15. If there is more than a single cluster in a network, the communication periods for individual clusters are separated in time. Each cluster has its active period, when its members can exchange data. Clusters sleep between active periods. Only the dynamically selected router node stays active at cluster sleep periods and exchanges data with cluster partners, i.e., clusters that are adjacent to the given cluster. Cluster active period is divided into slots so that each cluster member can transmit its data during the slot allocated to it. Therefore, the end-to-end delay for data can be time varying. Single node analyses the qualities of the links to other nodes, based on the results of the analysis, the node forms or joins a cluster. In case the link quality changes, then also the cluster compositions may change. When a stable situation has been achieved, the network has stabilised clusters, then each node in a cluster has around 30ms to transmit its messages and Cluster router node can transmit accumulated packets when it is joining the communication session of an adjacent cluster. These numbers are very general because the only purpose of this section is to give a reader some understanding what is meant by the network that is used during the experiments is self-organising and why the end-to-end delays of even same communication paths can differ in time. Radio link quality depends on distance, interference from other electrical devices or any other random unknown influences from surrounding electromagnetic environment. In case the link quality is disturbed, the sensor nodes either fall out of cluster or may start switching between the clusters.

The communication delays in a network can arise also in situations where sending node is unable to transmit due to its periodic activation, low duty cycle or other network scheduling policies. The resulting queuing delays for transmitting messages to partner nodes must be taken into account, by incrementally computing message age (explained in section 4.3.6). Although the execution periods of the processes in network nodes may be highly deterministic, the messages are delayed and transmitted at non-deterministic times. This can cause the end-to-end delays to be highly unpredictable, the same applies to the sequence of data elements as packets may arrive out of order. Each time the system's structure changes due to changing goals by users or the environmental conditions, the network must adapt to the changing interaction patterns and delays.

Another aspect complicating timing analysis in an ad hoc sensor networks is unpredictability of the data generation by autonomous nodes. Although the data production is usually designed to be periodical, the contemporary smart sensor nodes are able to utilise the local situation and consider the behaviour of the monitored physical phenomena and the user requirements. Firstly, the traffic rates of produced data by sensor nodes depend on the application, sensor modalities and sensor process signal processing capabilities. For example, more intelligent and autonomous sensor nodes can avoid reporting altogether if the monitored situation is unchanged or report only as often as required by the rate of change of situation. Sensor nodes that monitor slowly changing physical phenomena or environmental aspects may not need as high rate of reports as other sensors measuring fast changing current or voltage spikes, or tracking a mobile object. Secondly, nodes in large scale networks often apply some kinds of duty cycling or other transmission scheduling policies to mitigate bandwidth and energy usage [19]. Making these decisions autonomously according the current situation, regarding environment or local energy level adds unpredictability.

2.3.1 Out-of-sequence data

The problem of handling out-of-sequence data for INDP and especially for sensor fusion is not well researched topic in scientific literature [67] and even less so in papers considering ad hoc WSNs. In the domain of multi-sensor data fusion, the related topic is called out-of-sequence measurements (OOSM) [67]. OOSM can be caused by variable propagation times for different data sources or by heterogeneous sensors operating at multiple rates. The problem becomes especially relevant in large-scale sensor networks consisting hundreds to thousands of disparate measuring devices, as the complexity of network communication increases and communication delays of data packages increase [84]. In the domain of multi-sensor data fusion, the solutions for this problem concentrate mostly on designing better filtering algorithms (e.g. Kalman filter or particle filter) that cope with measurements arriving only a single or a few steps later [67, 132]. This thesis considers those approaches not appropriate for in-network multi-sensor fusion in ad hoc WSNs. The delays in such networks can be considerably more unpredictable and orders of magnitude greater. It is also clear that existing approaches considering filtering and state estimation are often computationally complicated [84], especially in the context of low-cost embedded devices. Liu et al suggests to solve OOSM problem with artificial neural network solutions, however, currently these type of solutions are too computationally resource demanding for low-cost embedded devices. Some early examples that have evaluated solving out-of-sequence arrival of data for in-network processing in WSNs are Shi et al. [130] and Xiaoliang et al. [139]. While both papers consider OOSM filtering approach for discrete step delays, the former handles mixed and bounded delays from a single sensor and the latter deals with delays from multiple sensors with delay length of a single sensor data refreshing period (one step). These approaches are still in their early stages and not yet ready for DIL and ad hoc WSNs where multi-sensor fusion is considered.

A good overview of the existing data fusion techniques for WSN has been described by Yadav et al., [140]. However, none of the listed works in given survey neither consider variable arrival delays in ad hoc networks or streams that may result in out-of-sequence arrival to the fusion node nor give sufficient attention to other timing characteristics other than freshness of data. Some examples of distributed multi-sensor data fusion in WSNs are described in Bahrepour et al. [10] and Lai et al. [74], where events detected and respective sensor readings collected by individual sensor nodes are assembled by a fusion node. Nonetheless, these works do not discuss the validity or consistency of the input data from disparate sources for the fusion algorithms. A spatio-temporal alignment problem for low-level sensor fusion, similar to the approach taken in this thesis is described in [133]. The paper explains that spatio-temporal alignment is difficult due to synchronisation problems in distributed sensor networks, especially when sensor nodes with different update rates are used. In order to solve this issue, they take the sensor S_m with the highest update rate as a reference when searching for temporally compatible elements from other streams. This gives a maximum error of half of a S_m update interval. However this work limits its research to fixed strategy with choosing reference sensor with the highest update rate and they do not discuss cases with several sensors.

Simulating an ad hoc sensor network is a non-trivial task. Classical models for simulating distributed sensing systems often use abstractions at various levels to compensate for timing non-determinism [55]. Examples are lock-step synchronous models [58], fixed or no drift in individual clocks [30] and/or delays with fixed bounds [38] (which essentially models a subset of synchronous systems). This thesis considers the Q-modelling formalism [94] for the analysis of ad hoc sensor networks and as an underlying formalism for performing time-selective communications, as it naturally facilitates modelling of timing aspects of asynchronous communication and queuing delays across the communication paths through the network while considering the accuracy of data timestamps. One of the original purposes of the Q-model is to analyse timing correctness of inter-process communication of a collection of loosely coupled, repeatedly activated and terminating processes [96], where the objective of the time-selective communication is to select input data for the consumer process to be exactly from the desired time interval (not produced before or after that time interval – the most recent data is not always desirable). In this way, the time-selective communication used in autonomous and distributed real-time systems leads to decisions where some of the execution sequences and data produced by them are discarded and some may be used as inputs to another process several times [94].

2.3.2 Global time synchronisation in a Smart Environment

According to Edward Lee a CPS is an orchestration of computers and physical systems [78]. In this way, Lee considers a CPS a distributed system, which components have also a strong dependency to the real world processes and thus can by their very nature considered real-time systems. Such systems often control certain processes in the environment where they are located. Lee points out in [77] that "In particular, the passage of time must become a central property" and describes in [102] that real time data processing is essential. Integration of CPS into IoT and wireless sensor networks [79] have resulted in and will in future result in increasingly complex systems, where the central issues concern timing [42]. This means that instead of viewing an embedded device as a single solitary control system, the contemporary embedded systems are becoming increasingly more connected by different networks. However, classical solutions for coordinating activities in distributed systems, are usually based on a global notion of time. One way to agree on a global notion of time is to synchronise clocks of individual devices. This can be very challenging in case of low cost IoT devices and especially in resource constrained systems like WSNs as can be seen from reviewing an immense amount of the relevant literature on this topic [138], [117], [76]. When considering an ad hoc type networks of WSNs or IoT devices, the time synchronisation may even be infeasible due to mobility, intermittently available communication or low available bandwidth and constrained computational resources. One solution, which also is considered in the current work, is an untethered clocks approach, where every node maintains its own clock as it is, and keeps a time-translation table relating its clocks to other nodes' clocks [128]. In this case, every component in network is allowed to keep its own independent time counting system, including timescale/granularity, offset, drift, accuracy, stability [96]. This is also the approach that is considered feasible for the timing analysis of mediated interactions for this thesis, as elaborated in Section 4.3.6. The mediated interactions allow using validity constraints to filter out the required information, or to drop messages that do not satisfy constraints set by the individual processes. One type of such constraints can be a temporal interval. Complementary research on the topic of setting worst case bounds on time delays and processes can be found in papers by the Ptides working group in Berkeley University [37]. However, all these works concentrate on deterministic behaviour of CPS and do not consider emergent behaviour, which arises from autonomy and interaction with the social and physical world where systems (configurations) evolve in time.

2.4 Situation awareness via Smart Environments

This sub-chapter provides an overview of situation awareness theories and important concepts. Situation awareness research belongs to a field that strives to understand how a cognitive agent, usually a human, interprets the circumstances relevant or having an influence on the agent. The SA research stems back to a beginning of previous century, where one of the first attempts to formalise the situation understanding is by William Isac Thomas in his work "The Unadjusted Girl" in 1923 [121]. In this work, Thomas stated that *"individuals do not react to reality or facts, but rather their perception or personal "definition" of these situations and facts"*. This statement also differentiates between perception and comprehension by stating that human individuals interpret facts subjectively based on their perception of the world. Contemporary theories about SA started to emerge in the 70s, where one of the most famous works comes in 1960s from Boyd's work on the OODA loop [118], this can also considered a starting point after which the military community began to show serious interest towards the concept of SA . Obviously, due to its obvious relations to the effective decision making in the combat environment [104]. Hence, it is rather understandable that the most cited definition of SA comes from United States Air Force chief scientist Mika Endsley [39]: "*Situation awareness is the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future*".

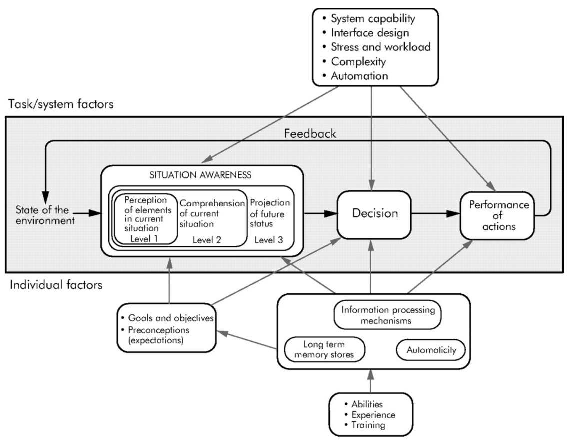


Figure 10 – Endsley's model of situation awareness (adopted from [41]).

The three levels of SA in Endsley's model are depicted on figure 10. Level 1 SA is perception of elements in current situation, level 2 SA is comprehension of current situation and level 3 SA is projection of future status. In more simple terms, Endsley's definition means that a human operator or a team of operators needs continuously updated (up-to-the-X, X depends on application, in some applications X can mean days, in other applications X can mean seconds) cognisance or awareness required to operate and carry out tasks in a certain environment [40]. This thesis investigates how this cognisance or awareness can be achieved via situational information which is provided by a Smart Environment.

The tasks of SA can be diverse, operation of some equipment (e.g. a mechanical tool or control of a nuclear power station), vehicles (e.g. a fighter aircraft, a semi-autonomous vehicle or just a bicycle) or in general maintain a system by detecting and predicting changes in ongoing situation [40]. Usually in these cases human operators acquire the required situational information via multiple gauges, screens and indicator lights or just by looking at them and reading the required information. In more advanced systems, the information can also be displayed on a single display with advanced information presentation methods or even in a form of augmented reality by the use of special glasses or a helmet [44].

There are also SA theories which state that the situational information may be distributed among agents residing in the surrounding environment [131]. In this work Stanton extends the definition of SA by stating that the term agent can be used for both humans and cognitive artefacts [131]. In [120, 131], Stanton and Salmon suggest that artificial distributed agents have some level of SA in terms that they can be holders of relevant situational information that may also have been learned about their surrounding environment. Stanton and Salmon suggest that the distributed SA is a system's SA, where agents may provide information to each other via interactions in order to avoid degradation of situational information and that this represents an aspect of the emergent behaviour associated with complex systems. Of course a suitable technology must be selected to collect and exchange this information from and between the agents who have the necessary pieces. For this aspect, this thesis suggests to apply the concept of mediated interactions. In comparison, the two above-mentioned models (Endsley SA and DSA by Stanton and Salmon) are not directly easily comparable, however, considering system view, the second model of DSA could be considered more suitable for analysing SA in SE.

However, in most cases the theories of SA assume that the information necessary for the level one SA (i.e. perception) is brought to the cognitive agent "on a silver platter". The classical SA theories do not usually handle instrumented sensor networks, information acquisition, signal processing and information transportation parts. In the following some examples are given to explain this statement. In the works of Mica Endsley's classical model of SA [39], the stage of primary information sensing and human information acquisition has been omitted by default. The reason for this was most probably Endsley's interest which have been focusing more on information perception and comprehension. Similar position has been taken by researchers form the area of Situation Calculus by McCarthy and Hayes in [82], by the researchers from the area of Situation Semantics originally started by Perry and Barwise in [13, 103, 31] and from area of JDL Information Fusion by Lambert et al [75].

Later, a theory of a framework for Cognitive Situation Control introduced by Jakobson [56] is one of the first to explicitly include situation sensing (along with instrumented sensors and situation recognition based both on sensors and reports from human agents) into the negative feed-back loop of Situation Control. The conceptual diagram of Gabriel Jakobson's framework is given on Figure 11.

The examples of this specific research cap also can be found also from outside of SA research [70]. In [70] the authors show how time delays in network affect the human machine interaction. In this work Kolar et al. bring out that large scale autonomous systems, which interact with humans, need new scalable communication, control and computation techniques.

Hence, the models and architectures of instrumented information sensing with sensor networks have still remained out of the primary focus of SA researchers. However, mostly due to technological advancements of both embedded hardware and communication technologies, there are plenty of reasons to research the methods and techniques for using sensor networks for situation recognition from the viewpoint of SA. With regards to SA this thesis identifies at least three aspects where SA theories could help to improve

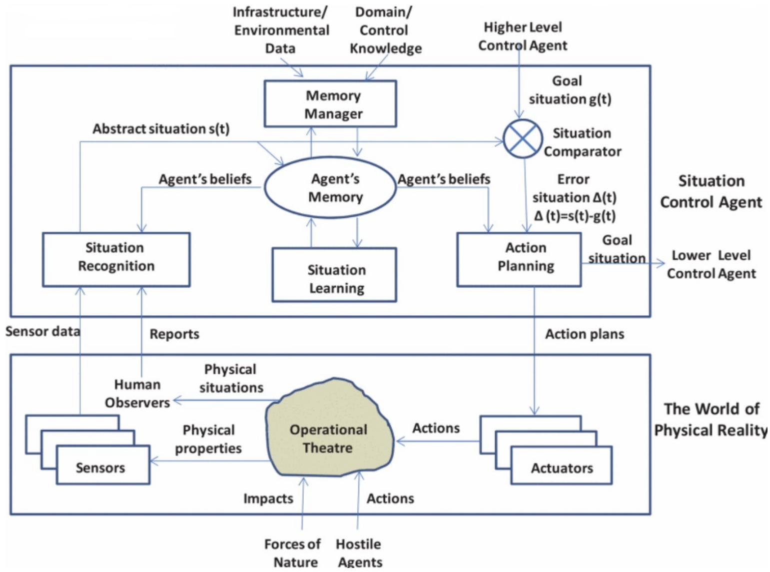


Figure 11 – Cognitive situation management (adopted from [56]).

both performance and functionalities of massive ad hoc distributed sensing networks:

1. move cognitive processes ("intelligence") closer to the edge (into periphery) of a massive distributed sensing network,
2. make sensing and information processing driven by SA needs (by describing new functional and performance opportunities for Smart Environments),
3. develop methods for controlling and advancing smart information sensing with models and architectures of cognitive information processing. In the classical situation awareness theories, information sensing has been looked as a process of passive instrumentation that served the "information consumer", who in turn exercises high cognitive levels of situation awareness and control.

This thesis suggests that in order to build SA systems that utilise large scale ad hoc sensor networks for situational information acquisition, requires good understanding about SA theories and requirements. Current work utilises situation-focused approach while carrying out the situational information acquisition, exchange and distributed processing of situational information in ad hoc sensor networks. The general requirement of the up-to-date SA means that the relevant information should be constantly updated - i.e. streams of data are created from disparate required sources in order to obtain the required relevant situational information. These streams are transported to the destination via an ad hoc hierarchical system (formed on-line according to the hierarchical build-up (description) of situations) of networked nodes which topology can change. Also the requirements for what is relevant information can change on-line. Same goes for the required update rate of the required information production. This means that the acquisition of situational information should be goal driven or purpose driven by real SA needs. The ad hoc system of networked nodes that is formed on-line after receiving information requests causes unpredictable and time-variant delays. The next section continues with an analysis of military SA systems in order to derive specific SA requirements for the sensor network.

2.4.1 SA requirements for Smart Environments

The specific operational and functional requirements for Smart Environment that is used for SA applications (where one example is military sensor networks and overlying ISR systems) call for a different architectural design and adaptive topology of sensor networks compared to classical (e.g. environmental data collection) examples. This section reviews some of these requirements. At the core of most of these requirements lies the need to have a situational understanding of events taking place in a monitored area. The situations (or rather the events) can be very versatile and numerous and it is not always clear at design and deployment time, what events can occur and need to be detected. With regard to situation definition this section refers to section 2.4.

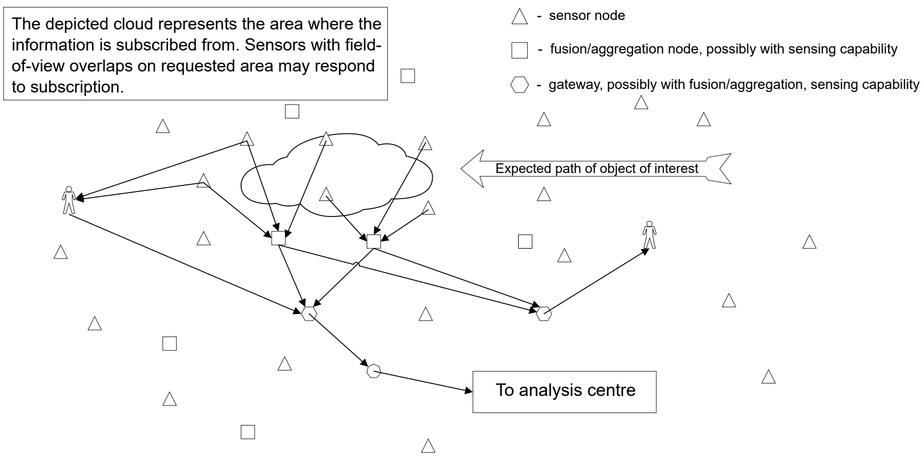


Figure 12 – WSN deployment and dynamic formation of network links at run time according to the required situational information. Military units, such as ground patrols and unmanned aerial vehicles, acquire data directly from network nodes when in vicinity. When requested, data are also forwarded to analysis centre for online and/or offline analysis.

The SA requirements in a military situation for in-the-field units who need situational information relevant to them in a timely manner, preferably directly from network nodes as depicted on Figure 12 and in human understandable form is especially interesting from the context of current research. Information for situational awareness therefore needs to be provided already at sensor network level utilising the capabilities of network nodes. Emphasis is put on requirements that are derived from highly dynamic Intelligence Surveillance and Reconnaissance (ISR) military situations and the corresponding complex data flows within ISR sensor networks. This thesis exploits the design paradigms of Mist Computing and D2D paradigm which are about pushing computation to the edge of IoT network and bringing correct data in timely manner to the right decision makers. One of the main requirements, which usually differentiates military sensor networks from typical civilian special purpose (scientific and commercial) sensor networks, is the need for highly dynamic network structure – the ability to add or remove nodes and reconfigure communication paths on the go. In ISR applications, information collection is context based (i.e. constraints for information are contextual), therefore precise data requirements for tactical operations of military units are not known before sensor network deployment and often change dynamically during operation. A sensor network with fixed structure and functionality cannot cater for the changing SA needs. Sensors and communication paths for specific situational information must to be chosen and formed on an ad hoc basis. Sensors may also leave these networks, as their power resources become depleted or they get destroyed, and new sensors, possibly with different functionality, may join the network according to the changing demands of the overlying ISR system. It is therefore not possible to design a complete sensor network for SA applications, with fixed configuration, structure and predefined data flows. Rather, the system must accommodate dynamic run-time sensor discovery, tasking of data providers, that is, identifying and subscribing to available data sources (sensor and/or other information providers) and ad hoc formation of communication paths to cope with the changing goals and environment. As is described in Publication IV, the data consumers do not subscribe to data from specific data providers with their unique identification numbers, but rather subscribe to data according to its type, location, time and other parameters that describe (and give context to) the needed situational information. Another requirement, from the point of in-the-field military units, is the need to have an ability to acquire data directly from nearby network nodes rather than connecting to a remote database. The advantages of this approach are that connecting to a remote database takes time, communication can be intermittent and the database may not have the latest data. Also all sensor network nodes would have to constantly and regularly update the database, which consumes network bandwidth and takes time. As an alternative, distributed data aggregation and fusion must be performed at network node level to provide the necessary situational awareness to military units. Sensor and other nodes must be capable of carrying out the necessary signal processing, data fusion and aggregation calculations, while at the same time assuring that the data used are valid and mutually conforming. Performing in-network data fusion and aggregation in timely manner requires consistency of data collected from different sensor nodes in both temporal and spatial domains. In order to ensure the validity and usability of fusion and aggregation results, contextual constraints must be applied to collected data. Spatial constraints, such as bounds to the area of interest, and temporal constraints, such as acceptable age interval of data, are defined within the subscription made to the WSN by the user. In either case, data providers, the different sensor nodes, must augment all collected data with appropriate temporal and spatial metadata tags, which are later used in the data validation process. In addition, network communication layer must support in-time packet delivery (within the pre-specified delivery interval) or inform data provider and consumer of failure to (temporarily) meet these requirements during operation. The wireless sensor network examples presented in this thesis utilise a messaging syntax and communication protocol for the sensor network that facilitates satisfying the above described data validity needs [95], [107]. The expected environmental conditions for tactical military sensor networks are for the most part similar to those in typical civilian environment monitoring sensor networks – sensor nodes are situated in harsh environments and need protection against the natural elements. Operational conditions, however, are different due to constantly changing military situation and the existence of malicious adversaries trying to disrupt network operation. Among other properties, it is desirable that sensor nodes be physically and electronically inconspicuous and if possible resistant to tampering, denial of service [137] and deception type of attacks. The latter properties that concern security and electronic warfare are outside the context of this thesis. Based on the above, one can identify six major requirements for tactical operation purpose distributed ad hoc sensor networks:

1. It should be possible to identify and task data providers at run-time using context-based data constraints.
2. It should be possible to ensure the contextual validity of the situational information

on-line.

3. It should be possible to assure online that data from distributed sources are mutually conforming.
4. Situational information should already be created on sensor network level.
5. Dynamic network structure and functionality is preferred along with ad hoc formation of communication paths.
6. Network nodes should be capable of in-sensor signal processing, distributed data aggregation and fusion.

The list is not exclusive but serves as a starting point for developing distributed sensor network systems for SA applications.

2.4.2 Situational information acquisition driven by real Situation Awareness needs

From the definition of situations it is possible to derive that some situations are more influential and thus relevant to the agent than other situations. This and the continuous need of up-to-date SA requires that the acquisition of situational information must be goal driven. Or driven by the needs of the cognitive agent. In distributed systems of agents this involves requesting data from other agents in a goal driven manner.

Information collection task can be carried out both by data centric approach (bottom up) and by goal driven approach (top down). The approach where all data is mechanically processed from raw data to higher level information or situation descriptions is called data driven. Data driven approach can be compared to a notion of passive SA as described in Publication VII, as information is just collected from currently available sources and stored to a database. Other two types of SA - the reactive and proactive types of SA, described in Publication VII require already a goal driven approach for data collection as both types require active control of information collection. The reactive type of SA reacts to an ongoing crisis and needs a goal driven approach on data collection for finding ways to mitigate ongoing situation. The proactive type actively defines and redefines the information requirements (goals) to detect signs of future crisis to avoid them.

The problem with the data driven approach (passive type of SA) is that due to physical limits to information collection not all data can be continuously collected and stored at cloud servers. So when the data is required for some specific SA needs, some information can be missing or are left uninitialised. It can be said that data to valorise a situational parameters is not available. In this case, as one possibility, default values could be used for them. Another way is to used top down approach, or a goal driven approach. Goal driven approach uses situation description at higher abstraction level to find what data is missing in order to understand the current situation correctly or to simulate current situation into future. When missing situational information has been identified, then, instead of leaving them uninitialised or assigning them default values, the agent deliberately targets its attention to acquire the missing data. In a model, where agents acquire data using a subscription-based model, the agent updates data subscriptions. It is the task of the SA system to propagate the subscription to sensor nodes that have the needed data. One could also think out of the box here and consider a System of Systems approach to include either mobile sensors (e.g. a UAV) or even task a capable agent to deploy the required sensors in ad hoc manner.

2.4.3 Examples of systems where considering Situation Awareness in artificial agents is required

Some example fields of applications for SA theories and concepts comprise air traffic control, shipping, medical systems, rescue coordination centres, military ISR systems, etc. Another sample use case could be a smart city lighting system that adjusts lighting level according to traffic density (which is assessed using autonomous sensors on city streets) or a smart office that turns on its lights when person nears or enters the office room. It can be said that the office room detected a predefined situation that required an action. Most of these applications could well make use of INDP in distributed sensor networks.

3 Concepts and building blocks for situational information in Smart Environments

This chapter describes necessary building blocks required for building up SA by utilising Smart Environments. The concept of Smart Environment is tightly related to the field of Systems of Systems, which consists of cyber-Physical Systems. Hence, the next two consecutive sub-chapters are dedicated for explaining both concepts. In next sub-chapter, the concept of mediated interactions is explained, as it is used for the management of interactions and information flows within the networks of sensing nodes. The mediation process is controlled by publish subscribe method, which is explained in next sub-chapter. The cognitive agents subscribe to information services provided by Smart Environments. The information collection and exchange for SA between distributed nodes in turn is based on situation parameters. This chapter also provides definitions for the concepts of situation parameters and their validity and relevance and how situation parameters are used for ISDP and INDP algorithms. Finally both ISDP and INDP are explained.

3.1 Systems of Systems

Smart Environments can be composed of various sensor networks and IoT devices that originally may not have been designed to work together, but which still either influence each other or are in interaction during their life cycle. In this they they qualify well to a very general definition for a SoS is provided by Brook "*A SoS is a system which results from the coupling of a number of constituent systems at some point in their life cycles*" [45]. Such systems are called Systems of Systems as the components, in turn are systems, not just components. The relation between the IoT and SoS have been very elegantly described by Michael Henshaw in [28] and is depicted on Figure 13. The most interesting portion of the figure are the circle surrounding the IoT and the segment on the right side of the IoT circle, but still belonging to the SoS circle. This segment describes CPSs which are connected and that could be interacting using non-internet technologies (e.g. Zigbee, Thread, Lora etc.). However, in modern world, the systems that are formed of these CPSs are also connected to internet via a gateway (a Fog level node). In such Systems of Systems the overall nature and behaviour manifests during the operation and such behaviour is not reducible to their constituent components specifications.

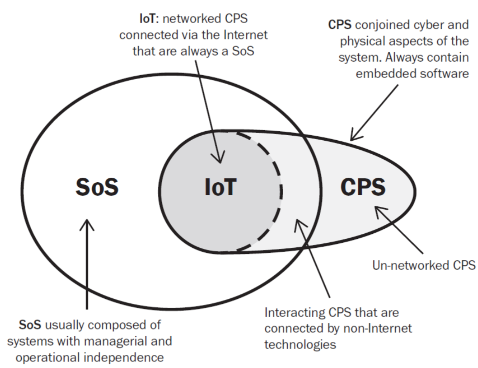


Figure 13 – Relations between SoS, IoT and CPS. (adopted from [28]).

One of the differences between a system and a SoS is that the system generally has a specific functionality or a specific purpose, whereas a Systems of Systems can exist without a top level goal functionality (but may have it) [17]. An SoS-based approach is different from a systems-based approach, as a SoS [17]:

- treats integral parts as unpredictable or partially autonomous
- expands the view of an individual system to the environment by accepting other unpredictable or partially autonomous systems as parts of the environment (environment is richer)
- components create associations with a SoS voluntarily and motivated by the component's own interests to belong to the whole
- components have relationships which are dynamically emerging and changing. The possibly large number of connections and different ways to connect (connection types) make their effects more complex to analyse (e.g. indirect effects).
- exhibits emergent behaviour (both beneficial and harmful). The detection of emergent behaviour is possible only dynamically during operation.

A SoS can be well defined and have a fixed structure at some point, but over time it can be constantly changing. The changes are mostly slow. The SoS:

- is with open-ended goals
- does not need to be designed for only for one purpose
- components can each have its own goals, collaboration helps to achieve them
- systems and their components are constantly changing
- can be robust or resilient, replacement or loss of one system will not normally terminate the operation of a SoS
- change in composition and properties is a problem in safety-critical domain (verification of SoS can be very difficult or not possible at all, e.g. in the fields of aviation, car control systems).

However, SoS approach is needed as it allows to describe and analyse today's various IoT and sensor networks as a single whole, as a Smart Environment.

3.2 Cyber-physical systems

Contemporary computational devices embedded in physical world equipped with sensors and actuators are called Cyber-Physical Systems (CPS). They function in and with the physical world and should thus be viewed as unified into systems together with their environment. The computational devices are deeply embedded in the physical environments and the activities of the cyber part are first and foremost driven by ongoing situations in the physical environment. According to Lee [78] a CPS is an orchestration of computers and physical systems. Some CPS which are standalone systems, operate alone while controlling some specific physical process, others can be connected to each other, connected either via the internet or by non-internet technologies. The connected ones form distributed systems which belong to the class of Systems of Systems [28]. Hence, the SoSs consist of components located in both virtual and physical environments, which have connections to and interactions both with each other and with the surrounding natural environment. The systems which are in interaction with their surrounding environment are also called open systems. The interactions can involve both exchange of energy, matter or information. The open systems are more likely to exhibit emergent behaviour than closed systems.

A simple example of a CPS would be a sensor device which periodically collects data about a specific physical phenomena and takes action if it detects a change. This action can be a reaction to the change or even a proactive action in anticipation of some future changes. An action itself could involve several activities, such as turning on additional sensors in order to obtain a more detailed observation about the change, more resource demanding data processing, data requesting from other networked devices or from general internet to provide more info etc.

The concept of CPS differs from the concept of embedded systems by its forced adaptation to the physical environment. The devices of CPS must often be capable to change their way of behaviour due to changing situation in their physical environment - they need to reorganise, change the systems topology, redesign the interactions according to what the surrounding physical environment dictates.

CPSs are often used to control processes in the environment where they are located. Instead of viewing an embedded device as a single solitary control system, the contemporary CPSs are becoming increasingly connected by different networks. The problem that emerges is that a whole behaviour of CPS-s and emerging systems of CPSs cannot be defined by the specifications of individual embedded components. According to SoS theory, one of its main features is that its behaviour and characteristics cannot be described through description of the behaviours or properties of its components [17]. This is especially the case when one considers IoT and large scale sensor networks as Smart Environments. In order to incorporate the sensor network (Smart Environment) as a network of Cyber-Physical Systems into System of Systems view, one must consider both the monitored phenomena, the methods used to monitor, influence or control the phenomena and also the overlaying information management system to help the user to take the necessary decisions. In the context of seemingly non-deterministic interactions between systems the validity of the data that is exchanged is crucial for ensuring correctness of the outputs of algorithms using the data as an input.

For example in Publication III, Kaugerand et al. describe a SoS consisting of a network of ground sensors and an unmanned aerial system (UAS) with a task to detect trespassers crossing a certain perimeter. Each component is also a CPS. An aspect in designing the SoS considered a collaboration between UAS and UGS network on the ground. Both the UAS and UGS network are a highly dynamic systems and a fixed and synchronous communication plan wouldn't have worked. The solution was to create a SoS, which is assembled from autonomous systems, eliminating the need for detailed planning before deployment and operation and also for central coordination. Our team showed that the UAS that is operating in an open and dynamic environment, in order to accomplish a collaborative tasks, should not just be automatically executing a pre-programmed sequence of steps. Instead, the UAS required dynamic interaction with the UGS network in order to be able to autonomously decide between choices presented by the perceived situations and adapt to changing environment in order to achieve efficient results. Authors in Publication III showed that one of the possibilities to implement such a SoS is to apply an approach of mediated interactions.

3.3 Emergent behaviour in Smart Environments

Emergent behaviour is very important in Smart Environments. Especially so, as a Smart Environment can potentially be created or rather often it emerges during gradual addition of different IoT or other embedded devices to the environment, where they are able to interact with each other, in one way or another. Often such devices are not originally designed to cooperate or even if they are (as can be the case with mesh WSNs with INDP capability) the use cases and applications can develop later during the operation. Either via software updates or via human ingenuity for finding new applications for existing IoT solutions. Understanding emergent behaviour requires some insight into systems theory. Systems theory describes two levels, macro level of a system and micro level where the components reside [29]. The novel behaviour which appears on a system level and cannot be deduced from individual system components is called emergent behaviour. In their work [29], De Wolf and Holvoet provide a thorough explanation to the emergent behaviour and give its comparison to the notion of self-organisation. While self-organisation exhibits similarly to emergent behaviour 1) increase in order, 2) autonomy, 3) adaptability, 4) robustness and 5) dynamical appearance, the phenomena of emergent behaviour in addition requires and displays properties such as 5) micro-macro effect, 6) radical novelty, 7) a persistent pattern, 8) interaction between parts, 9) decentralised control and 10) two-way link between micro and macro level. A good example of self-organisation is an ad hoc network which builds up its structure while nodes detect each-others presence. The core of De Wolf's and Holvoet's work is depicted on Figure 14. On this figure the two levels of systems behaviour are presented: micro level-part of the emergent system is usually very complicated and disordered, but through negative and positive feedback, the increase in order is achieved on the macro-level where emergent behaviour and very possibly a resulting self-organisation happens. It is one of Wolf and Holvoet's main suggestions that emergent behaviour often appears together with self-organisation - it is both possible and also very useful to have self-organisation supported by emergent behaviour. In the Figure 14, the appearance of self-organisation is depicted with curved line. It means that despite the emergent behaviour at the macro level, the system can still exhibit purposeful order organised by system itself. What is important here, is that the authors bring out that the changes that occur on macro-level of can be either amplified or suppressed by the negative or positive feedback loops between the micro and macro levels.

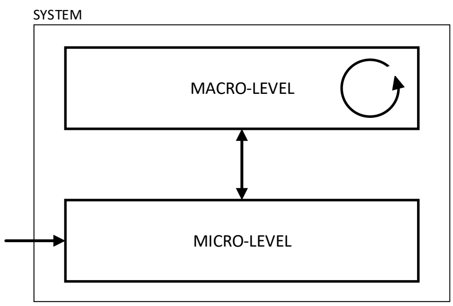


Figure 14 – Emergent behaviour combined with self-organisation [29].

Similar idea is also provided by Parunak et al in [101], where authors explain that emergent behaviour appears in non-linear systems. In linear systems, the whole is always equal to the sum of its components. The authors also mention that some known non-linearities that cause emergent behaviour are capacity limits, feedback loops and temporal delays [101]. Authors continue and claim that although emergent behaviour often appears as s superficially random and in that case, the only solution to mitigate it lies in structural modification or parameter tuning, not tighter control over varying environmental conditions. The reason for this is that according to authors of [101] the emergent behaviour, however seemingly random, results from completely deterministic processes. This thesis uses these ideas to foster the positive emergent behaviour and suppress the negative emergent behaviour by using the concept of mediated interactions which are controlled by subscriptions and respective contextual constraints by subscribing agents (data consumers), explained in 3.4.2. However, it remains as One of the future research tasks to analyse if the constraints could be determined automatically according to the network status. It is possible that would lead to self-organisation together with emergent behaviour.

Historically philosophers have described this phenomenon of emergence happening in nature or in the field of physics. In the following, a brief historical overview over scientific research about emergent behaviour is described. Earliest hints about contemporary understanding of emergent behaviour originate already from 19th century. In 1868 T. H. Huxley asserted that the peculiar properties of water, its 'aquosity,' could not be deduced from our understanding of the properties of' hydrogen and oxygen [90]. Not being able to deduce macro level behaviour from the properties of systems components is one of the characteristics of emergent behaviour. Another view to the same characteristic is given by Bahm Archie in 1981 in his "Five Types of Systems Philosophy" [9]. Bahm explains that emergentism is something where parts exist prior to wholes, suggesting that the whole, together with its novelties, emerges from the connections, relations and organised interactions of parts. One of the first attempts to describe emergent phenomenon in the field of engineering was made by Philip Anderson in 1972. He explained in [5] that the behaviour of a complex system cannot be understood in terms of a simple extrapolation of the properties of its components, elements, and entities. He gave an example to explain limitations of reductionistic view. He explained that if you take any complex object from nature and decompose it into boxes where every rule can be explained (and if not, then decompose until you can) and then take these atomic boxes and try to rebuild the original system, you may find it impossible. The decomposability of an emergent system has also been confirmed by Susan Stepney in [105], where she together with Fiona Polack proved that the emergent behaviour cannot be predicted analytically using contemporary formal methods during system design time. Today there are different approaches (schools) to emergent behaviour. A good overview can be found in [62] where Johnson describes different views of how emergent properties are described by different authors. Some include any unexpected properties, others refer to properties that cannot be identified through functional decomposition (An example of latter is given by Fiona Polack and Susan Stephney in [105]). Yet another view is a distinction between weak emergence (that can be discovered by simulation) and strong emergence, which arises from downwards causation, meaning that emerging macro level in turn affects the behaviour of components at micro level [62]. Johnson also explains that most often the emergent properties are used to distinguish complex systems from applications that are merely complicated. However, some points of view can also be found that do not support the existence of emergent behaviour in engineered systems, this is a popular view among supporters of contemporary formal methods, based on decomposability and refinement. Common to these views is that they say that the complexity in macro level behaviour of systems (and emergent behaviour) appears only when complete information about the components atomic behaviours and their interactions is not available during the design time. For example Edmonds, B., in his PhD thesis (1999) [35] raises an open question whether emergence can actually exist in a system made up entirely of engineered components. It is a question about emergence versus ignorance. In [43] (2013) Felder reformulates this question: "Is an unpredictable result truly a feature of the complex system, or merely an artefact of our lack of understanding?". In [23] Baldwin together with Felder seek to develop a mathematical test of this hypothesis, but the effort is still in its infancy. They also raise the question whether the mathematical methods, tools and engineering methods are good enough for designing complex systems. This view is justified by the fact that despite of the contemporary rigorous formal engineering methods, especially in safety critical systems, there are still too many accidents due to unexpected behaviours or interactions between subsystems.

The reasons why this thesis is interested in the concept of emergent behaviour is firstly because, as elaborated above it exists in systems relevant for this thesis and secondly because of its useful properties. It is known from rich scientific literature that emergent behaviour can potentially have some very useful properties, although usually only observed in nature. Still it would be very beneficial to have the same properties in behaviour of artificial SoS. To name some of these properties: decentralised control, robustness, flexibility, dynamical appearance, persistence in time [29]. The concept of SoS is a related field to computer and systems engineering where the concept of emergent behaviour is especially well defined [98, 28, 51]. In SoS the emergent behaviour manifests itself mainly due to the distributed nature of its components (loose coupling), because components have high level of autonomy (i.e. instead of central control) and due to interactions between the components. For example, many solutions, which can be described as SoS, are designed with components which are exercising various degrees of autonomy with decentralised control. It is theoretically possible that the emergent behaviour appears dynamically from interactions between the components, remains persistent in time, does not have central weak points, is robust and flexible with regard to changes in its environment.

3.3.1 Managing emergent behaviour in distributed Systems of Systems

Scientific literature contains very few examples concerning detection and control of emergent behaviour in distributed systems. The few that do, are based mostly on central approach. In [122] a model based testing and real time simulation approach is taken. A central RFPPE (Run-Time Fault Prediction and Preventing Engine) gets inputs from constituent systems and applies adaptive and robust distributed system model that is updated in real time to run real-time simulation and configure the distributed system according to the predicted faults. In [135] a probabilistic online reliability time series prediction model is introduced. An overview paper [22] investigates current research on run time assurance methods for CPSs and briefly discusses problems related to distributed systems. Clark et al suggestes that a closest related contemporary approach to guarantee the online deterministic behaviour of a distributed system is runtime verification (RV). Especially time-aware RV. The RV has evolved from traditional verification techniques such as theorem proving model checking and testing with the focus on functional correctness [24] of monolithic systems [48]. An example is described in [12] for potentially steering the application back to safety region if a property is violated. But RV is not yet suitable for SoSs for many reasons, some of which are that as these methods suffer from state explosion [81], require manual effort [81] or are infeasible due to economic reasons. In RV, the execution of a program is checked against the specification described in a formal language (usually standard temporal logics), which requires synchronous time triggered architectures. This in turn means that systems in a network are periodically synchronised and everybody must agree on a "wall clock" time. RV for geographically distributed SoSs where synchronisation is often not possible or feasible, becomes a challenge. The problem is that the RV monitors installed on distributed systems will form another distributed system, which

also may fail. To avoid this, the few existing attempts implement a central solution i.e. [14] where distributed nodes report to a central diagnosis engine that attempts to steer the distributed system to a safe state.
The conventional online verification methods, based on the theory of Turing model of computation, fail to handle systems that integrate autonomous components, and may have dynamically evolving structure [94]. This thesis avoids the central control architectures and builds upon previous works, related to distributed validation, like [95], where self-aware architecture to support partial control of emergent behaviour is described in 2012 by Leo Mõtus and [107] where on-line data validation is described in 2013 by Jürgo Preden. The hypothesis is that by local ability to analyse the time and location properties in interactions of a SoS, it may also become possible to detect and mitigate (manage) the emergent behaviour. Investigation of methods for influencing system-wide behaviour dynamically (i.e. run-time) by adjusting local temporal and spatial constraints for validity imposed on constituent systems and on their interactions might enable us to detect anomalous (e.g. emergent) behaviour in due time and mitigate unwanted behaviours, foster favourable impact of goal directed behaviours, and keep the SoS in safety and security envelope.

Examples of emergent behaviour are for example cases where some communication delays or processes take longer time than expected during the design and for this reason the resulting situational information becomes insufficient for SA applications. Another example for emergent behaviour is how sensors form hierarchies for data collection and for INDP. These patterns are not known during the design time, they form on-line according to the user requests for information. Yet another example case of emergent behaviour can be described with a large distributed ad hoc sensor network. If one chooses a simple task of vehicle monitoring in an urban area, then for example a number of microphone-array sensors could be used for this task. These microphone-array sensors (explained further in Section 3.7.2) are capable of computing a direction to the sound source and a class of the vehicle (e.g. size and type of engine). Each sensor node is deployed so that their field of views overlap. By combining sensor readings for example from N autonomous microphone-array sensors a data fusion of the readings can be carried out in order to compute a location of a passing vehicle. This in itself is not a case of emergent behaviour, as the position computation function is a deterministic function and there is nothing novel in the result. However, the way that specific sensors in ad hoc manner are chosen for this task according to contextual (e.g. spatial and temporal) constraints can be. It is also possible that during the task, the system adds additional or removes dynamically sensors. Other sensors with different modalities which have their field of view overlapping or close to the field of view of the microphone array sensors. For example radars, movement detectors, video based sensors, etc. The resulting final picture represented to a cognitive user can lead to synergy between different sensor data. The resulting dynamical, robust, persistent SA can definitely be interpreted as emergent behaviour.

3.4 Mediated interactions

This section gives a brief overview of definitions of different type of interactions that can occur within a CPS in a Smart Environment which essentially can be interpreted as SoS. The term interaction is well known both in the fields of systems engineering, multi-agent systems and theory of interactive computation. While in systems engineering the interactions and behaviours of systems is well defined and understood in the field of interactive computation the behaviour of a system depends both on input and systems internal memory. The interaction is how systems or cognitive agents exchange energy and information or even influence each others behaviours. By definition the interaction is an ongoing two-way or multi-way exchange of data among computational entities, such that the output of one entity, interpreted by another one, may causally influence the later outputs of the second entity [66]. When returning to the topics of CPS and SoS, it can be reasoned that while CPS often exhibit a high level of autonomy, they may still have a monolithic architecture. By introducing the concept of SoS, the individual CPSs could be composed into large scale networked systems. These networked CPSs can organise into different collaborative systems of CPSs according to their individual goals. Each of these networks can be viewed as a SoS, but also together they may form a large scale SoS. Modern CPS are often expected to be able to operate in dynamic, open and unpredictable environments and more importantly also in the context of a changing SoS configuration. In an SoS, the systems making up the SoS interact autonomously with each other to collaborate in order to achieve their own goals or higher level goals. The interactions form a crucial part of a SoS configuration as the higher level functionality can be only achieved via a interactions between the systems. Unlike a system with a fixed structure, where the functionality of the components and their interaction patterns are well controlled and predictable, in a dynamic SoS the interactions are not fixed as the system configuration itself is not fixed. Having interactions mediated by a smart agent or a demon, it should be possible to influence systems behaviour. However, this thesis is investigating if interactions and information flows can be managed by using smart proactive mediator agents. The hypothesis is that mediator agent, which is capable of influencing the interactions already close to the phenomena of interest is needed for obtaining SA. This question was analysed in Publication IV by Preden and Kaugerand, where authors argue that in CPS-s, especially in distributed systems of CPS-s it that are applied for SA, is useful to have both control logic and computation close to the physical phenomena of interest.

Before coming to the definition of mediated interactions, other types of interactions need to be explained. There are different types of interactions, direct, indirect and mediated. Direct interaction is usually interaction via messages, where destinations of messages are specified in the message [66]. This is depicted in figure 15. In everyday life, direct interaction is something that influences another object directly, in this way agents can influence or be influenced by their surrounding environment directly. For example if an agent sends messages to another agent or is directly controlling a physical process one can say that this agent is in direct interaction with either another agent or with the physical process.

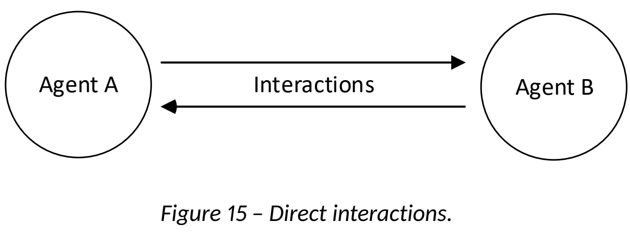


Figure 15 – Direct interactions.

Another type of interactions is indirect interaction. By definition this type of interaction is an interaction via persistent, observable state changes and destinations or interaction consumers are any agents that will observe these changes [66]. This type of interaction usually assumes an existence of operational environment which can be influenced by residing agents in a way so that this influence causes a change that can later be observed or detected either by the same agent or other agents residing in the same environment. This concept is depicted on figure 16. The object on the figure can be anything which state can be influence and later observed by any other agent. The indirect interactions are ubiquitous, examples include stigmergy of social insects that interact by modifying common structures, in anatomy cells exchange information via hormones, an economical market is an environment for buyers and sellers, in computer science a classical solution to producer-consumer (a multiprocess synchronization) problem is a use of semaphores [66]. Another very common example of indirect interactions in distributed wireless sensor networks is a broadcast message in a wireless network. The most significant properties of indirect interactions, listed by Keil are asynchrony, anonymity, geographical distribution, non-intentionality, hybrid nature and late binding of receipient.

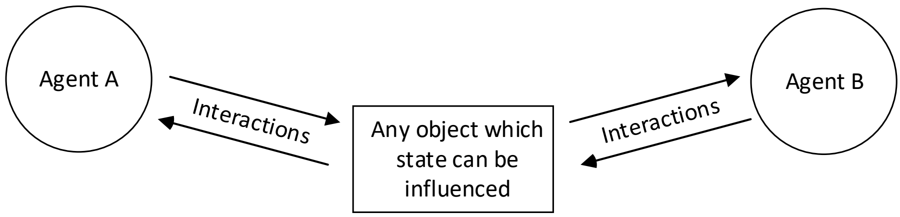


Figure 16 – Indirect interactions.

The third type of interactions is mediated interactions. This type of interactions is similar to indirect interactions but allows to introduce a concept of a mediator who can be modelled as an intelligent agent [94]. This intelligent mediator has a capability to manage the interactions and to manipulate their content. The management of interactions is done by middleware services and by manipulation of interactions in the context of this thesis means the manipulation with contextual validity and consistency. Both methods are explained further in Sections 3 and 4. Mõtus et al suggests in [94] that a middleware (mediator software) needed for collection and exchange of situational information should be designed based on the concept of a mediated interactions. The concept of an intelligent mediator is depicted on figure 17.

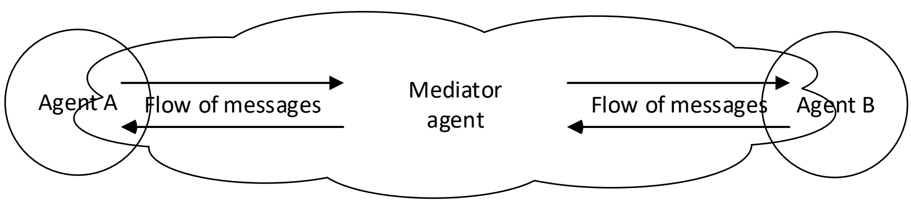
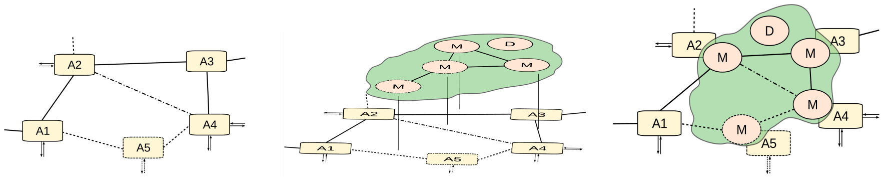


Figure 17 – Mediated interactions.

The abstract notion of a mediator agent is depicted with a cloud which covers parts of both interaction partners (in theory the number of interaction partners is not limited). Practically the mediator is implemented as a middleware software which is implemented in all participating agents. On the figure the information flows are depicted as discontinued arrows, this demonstrates the capability of the mediator to manipulate the contents of the interactions (e.g. to disregard the data items which do not satisfy the mediation rules). The middleware design based on concept of an intelligent mediator has been earlier elaborated by our research group in several works. For example in [107], where Preden et al describe thoroughly on-line validity checking in data streams in distributed sensor networks.

In [95] Mõtus et al suggests that a system of distributed autonomous agents could be capable of achieving self-awareness about its architecture by substituting some conventional interactions by mediated interactions. The mediated interactions allow to observe, analyse, verify and even partially control systems emergent temporal and spatial behaviour from the interactions. The concept is explained on Figure 18.



a) b) c)

Figure 18 – A theoretical model of a SoS with explicit interactions between components. Solid lines represent current interactions and dotted lines future (possible) interactions. Sub-figure a depicts a conventional SoS, sub-figure b depicts a SoS with mediated interactions and a concept of a demon for self-awareness and sub-figure c depicts an implementational view of mediated interactions. Adopted from [92].

The Figure 18 a depicts a fragment from a theoretical model of a CPS with explicit interactions between components, where systems topology may change in time. Solid lines represent current interactions, dotted lines future (possible) interactions. Dot and dash line represents a current interaction that may disappear in future. Arrows represent interactions with the environment. The Figure 18 b depicts a concept of a demon for self-awareness. Interactions and mediated interactions form an instrument that is used to generate self-awareness coordinated by demon D. The Figure 18 c depicts an implementational view of mediated interactions. Agents are equipped with network interfaces (soft- and/or hardware) that also cater for mediated interactions.

This thesis continues this line of work and demonstrates the applicability of this idea through several experiments described in Publications I, II, III and V. As theoretical contributions, in Publication III, the author demonstrates that the concept of mediated interaction enables viewing a set of collaborating heterogeneous components as a System of Systems, in Publication II the authors demonstrate the considerable bandwidth usage reduction through mediated interaction and in Publication I the author demonstrates that in addition to the on-line validity checking, the intelligent mediator (implemented as a middleware) must be able to assure that interactions between distributed sources should only exchange temporally consistent data.

The application of concept of mediated interactions creates a possibility for managing the systems to adapt to new conditions either by manipulating with contextual constraints or with respective validity intervals. The on-line validation of interactions via time selective communication should help to filter out the negative behaviour (i.e. interactions which do not satisfy current requirements). This thesis expects to work towards answers to these questions both by theoretical reasoning and empirical experimentation, a possible outcome could be one step closer to a SoS that can achieve its goals while adapting to the changing environment, user SA requirements and coping with the dynamically joining and leaving components and while at the same time continuously providing required services and successfully avoiding overloading the network with data collection tasks.

Another challenge is detecting emergent behaviour in this context. Theoretically this would be possible if the system specification is available. It could reveal if system behaves differently from specification. However, this thesis considers large scale sensor networks for Smart Environments where specification regarding communication paths and the contents of messages being communicated is not available at design time. This is because on one hand the communication paths are formed dynamically based on in-progress link qualities depending on the RF environment. On the other hand, both the composition of the system is evolving and changing (new nodes can be added and old removed) and it can be difficult to predict the evolving needs by SA users. In this case, instead of specification, the intelligent mediator agent, described in this section, applies specific rules or rather contextual constraints on interactions that are mediated. If values contained in certain messages do not satisfy the mediator rules, then these messages are not passed on. This effectively eliminates reactions to the interactions which are invalidated.

Practical examples in this thesis utilise the concept of mediated interactions. The mediating agent passes on (or mediates) only interactions that guide the system towards its goal, other interactions are denied. This is an attempt to deny the feedback that guides the macro level result away from the behaviour which is not desired. In section 3.4.2 the thesis proposes that the subscription based system for situational information collection enables autonomous creation of a logical hierarchy of nodes. The resulting information collection system is emergent. However, the system is influenced by certain rules that allow to filter out interactions, that do not satisfy the rules. The expectation is an emerging SA for a cognitive human or artificial user. This type of mitigating or fostering the emergent behaviour helps to guide the system towards its goal. The mitigation of negative and creating conditions for positive emergent behaviour can be useful because the emergent behaviour is present in such SoS and of its potential positive properties as has been explained in previous Section 3.3.

3.4.1 Q-model - a formalism for modelling mediated interactions

This section gives a general overview of the Q-model formalism [96] as used to describe mediated interactions. The Q-model handles time-sensitive and time-selective computation and is a prototype of a multi-stream interaction-centred model [93]. It is the only known formalism that is used for modelling mediated interactions. Q-model is based on concepts from abstract communicating processes, which in turn is based on a process algebra first published by Hoare in 1978 [8] and work done by Quirk and Gilbert [114]. Q-model applies weak second order predicate calculus for expressing and analysing interactions and timing behaviour in real-time software already very early during the design [96]. A statement from Goldin and Keil in [46] partly explains the essence why higher order logic is needed for modelling interactive computation – no sequence of preordained steps (or series of interactions) can model the multiple-stream encounters between multi-interacting evolved agents and their environments. The WSN which forms a Smart Environment, is at an abstract level a SoS which is composed of loosely coupled autonomous agents (both CPSs and possible human users) which communicate by subscribing to and producing situational parameters. This abstract description of such SoS at this level is very similar to how processes in the Q-model can be described as autonomous agents and interactions between the agents can be modelled by asynchronous, semi-synchronous or synchronous channels. The communication between such distributed embedded autonomous systems is not fully analysable in (first order) temporal logic or for example in timed Petri-nets. If one would compare Q-model against Petri networks, regarding modelling data exchange in wireless sensor networks, then several aspects speak in favour of Q-model. Some of the more significant aspects that should be highlighted are: Petri networks assume that all causal relations are known, Q-model takes another approach and allows to approximate unknown causal relations with time constraints. Another difference is that Petri networks assume unbounded execution time for all processes, Q-model instead sets temporal constraints for execution intervals of all processes. Of course there are many extensions to the Petri networks, such as Timed Petri networks and Stochastic Petri Networks, however, they do not allow analysis of time-selective inter-process communication provided inherently by Q-model. Another similar model for modelling real time distributed embedded systems is provided by Caspi and Halbwachs [20]. This model uses metric time to describe temporal behaviour of real time distributed systems and it also uses a concept of an asynchronous distributed arbiter to manage the systems access to communication bus. The synchronous, declarative programming language for reactive systems - Lustre [50] and industrial environment SCADE [16] are outcomes of this the work of Caspi and Halbwachs. However for modelling the mediated interactions in Smart Environments the Q-model has been found more suitable due to its abilities to describe the timing correctness of interacting asynchronous processes and other aspects described in this section. Q-model has the description and analysis power of a multi-stream interaction machine and supports automatic prototype generation at the very early stages of systems' development. The Q-model is very suitable for modelling timing issues of distributed processes and analysing resource conflicts since those characteristics are an essential part of the Q-model ideology. Another model that could be used to model temporal issues in interactions between distributed embedded systems is UML MARTE Profile adopted by Object Management Group (OMG) [6]. It incorporates many issues into the concept of modelling time and performance used in the Q-model. However, in the context of modelling mediated interactions, the Q-model is considered here more appropriate as it allows a selective handling of inputs for processes, which is an essential property to model mediated interactions.

Q-model is composed of a processes and channels. In its representational form Q-model is a graph model, where nodes represent processes and where each arc of the graph is a channel with its own set of attributes. However, contrary to classical graph models which usually are time invariant, the Q-model is different. Q-model utilises a time model [93], based on the simultaneous use of multiple time concepts - fully reversible, strictly increasing and relative. Q-model can be classified under a class of modelling methodologies that can be used to research special type of problems. Q-model is designed for modelling and specifying temporal aspects in complex real-time systems of distributed embedded computers. Q-model processes are a set of loosely coupled repeatedly activated terminating processes which transform input data to output data (mapping of data from definition domain to value range). Processes exchange state values via intelligent communication channels that intelligently guide how and which input data is made available to processes. In Q-model one process may provide data to several consumers, data is copied to several channels, one for each consumer process. In the same way one consumer process may receive data from several data producer processes.

Each interacting embedded system can execute several different processes p. The communication between the processes in different nodes across the network is modelled by a channel σ_{ij}, where i denotes a producer process and j respectively a data consumer process that are communicating. The channel is a logical tool that maps the output from one process to input of another process according to their timesets with the Equation 1.

$$\sigma_{ij} : T(p_i) \times T(p_j) \times val\, p_i \rightarrow proj_{val\, p_i} dom\, p_j \tag{1}$$

The Equation 1 represents the mapping, which conveys the producer process values to consumer process domain of definition or in the context of current work, conveys sensor process values to a INDP process domain of definition. Here, the sensor processes are considered to be running respectively in disparate network nodes for sensing and INDP (in some scenarios it is also possible that a single network node is running both processes). The variables $T(p_i)$ and $T(p_j)$ represent the execution timesets of these processes. The mappings between processes are activated repeatedly, either periodically or sporadically.

Because of the properties described above, the Q-model formalism is very suitable for applying as an on-line timing analysis tool for described SoS. What makes Q-model especially suitable for modelling mediated interactions is its ability to take into consideration only those elements in streams (sequences/histories of produced states or data) that are allowed/permissible. The description of Q-model formalism for modelling the mediated interactions continues in section 4.

3.4.2 Subscription-based control of mediated interactions

In dynamic, quickly changing environments, where SA applications are typically used, only relevant data must be exchanged in a timely fashion and guided by real needs. Central data collection (to a remote database) and distribution comprises a lot of redundancy and is often not flexible enough to provide the necessary timely situation awareness. An alternative approach is one, where service agreements between data users and providers are established at run time, based on actual needs. In this case, communication links are formed locally in an ad hoc manner, increasing system robustness and efficiency. While it is possible to deploy several gateway nodes for data collection, the design of the sensor network for SA applications in Smart Environments should be more oriented towards establishing complex data flows inside the network. This is possible via publish-subscribe paradigm used at the edge in order to manage the situational information collection. The nodes publish their data according to the received subscriptions, which contain rules for data production periods, aggregation and various constraints, such as spatial, temporal, confidence etc. For example, rules for aggregation can describe how data is to be abstracted into higher level situational information, the constraints in turn define the situational information into specific user context.

Considering the described requirements in section 2.4.1 to sensor networks used in SA applications, this thesis builds upon a data exchange model described in [111] for managing the interactions and information flows in the sensor network. The data exchange is carried out by a service oriented proactive middleware - ProWare, developed at the Research Laboratory for Proactive Technologies [94]. Middleware is an abstraction layer that acts as an intermediary and manages interactions both with applications residing within the embedded devices and also with applications across networked devices.

ProWare is a set of IoT software packages that provides communication functionality between sensors, actuators and users over IP or radio link, and makes their functions available on request. The goal of ProWare middleware is to offer an easily deployed solution for discovering and handling services provided by network nodes at the edge (e.g. collecting data from sensors) for creating Smart Environments (e.g. creating a smart home security system or a city-wide traffic monitoring system). ProWare provides the necessary communication services for the INDP. These include handling data requests (in the form of subscriptions), run-time data provider discovery, establishing service agreements with suitable data providers and facilitating the delivery of produced data to consumers.

Furthermore, advanced patented mesh networking technology for MAC layer called Beat-Stack, explained in Section 2.3, together with Dynamic MANET On-demand (DYMO) routing protocol, makes it possible for any individual node to communicate with any other node in the network (provided availability of enough radio link quality and coverage). A means of tracking the time that a packet spends in transit from source to destination is provided, allowing for events to be correlated with an accuracy below few tens of milliseconds. ProWare middleware absolves the sensing and INDP applications running on network nodes from locating and contracting data providers themselves. Nodes need only to specify the type of data they produce and data they consume (when the need arises) and ProWare data mediator is responsible for arranging the communication. A node may be both a consumer and a provider, depending on the situation and on its functionality. In Figure 19, node 1 is a data consumer for node 2 and a data provider for node 3.

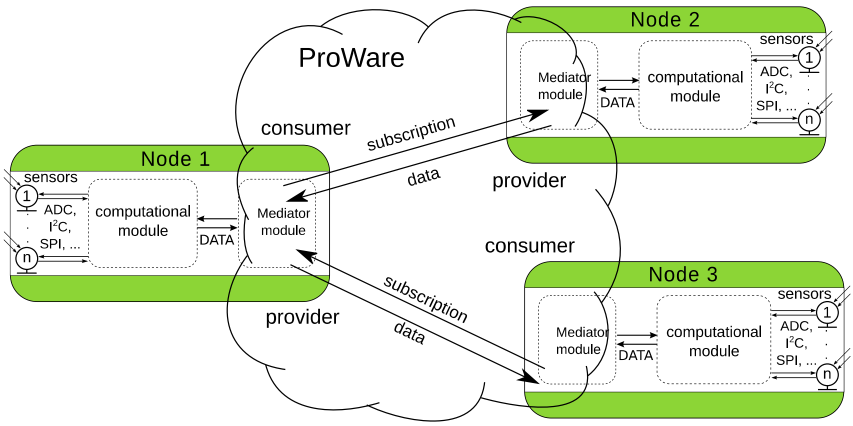


Figure 19 – Network nodes equipped with ProWare forming communication links

A similar and interesting approach has been taken by Berkeley university [85]. In this work Marten Lohstroh and Edwards Lee investigate an approach where they make use of "accessors", which have similar mediation function and act like proxies between heterogeneous IoT devices (actors). This approach is similar to the one used in this thesis, where this thesis makes use of ProWare mediator in order to interface heterogeneous systems. However, Berkeley team does not use "accessors" to manage the interactions and information flows. ProWare mediation functions allow also to manipulate with the validity metadata of the data. This is done via on-line adjusting and checking of the contextual constraints for exchanged data. Theoretically, such an architecture, enhanced with mediator software facilitates predictable operation also in a changing SoS configuration.

3.4.3 Subscription message structure

A simple subscription message structure is depicted on Listing 1. The subscription is represented in XML format, this makes it human-readable. The sensor nodes make use of a compact binary encoding [95]. The XML message construction scheme, where an object - subject - value approach is used, allows a flexible message structure. It is is a structure, where each object has a type, an optional value and an optional reference to another object. Having a reference to another object makes the referring object a child of the referred. Having no reference places the object in the root of the message. The values are currently limited to integer types or byte buffers. Object names in the binary encoding are replaced with integer identifiers, an info byte is used to declare the structure of the object. Object identifiers and values are encoded with the least possible bytes. The structure of the XML is preserved using tag indexes in the binary representation and subject fields that reference the indexes.

Listing 1 – Generic subscription message structure

```
<subscription_grouping>
    <subscription_parameters/>
    <request_grouping>
        <data_object>
            <action_grouping>
                <action>
                    <trigger_grouping>
                        <constraints/>
                        <temporal constraints/>
                        <spatial constraints/>
                    </trigger_grouping>
                    <input_grouping/>
                </action>
            </action_grouping>
        </data_object>
    </request_grouping>
    <subscriptions_grouping>
        <additional_subscription_information/>
    </subscriptions_grouping>
</subscription_grouping>
```

The subscription structure is hierarchically divided into groupings, to identify properties of various objects by their nature and to group the properties together by their context. Groupings usually do not carry a value. Subscription messages start with various subscription parameters and then contain a data object that marks the desired data type. Action (or control) groupings are used to initiate read and write operations. Various conditions can be placed to control the behaviour of the subscription. An action grouping can contain several actions. Actions are evaluated in the order they are represented in the message.

Subscriptions must have identifiers for both the consumer and the subscription itself. The suitable data producers can be identified in the network by either its ID or by specified constraints. Subscriptions also must have a start and end times for the subscription, which specify when the execution of a subscription should be started and when it should be terminated (this should not to be confused with a trigger event, which can also be just a start time, that may start a specific action of the subscription e.g. a measurement of a physical phenomena). As sensor networks typically do not use global synchronisation, the start and end times are converted to age (i.e. the end time can be translated as elapsed time since subscription acceptance).

A simple subscription must have at least one or more parameters that it either reads or writes or both. This manipulation can be done once or periodically more than once; or triggered by another event once or more than once. Each input parameter will also have constraints for validity of the data (e.g. spatial and temporal). For example spatial constraints for read or write actions specify the area where the parameter must be measured from or have effect on. A spatial constraint can define a specific device in a network, its field of view or effect area by geographic position or a polygon. Temporal constraints can describe the valid duration of an effect for write actions or validity period for read actions.

Triggers usually start some action. Subscribed manipulation can be stopped at any time by respective stop subscription command.

Listing 2 – Example subscription for microphone array sensors

```xml
<xml_packet>
<dt_subscription>
    <dt_priority value="99"/>
    <dt_type value="dt_threat">
        <dt_constraint_spatial>
            <dt_provider buffer="0151F40E150000F7"/>
        </dt_constraint_spatial>
        <dt_actions>
            <dt_read>
                <dt_operator_and>
                    <dt_time_passed_ms>
                        <dt_timestamp_function_done_ms/>
                        <dt_greater value="2000"/>
                    </dt_time_passed_ms>
                    <dt_data_source value="1">
                        <dt_type value="dt_acoustic">
                            <dt_constraint_temporal>
                                <dt_operator_and>
                                    <dt_time_passed_ms>
                                        <dt_timestamp_ms/>
                                        <dt_less value="4000"/>
                                    </dt_time_passed_ms>
                                    <dt_latest/>
                                </dt_operator_and>
                            </dt_constraint_temporal>
                        </dt_type>
                    </dt_data_source>
                </dt_operator_and>
            </dt_read>
        </dt_actions>
        <dt_subscriptions>
            <dt_data_source value="1">
                <dt_type value="dt_acoustic">
                    <dt_constraint_spatial>
                        <dt_provider value="0x0005"/>
                        <dt_provider value="0x0006"/>
                        <dt_provider value="0x0007"/>
                        <dt_provider value="0x0008"/>
                    </dt_constraint_spatial>
                    <dt_actions>
                        <dt_read>
                            <dt_time_passed_ms>
                                <dt_timestamp_function_done_ms/>
                                <dt_greater value="1000"/>
                            </dt_time_passed_ms>
                        </dt_read>
                    </dt_actions>
                    <dt_constraint_temporal>
                            <dt_time_passed_ms>
                                <dt_timestamp_ms/>
                                <dt_less value="2000"/>
                            </dt_time_passed_ms>
                    </dt_constraint_temporal>
                    <dt_metadata>
                        <dt_coordinates_utm/>
                        <dt_orientation/>
                        <dt_address/>
                    </dt_metadata>
                </dt_type>
                <dt_expected value="4"/>
                <dt_requirements>
                    <dt_all/>
                </dt_requirements>
            </dt_data_source>
        </dt_subscriptions>
    </dt_type>
</dt_subscription>
</xml_packet>
```

One example subscription is described by Listing 2. This is a high level subscription sent to fusion node, which task is to combine data from four microphone array sensors. It can be noted that in this case all providers are addressed by their ID-s. The fusion node executes every 2000 milliseconds and it has a temporal validity constraint of 4000 milliseconds. The fusion node mediator will decompose this subscription and send out four separate simpler subscriptions to the microphone array sensors. The four simple subscriptions for the microphone array sensors define the four specific devices, their reporting interval as 1000 milliseconds and maximum temporal validity interval for produced sensor readings.

3.4.4 Mediation enabled collaboration in Smart Environments

This thesis uses the mediation service as a coordination fabric for sensor network information flows. Mediation via manipulation of validity metadata enables management of the interactions and fosters collaboration between distributed network nodes. The autonomous nodes in the sensor network are modelled as agents. The systems of such IoT devices can in turn be modelled by multi-agent systems (MAS). Autonomy property in such models means that the inner logic of individual agent is hidden from other autonomous agents. One can even consider that each autonomous agent in a MAS could make use of a different model of computation. Such IoT devices need mediated interactions in order to collaborate. ProWare mediators create a coordination environment, which allows to dynamically filter out the interactions which are not relevant or valid. This includes both read and write actions. This way an effective coordination environment is created where systems of autonomous agents can execute, plan and reason about actions together. An example of collaboration is grouping a certain number of nodes into a dynamic subsystem by subscriptions that request situational information from an area which overlaps fields of views of several distributed sensor nodes. Microphone array sensors are a good example for this use case. Several closely deployed microphone array sensors can detect vehicles in the same area. Despite the fact that each sensor node operates autonomously and possibly asynchronously from the other microphone array sensors, a specially assigned network node (or agent) can compose subscriptions specifically tailored for each individual data producer - microphone array sensor (not only by node id-s but also by their context e.g. geographic location). The specially assigned node receives data streams and checks the validity information from each stream, evaluates if the validity intervals of received data are both valid and overlapping and only then uses the data to carry out the computation of possible vehicle position.

Another example of ad hoc collaboration between autonomous network nodes is when a new node is expected to join the network. First the newly joined node publishes the list of the services it can provide. The services can include types of sensory data it is capable to provide, but also actions in case if the device configuration allows. This list of services is stored in the network, usually by a network node which is computationally and energy-wise better positioned - called a subs-server. When a certain service is required by a node in the network (either data or action), then the smart middleware mediator turns to the subs-server and checks if the service is available. If it is, then the mediator connects the service provider and service consumer. The mediator creates a data exchange and delivery agreements between the service provider and service consumer. Publication IV describes that the capability of dynamically organising local interactions, based on actual needs of the users is especially suitable for INDP tasks. Since in practice the higher quality of the INDP results can be achieved if the combination or correlation of data can to take place between nodes that are in close physical proximity and only the correlation results needs to be communicated further in the network and the computation can remain local.

3.5 Situation parameters

This sub-chapter gives an overview over the concept of situation parameters and explains its role in the current work. Section 3.5.1 gives a formal description of situation parameters referring to some extent to the previous work that has been conducted at the Research Laboratory for Proactive Technologies. Some more novel ideas suggested by author are described in Section 3.5.2 and the last Section 3.5.3, which also gives a brief explanation of the relationship between context and situation and describes what is the validity of situation parameter and how it is used by the concept of mediated interactions.

3.5.1 Formal description of situation parameters

The definition of situations is provided in [94] as: *a situation is the aggregate of biological, psychological, socio-cultural, and environmental factors acting on an individual or a group of agents to condition their behavioural patterns. Here agent denotes natural (i.e. humans) or artificial (i.e. computing systems or software-intensive multi-agents) agents, and environment means mix of natural or artificial environments.*
This definition suggests that a situation can be subdivided into a number of factors or situation parameters. These specific parameters, defined by the SA consumer, can be used to detect a situation that has the potential to influence the consumer agent's behaviour. It can be said that a situation is an aggregate of factors (described by situation parameters) relevant to a given agent and for its task. The number of parameters can be very versatile and potentially very numerous, exactly as different situations. The situation parameter is a predefined situation description in a situation ontology. Formally, each situation is defined by a 3-tuple $S = \{S_p, S_t, S_a\}$, where S_p denotes a set of situation parameters and S_t and S_a comprise, respectively, temporal and spatial information about the situation. A situation parameter Sp is defined as any type of information that is used in the process of inferring higher level situations, as described in section 3.7. The notation of situation parameters allows for general discussions over sensor networks requirements and feasibility at the preliminary design phase and technical discussions at the detailed design phase, while also transferring seamlessly between the different phases, when changes need to be made. High-level sensor network architecture can be decided without the need to specifically define S_p, S_t and S_a, these are only determined at the detailed design phase, where all relevant factors (environmental, technological, etc.) are taken into account.

The hierarchical system of situation parameters is modelled as a K-ary tree. Each situation parameter in turn can be interpreted as a situation and each situation can be used as a situation parameter for composing more complex situations. Situation parameters allow decomposition of high level situation detection tasks into several lower level situation or event detection tasks. The lowest level situation parameters can be associated with a specific sensor reading or derived from one using predefined situation models. In this way, the situation parameters at the lower levels are produced already at the edge of the network. The flow of the situation detection is modelled from leaf nodes towards the root of the tree, where the root node corresponds to the final constructed abstract situation. Each situation recognition sub-process is implemented as either a local in-sensor data processing or an in-network data processing algorithm which can receive inputs from several disparate sources. However, the necessary situational information for situation detection can come also from the other direction. For example a cloud service could provide necessary context for specific situation at leaf node level. There are no limits on how this hierarchical tree of situation parameters is decomposed, however in practice it depends on how well the application domain is analysed and which predefined situation descriptions (situation ontology) are available [56].

The situation parameters can be numeric variables or other situations, representing the hierarchical nature of situations. For example, the temperature of an area is a basic situation $S_{temp} = \{S_p, S_t, S_a\}$, where S_p is a unit set holding the actual temperature value. A similar construct can be made for humidity S_{hum}. A higher level situation is composed of other situations either by fusion, for example, $S_{operation} = \{S_p, S_t, S_a\}$, where $S_p = f(S_{temp}, S_{hum})$ is a function of temperature and humidity situations, or by aggregation $S_{weather} = \{S_p, S_t, S_a\}$, where S_p is a two element set $S_p = S_{temp}, S_{hum}$. Here, $S_{operation}$ is for instance the suitable environmental conditions for the operation of some device and $S_{weather}$ is a set of weather parameters. Temporal and spatial information, S_t and S_a, are defined based on application needs and may include different information, for example, constraints, requirements and/or other data necessary for the interpretation of situation parameters S_p.

The complexity range of situations can vary a lot. The hierarchical principle for defining situations allow for a very dynamic and open system of situation hierarchies to be created. On one side there are relatively simple situations created by a bounded environment where sensors have collected data about some specific physical phenomena and produced respective situation parameters which in turn have been used to estimate the ongoing situation S_p regarding the physical phenomena under question. However, on the other side, it is possible to describe extremely complex situations composed of situation parameters collected from Smart Environments in an urban environment or comprehensive situation awareness of a nation [97]. In the latter case the number of situation parameters S_p can grow exponentially if not abstracted to a higher level already at the edge. Similar analysis can be made about temporal and spatial information S_t and S_a about the situation S_p. In a bounded environment case both temporal and spatial intervals can be rather small. In case of situational information collected by a large sensor networks for example about a situation of nation, the constraints on temporal and spatial intervals can inevitably grow together with the process of moving upwards in a hierarchical abstract tree of situations. Although this thesis brings examples of Smart Environments comprising of a wireless distributed sensor network, the thesis also suggest that same methods apply on the higher levels of abstractions, e.g. when estimating a comprehensive situation awareness for a nation [97].

3.5.2 Relevance of a situation parameter

In many cases some types of information are more important than others for inferring a situation, e.g., high body temperature may be a strong indication of a general sickness of a person while other attributes may not be so important to infer that specific situation. To model this difference in the importance of situation parameters for inferring a situation, this thesis suggests to define the relevance function, which assigns weights to situation parameters regarding their relevance. The weights reflect how important each parameter is (relative to other parameters) for describing a situation. The relevance of situation parameter, models the relative importance between the situation parameters of a situation space. Relevance of a situation parameter can be later used by INDP algorithms to decide which streams of situational information are more important. The relevance of a situation parameter is provided by the consumer node, which also generates the subscription (a subscribing agent). The relevance is used at least in two cases, firstly in the prioritising streams through the network and secondly later when combining the situation parameters in order to detect or identify a situation of interest. In the former case the network node that needs to relay messages can raise or lower the priority of messages based on the relevance of situation parameters. For example in case the amount of traffic temporarily exceeds the network capability then the messages containing situation parameters with lower relevance are allowed to be buffered until the communication load decreases to an acceptable level. For the latter case, this thesis suggests that when combining the situation parameters into a situation of interest, the most relevant situation parameter should be taken as a reference parameter when selecting other compatible situation parameters. The specific method how the selection process is carried out is explained in Chapter 4.4.6. Selecting the most relevant parameter as a reference helps to find other such parameter which are valid in the same context. The contextual validity is explained in Section 3.5.3.

3.5.3 Contextual validity of situation parameters

The contextual validity interval is an assessment about contextual bounds or thresholds within which some specific piece of information is valid. In other words, if the piece of information is out of the contextual validity interval, then it should not be used as it could lead to a an incorrect assessment of the situation. Before continuing, a very brief explanation must be provided, of what is context and how it is related to situations. One of the first widely known definitions of context is given by Schillit et al. in [127]. Schillit et al. worked with ubiquitous systems and considered context to be all information that can be acquired by any means (e.g. by ubiquitous systems such as sensors) and which can describe the current environment where the agent is in. However today the scientific literature has moved on from this simplified description and accepted a more general definition. For example, according to Padowitz et al. in [100], the paradigm of context-aware computing can be regarded as an attempt to obtain information with limited sensing capabilities, but which nevertheless reflects circumstances useful to the application at hand. According to Padowitz et al context awareness gives a system the ability to act based on the state of the world around it, it gives the system ability to achieve situation awareness. This is similar to the approach described in this thesis. However, this thesis uses SA based approach from the perspective of cognitive agent, who subscribes to situational information and suggests that situation parameters can only be valid in a specific context. A situation parameter is sensor data that is specifically conditioned according to the context required by user for inferring a specific situations. Situations in turn are hierarchically composed of situation parameters belonging to the same or overlapping context. The latter is called contextual consistency. Performing in-network data fusion and aggregation in a reliable manner requires contextual consistency of data collected from distributed sensor nodes (e.g., in temporal and spatial domains). In order to ensure the usability of fusion and aggregation results, contextual constraints must be applied to collected data. Spatial constraints, such as bounds to the area of interest, and temporal constraints, such as acceptable age interval of data, are defined within the subscription made to the Smart Environment by the user. However, the contextual constraints are not limited to temporal and spatial measures, other norms (e.g. confidence, reliability and relevance) may be included. In any case, data providers, the different sensor nodes, must augment all measured data with appropriate metadata tags (e.g. temporal and spatial), which are later used in the data validation process. The spatial information is often provided to the network nodes by system designer, in case nodes are not capable of computing their on spatial position. However, to enable, the on-line temporal checking of data the network communication layer must in addition support measuring packet delivery time or inform data provider and consumer of failure to (temporarily) meet this requirement during operation. In an extreme case of distributed computing in ad hoc distributed multi-hop networks, the case is that the global clock synchronisation within the network distributed sensor nodes is not viable, see Section 2.3 and other methods, such as computing the accumulated delays during the data transport, must be used, see Section 4.3.6. In the concept of distributed computing each node has its own private memory, processing and an independent time keeping system. This is especially true in case of embedded systems that are building blocks for Smart Environments. The network architecture presented in this work utilises a messaging syntax and communication protocol that facilitates satisfying the above described data validity needs [95, 107].

When a cognitive agent who is building up SA, requests data about a certain situation, the response to this request can for example be a single sensor measurement or a stream of measurements. Without any context, this measurements could in principle be interpreted in many different ways. The interpretation of the sensor data depends on the contextual information such as time and location. In order to ensure the usability of sensor readings for the consumers SA needs, the consumer defines bounds to the contextual validity of requested data. These bounds are defined in the subscriptions and are also used as instructions for data producer to data production, communication and both in-sensor and in-network data processing. It can be said that the added validity metadata gives context to the required sensor measurements. Outside this context, the sensor measurements could mean something entirely else, they would become invalid.

All nodes in a distributed network that are used to build up situational information are essentially equal, their roles in the network are determined dynamically at run-time and automatically adapted to changing conditions. The use of the proactive mediator ProWare enables to set validity intervals and validity polygons (in case of spatial meta-info) for data that is requested from other systems and check the validity of the data in the context of the constraints on-line, while the data is being exchanged. The data consumers do not subscribe for service/data from specific producer, but to service/data constrained by type, time and location. For example, if the network topology changes while the data is being delivered, the differences in end-to-end delays from disparate sources can make fusion of situation parameters impossible. This is the case if the data from one data source arrives with a considerably different delay and becomes incompatible with the data from other sources. In this case the ProWare mediator firstly disregards the data items which are incompatible and secondly gives a possibility to renew the subscriptions to the data sources and to choose such validity intervals that the data may become consistent. This depends also on what is the maximum age for data that is used for SA, as ProWare cannot force a change in network topology that causes long delay, but rather request other nodes to buffer data to create longer delay data from also other sources. Currently a human operator or an engineer who adjusts the subscriptions to improve consistency, is needed. The producer (an autonomous agent such as UAS, self driving vehicle, sensor node) decides on its own if it is able to provide data that satisfies the resources and constraints.

On Figure 20 the concept of applying ProWare in a distributed INDP scenario is depicted. Every system that is part of the SoS has a ProWare component in it, which is responsible for interactions between the systems. The ProWare components makes requests for data (in the form of subscriptions) to other systems, makes data exchange agreements, delivers data generated by the local system to other systems and validates the data.

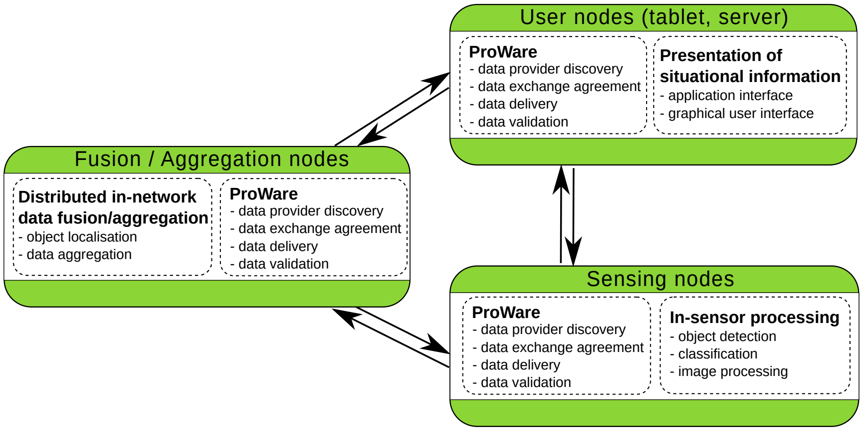


Figure 20 – ProWare mediator applied in a distributed INDP scenario.

3.6 In-sensor data processing

Sensor data processing on the edge of IoT requires that the computation is moved from the cloud to the edge of network and as close as possible to the embedded systems and sensor nodes themselves. This section will give an overview of in-sensor signal processing methods, also demonstrated in the experiments described in Chapter 5.

The Publication II by Ehala and Kaugerand provides descriptions of three applications of in-sensor signal processing: 1) identifying and counting objects in video-streams, 2) classification of objects based on sound signals and 3) estimating the direction of arrival of sound signals (to sensor node). The choice of applications for this work was motivated by the European Defence Agency (EDA) project IN4STARS 2.0 (Information Interoperability and Intelligence Interoperability by Statistics, Agents, Reasoning and Semantics) for which the applications and ISR requirements were developed. All these applications were applied in the use-case scenario described in Publication II.

Identifying and counting of objects made use of a Raspberry Pi platform and its onboard camera. The results were communicated to a subscriber by an attached WSN transceiver device called MURP, described in Publication II and briefly in Chapter 5.1. The application counted all moving objects in its field of view. According to the scenario of the exercise, the use case for this application was detection of a crowd gathering at monitored area. The second application of classification of objects based on sound signals made use of BeagleBone Black platform and a linear array of 6 microphones (where data only from one microphone was used for classification). The results were communicated to a subscriber by same way as described above. The application was able to detect and classify different vehicles. The use case of this application in the scenario was the detection a situation of presence of military vehicles. The third application of estimating the direction of sound signals was based on the same BeagleBone Black platform. The use-case of this application was to provide input to a INDP node which upon receiving directions to same sound source from distributed nodes was able to compute the location of the sound source. The details for these applications can be found in Publication II.

Another example of in-sensor signal processing is described in paper [119]. This is a cost effective COTS (commercial off the shelf) microwave radar. The application is developed for urban conditions for measuring traffic density. The data processing is basically based on fourier transform to estimate the vehicle passing moment and speed. A very similar approach to radar signal processing is analysed in [27]. This sensor is used as an example to demonstrate a building block for a Smart Environment.

For each application presented, signal processing was done locally on appropriate sensor nodes using local data acquired by the node itself. No additional information or data from outside were needed once operation had started. The difficulty of performing in-sensor signal processing lies mainly in efficiently coping with the limited resources and constrained computational power of IoT and WSN computing devices.

3.7 In-network data processing in ad hoc networks for Situation Awareness

This section explains the concept of in-network data processing (INDP) and describes its two main methods aggregation and fusion in case of distributed network nodes. The concept of INDP means that data is processed within the network and that the input data for data processing algorithms originates from distributed network nodes. While the concept of in-sensor signal processing represents methods and technologies for achieving advanced computational results on single device level, the concept of INDP allows considerably more flexibility but at the same time introduces a new level of complexity. Both concepts, advantages and introduced complexity will be elaborated below.

INDP methods do not directly assume the availability of cloud computing. It is rather assumed that the nodes collect the data not only by sensors but also by exchanging data with multiple functionally disparate and spatially distributed network nodes as described in Publication II.

In Publication II authors describe a military use case where in-the-field units need situation awareness information relevant to them in a timely manner and preferably directly from network nodes as depicted in Figure 21 and in human understandable form. This is one of the main reasons why useful information must be provided already at the network level utilising the processing capabilities of the network nodes themselves. As the alternative solution of remote data analysis, which requires central data processing, can consume too much time and network bandwidth.

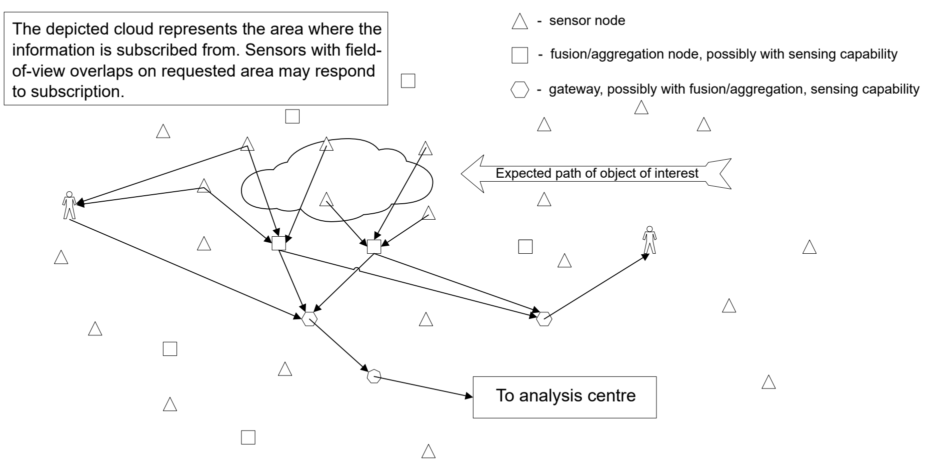


Figure 21 – A generic use case for INDP.

Figure 21 depicts a sensor network that consists of different nodes, some capable of only sensing, but some with higher computational capability also capable of INDP. The situation in the figure is that an object of interest is about to move through a monitored area. It is expected that the situational information about the object moving through this area is reported both to in-the-field users, who have a direct connectivity with the network and also to a remote server for data storage and long term analysis. It can also be noted that for this use case, only a handful of network nodes are activated by subscriptions. The system of network nodes for data collection is designed when it is needed, not off-line before the deployment of the network. The main motivation for the INDP is that it enables information production already very close to the real phenomena of interest and the SA information consumer can get initial estimation earlier than would be possible via for example cloud solution. Creating situational information already at the lowest levels of the network helps to avoid transmitting large amounts of sensor data to the cloud and enables providing situational information to local users directly. This is depicted on figure 22. On figure 22 subfigure a) depicts the case where all sensors provide data to the cloud directly, albeit via the gateway. In this case the amount of data transmitted via the gateway increases as a function of the number of data producing sensors. On the other hand subfigure b) depicts a case where INDP methods have been applied to produce situational information in-situ. In this case the nodes exchange situational parameters and produce situational information at a higher abstraction level as requested by users.

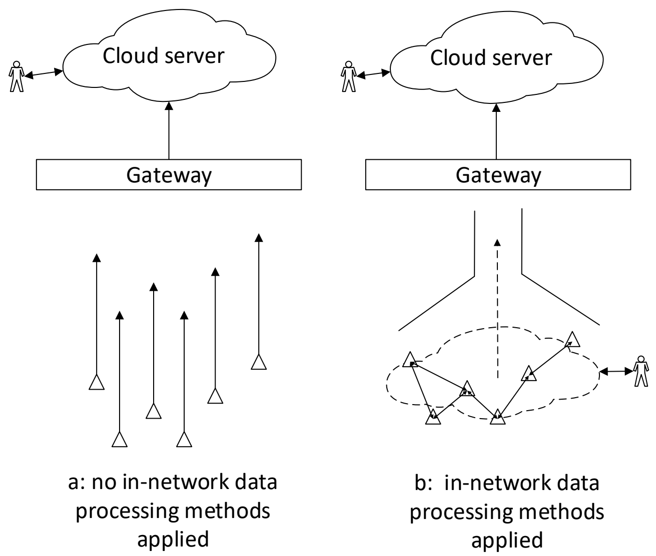


Figure 22 – In-network vs no in-network data processing.

As explained in 3.5.1, the situational information is built up hierarchically, this is another advantage supported by INDP. The hierarchical build-up of INDP can be used to build up hierarchical models of situations already within the network. This is the opposite to classical data driven sensor networks where data is processed in a stove-pipe like manner without any concern to users needs. Having INDP methods available enables horizontal data exchange between the network nodes. The following two sections provide an overview of the two INDP methods in distributed sensor networks for Smart Environments. Data aggregation and data fusion:

3.7.1 Data aggregation in Smart Environments

Data aggregation techniques for WSNs have developed along with the advancement and spread of WSN technology. The main motivation for data aggregation has been energy efficient data collection to extend network lifetime and improve the quality of WSN service [115]. Different data collection and processing schemes arrange that not all data are individually transferred to the network sink, but rather related data are accumulated temporarily somewhere in the network, where it is aggregated and only the results are forwarded to the sink. This reduces the amount and length of messages transmitted and subsequently saves energy and bandwidth. Example use cases of data aggregation in WSNs include Jo et al.[59] and Ramesh [116]. But this type of data aggregation is suitable for collecting the data to a central server. This type of sensor data aggregation does not serve the purpose of the situation awareness applications.

The aggregation of the data this thesis presents does not only serve the purpose of saving energy, but also aims to improve the situation awareness of data consumers. Although, majority of WSN in-network aggregation focuses on aggregating only the same type of data, there are two types of aggregation - lossy and lossless aggregation. The examples of lossy type of data aggregation operators include accumulation, summation, counting and finding values for average, minimum or maximum. In this approach, size of the packet is reduced, as only calculated value of aggregate function is inserted into the packet after compression, rather than sending the whole packet of every node. On the other hand, in lossless aggregation, the data is packed into bundles and all data is delivered to the recipient. This is very similar to the data fusion as it combines data from different types of sensors to characterise a specific event detected by the Smart Environment. The difference between data fusion and data aggregation is that different data types of sensor readings are not combined into bundles, but instead are fused into new types of data.

3.7.2 Distributed sensor data fusion

In narrow sense, the sensor data fusion belongs to a very specific field of science and is based on very clear mathematical methods. However, this thesis considers sensor data fusion in a broader context i.e. as a building block for SA. SA applications require data fusion and a single sensor is usually not sufficient for observing all required aspects of relevant situations. There can even be found applications that can be difficult to implement even by using all realistically available data. This can be the case, for instance, for urban environment monitoring (e.g. city traffic monitoring where vehicles of various sizes and speeds need to be detected and persistently tracked in the streets with many lanes), calling for innovative sensors and technologies. However, on the other hand, it is clear that through combination of data from different sensors i.e. sensor fusion it is possible to overcome some of the limitations of single self-reporting and observation-based sensor systems. Different sensor-technologies for monitoring different modalities present by their very nature different sampling intervals, data latencies (i.e. variability of the time lag between data collection and when the data is made available), errors and uncertainties. Classically, the term sensor fusion describes the combination of sensory data acquired from a number of sources such that the resulting integrated information has less uncertainty than viewing the same sources individually [15]. A more specific definition of fusion is possible regarding of the context and the purpose of the fusion, i.e., in particular the components to fuse. However, in general, the classical assumption behind the application of fusion is that fusing datasets from different sources improves the performance of the subsequent data processing [15].

In 2009 Lambert et al. defined the sensor data fusion more generally to be the process of utilising one or more data sources over time to assemble a representation of aspects of interest in an environment [75]. Lambert's main idea was to provide a design for sensor fusion for higher level SA. This broader and considerably more general approach fits well with SA requirements described in Section 2.4.1. Distributed sensor data fusion in the context of current work serves mainly the purpose of enhancing SA of the data consumer. For this reason, the sensor fusion is considered to be an integration of sensor data with different modalities and when a new type of data at a higher level of abstraction is created. The traditional roots of the data fusion community are in sensor fusion, where the "data sources" are established sensors and the "aspects of interest in the environment" are moving objects, each typically represented by a set of state vectors. The broader definitions reflect an increasing emphasis toward generalising sensor fusion into so called higher-level fusion, in which "the aspects of interest in the environment" are not restricted to objects [75]. This idea also supports our hierarchical buildup of predefined situation detection mechanism using concept of situation parameters.

The fusion of sensor data is a necessary step in order to detect the actual situations in complex environments where the sensors reside. Increasing the variety and number of sensors used, as is done with the large scale wireless sensor networks, together with the relevant performance and limitations information, results in a more complete picture of what is happening in the monitored environment (e.g. smart city traffic environment).

Classically the inputs for data fusion are acquired by data mining and stream processing techniques in a central server where all sensor data has been collected and stored. However, Smart Environments require situation detection already within the environment itself, which in turn requires conducting sensor data fusion online and close to the edge. This enables providing the situational information already within the environment to its inhabitants. At the edge the sensor data fusion can be carried out locally, where an individual sensor nodes comprises different types of sensors fuses the data acquired from sensor readings and communicates the fusion result to consumers. This is local data fusion. However, the SoS architecture requires also that the individual nodes exchange the situational information. This leads to the distributed sensor data fusion. In this case, the data from disparate sensor nodes are collected by a prefixed sensor/fusion node that performs the fusion process and distributes the result to data consumers. Examples of distributed data fusion in wireless sensor networks are described in Mayk et al. [89], Bahrepour et al. [10] and Lai et al., [74] where events detected and sensor readings collected by individual sensor nodes are combined to a new data type by a fusion node. However, these references do not discuss the consistency and validity of the inputs for the fusion algorithms. In cases of large ad hoc sensor networks and especially networks for SA solutions, Preden et al. emphasise that it is important that the consistency and validity of the fused information is analysed online [107].

By using online stream processing techniques at the edge both in individual nodes for local fusion and in a distributed nodes for distributed fusion, it might be possible to identify more complex situational parameters (situations) even such as distribution of speeds along the traffic lanes, vehicle travel times, vehicle type, weight, length, etc. The detected situations form the basis for predicting developing situations in short term future or detecting low likelihood behaviours (anomalies). (The detection of anomalies in city traffic can be based on specific predefined rules (e.g. maximum allowed speed). The situation detection assumes the knowledge of behaviours that need to be detected. An alternative approach is to establish the "normality" of vehicle behaviours and patterns and to detect deviations from such "normality". The resulting deviations would be flagged as "unexpected" and automatically brought to the attention of the SA user. Predictive analysis of vehicle positions can be performed using many different approaches, ranging from a simple linear model (usually valid for short propagation times of the order of seconds) to more complex context-based methodologies. Data driven approaches are based on the assumption that vehicles are mostly compliant with regulations so that by observing city traffic for a sufficient amount of time, one can extract the routs of the main traffic flows. The behavioural characterisation of vehicle activities in turn enables the understanding of collective urban uses. Moreover, the use of behavioural analysis can help in identifying the vehicle type based only on its dynamics. This also further enhances urban situational awareness enabling the verification of the detected vehicle type.)

4 Modelling mediated interactions

This chapter gives an overview of the necessary theoretical concepts for modelling mediated interactions for SE applications required for SA. The chapter starts with a short section describing the previous work and then continues in next section with short overview of the requirements for modelling mediated interactions. Then the Sub-chapter 4.3 continues with a description of several technical terms (e.g. temporal and spatial validity, consistency, overlap of validity intervals, simultaneity and end-to-end delays) necessary for achieving contextual consistency of mediated data in Smart Environments. Finally this chapter ends with Sub-chapter 4.4 for describing how network nodes and mediated interactions are modelled. The latter sub-chapter explains the concept of time-selective communication and provides description of the alignment and selection algorithm.

4.1 A brief overview of history of modelling mediated interactions

In [94] Leo Mõtus introduced an intelligent channel as an interaction mediator for improving achievement of the team situation awareness. This mediator is modelled as a channel function in the Q-model formalism. In [95] the concept of mediated interactions is elaborated further and applied for improving the self-awareness of the system architecture in order to detect and manage emergent behaviour, this has been elaborated in 3.4. The work [95] considers an ad hoc sensor network as a system under study which inherently consists of autonomous distributed components. In [111] Preden continues this line of work and describes a smart mediator, which makes use of validity information in order to carry out the on-line validation of interactions. In Publication II, Ehala and Kaugerand go further with this work and describe INDP in WSNs using Q-model formalisms such as processes, channels, execution timesets, temporal constraints and validity intervals. They use the concept of an asynchronous channel from the Q-model for modelling the service of data exchange provided by "ProWare" middleware. Although they do not explicitly mention this, the paper makes use of the concept of time selectiveness as a mediator function. The concept of time-selectiveness is elaborated further in current Chapter 4. The respective experiments and results are given in the next Chapter 5. This Chapter 5 demonstrates that selecting the suitable sensor readings from multiple streams in order to achieve relative consistency (explained in sub-chapter 4.3.3) of INDP inputs improves the quality of the fusion result. This thesis connects the dots and demonstrates that applying the concept of mediated interactions in Smart Environments and that Q-model is suitable for modelling the mediated data exchange in Smart Environments.

4.2 Requirements for a mediator model

A model is a depiction, representing the original. However, a model is always also an abstraction, and should be capturing only relevant aspects. A model has a purpose defining its use. Models can be used for documentation, communication, formal verification of properties, model checking of properties, test generation, specification, simulation (one model, one behaviour), for calibration, optimisation, application synthesis, software, FPGA, 3D printing, etc. A model may be universal or special, while a universal model should be applicable across a field of applications regarding the domain. Examples of universal models are Universal Modelling Language (UML) and Turing model. On the other extreme there are special models, in case of special modelling method each model usually characterises only a single aspect of a complex system. The Q-model is somewhere in the middle. For example, it has previously been demonstrated that Q-model and UML can be complementary models [126]. The Q-model can be used to describe temporal aspects of interactions between distributed systems, UML can be used to model different aspects of entire system. It has been shown that Q model can also be used to describe, model or study the behaviour of systems in order to detect emergent behaviour, learn to avoid negative emergent behaviour and to take advantage of positive emergent behaviour [95]. In distributed systems and especially in the domain of IoT, there are many aspects that need to be modelled, there are functional models, security models, fault models, interaction models, etc. Most relevant of them for modelling mediated interactions are interaction models. The class of interaction models is a class of specific models which can be modelled by process models, actor models, or even automata and state-machines. However, applying interaction models for distributed systems which have no notion of global time can become complicated.

One of the purposes of modelling is to formally and logically investigate systems and their components' behaviour. In the case of distributed systems, to which Smart Environments definitely belong, Jeff Kramer suggested already in 1994 in [72] that the analysis using first order calculus (most temporal logics) is undecidable for logical reasoning and that process algebras are required. Some examples of computing models based on process algebras [8] are Hoare's Communicating Sequential Processes (CSP), Milner's Calculus of Communicating Systems (CCS), Bergstra and Klop's Algebra of Communicating Processes (ACP), the Pi-Calculus also developed by Milner and the Q-model developed by Rodd and Mõtus. The Q-model can be described as multi-stream interaction machine (implementing a super-Turing model of computation) [94].

Managing interactions in Smart Environments designed as Systems of Systems consisting of Cyber-Physical Systems requires a systematic approach. SoS consisting of CPSs can be considered a real-time distributed systems, they are also called CPSoS. Typical CPSoS have several processes executing at the same time. This is especially true if the system components are embedded in Smart Environments and are autonomous. Such embedded systems can be considered smart agents which together form a SoS (could be viewed also as MAS), where each agent has its own time counting mechanism. However if there is a need to implement INDP algorithms, then the autonomy of the agents in SoS in turn leads to the necessity of modelling several time systems at the same time. One of the goals of this thesis is to improve relative spatio-temporal consistency of input data for INDP algorithms. The thesis claims that a concept of mediated interaction could be one potential solution. For this a model that can describe certain system behaviours and properties is required. Smart Environment composed possibly of WSN nodes is viewed here as a System of System. The model should be capable of considering following:

1. System is composed of components that are loosely coupled and in interaction with each other.

2. All components can also function independently of the system.

3. Global synchronisation of system's components' clocks is not feasible.

4. Distributed and parallel execution of processes. Disparate nodes in WSN can observe aspects of same physical phenomena simultaneously, and also process data regarding same physical process simultaneously (although, each from its own viewpoint).

5. Communication partners have selective temporal and spatial access to situational information.

6. Transporting a message through network takes time, the model should be capable of modelling the channel delay.

7. Communication for exchange of situational information is asynchronous.

8. Components and their processes have temporal characteristics that must be taken into consideration.

The Q-model satisfies all these requirements. All these aspects are elaborated in separate sections. However, in following a superficial overview how Q-model satisfies abovementioned requirements is given: 1. The Q-model processes can be used to describe SoS components that are loosely coupled and in interaction with each other. 2. The Q-model can also describe processes that operate asynchronously, thus being independent of the rest of the system. 3. Although the Q-model assumes a global clock synchronisation, it has a capability to analyse timing aspects also by taking into account clock imperfections of individual processes. 4. The Q-model allows description of forced parallelism, that occurs when the outside world determines when distributed processes must be executed in parallel. 5. The Q-model inherently provides the capability to describe time-selective inter-process communication. 6. The Q-model implicitly enables modelling of the channel delay in process execution time in the allowable delay between the start of a process and the request of data. This thesis includes another variable into the formalism used in 4.4.1 in order to model the channel delay explicitly. 7. The Q-model enables modelling of asynchronous channels for modelling asynchronous interaction between processes. 8. The Q-model includes capability to model both performance-bound properties of a system and also time-wise correctness of events and data.

4.3 Contextual consistency of mediated data

To eliminate the possibility of misinterpretation it must be stated that this chapter does not discuss context awareness in a sense, where context awareness involves knowledge about ones surrounding context data. Instead, this chapter is discusses relevant measurements with distributed sensors and combining them with the purpose of detecting and identifying situations of interest. Earlier in Section 3.5.1 it was stated that a description of real world situation can be subdivided into a number of factors or rather situation parameters. These situation parameters in turn can be generated by distributed sensor network nodes. However, when integrating the generated situation parameters into a single coherent situation description, one must consider contextual consistency of situation parameters. Each situation parameter, generated by a sensor node, is measured together with some other context (e.g. temporal - with sensor clock, spatial - usually hard-coded information provided by user, etc.). However, these context data form the contextual information for the situation parameters.

The distributed processing of situational information must be validated on-line so that only relevant, correct and consistent data are exchanged, especially when interactions are created on the basis of temporal and spatial needs. The concept of situation parameters is described in section 3.5.1, in the context of agents exchanging situational information. The situation parameters are equipped with contextual validity information. Validity information indicates for example where or when the specific parameters are valid (e.g. the spatial and temporal validity). Validity intervals of situation parameters can be very different. Some situation parameters have relatively short lifetime, this is called ephemeral data (e.g. instantaneous position-location information about a mobile phenomena), in which case older data is quickly obviated by newer data. By contrast, persistent data (e.g.

computed track of a mobile phenomena) has a longer useful lifetime and is not obviated by newer data. The contextual validity information makes it possible to validate the situation parameters against the temporal and spatial constraints defined in the respective subscriptions for situational information. The relevance of defining the optimal values for temporal and spatial constraints and their application is analysed by Kaugerand and Ehala in Publication II. The article describes the results of experiments where online checks are carried out for both temporal and spatial correctness of situational information exchange on both data producer and consumer side. In the spatial domain, the definition of spatial validity is usually straightforward. Spatial validity is defined in a form of an area, either a circle or a polygon. This allows data consumer to subscribe to information from a specific spatial area. Data for this area can be produced by several different data producers in case their field of view covers the required area. The data producers for which the required area is out of their sensing range are filtered out. The temporal constraints in turn, are in scientific literature usually defined simply as a maximum age for certain data packets. When this age expires, the data packet can be discarded. This thesis suggests that the temporal constraints should be defined as intervals, which define both the maximum age, but also the minimum age. Both ends of the interval should be adjusted for each interaction partner. By defining the temporal and spatial constraints respectively as intervals and areas enables time and location selective communication.

4.3.1 Temporal validity of situation parameters

This section discusses the importance of temporal validity of input data for INDP node and how the value of validity of sensor readings affects the selection of temporally compatible inputs for the INDP in WSN. The necessity of checking and ensuring the sensor data validity has been discussed in paper [110] and further elaborated in Publication II where it has been explained how every sensor reading has temporal and spatial validity intervals associated with it. These intervals may depend on several aspects, for example, the spatial validity area depends on the location of the sensor nodes and on the properties of the phenomenon being observed, while the temporal validity interval depends both on the properties of the environment where the node is located and on the nature of the phenomenon being observed. The sensor node augments its output data, with the validity intervals, and verifies that validity interval satisfies consumer constraints on validity (defined in subscription) before transmitting them. The INDP node in turn verifies that the validity intervals of the sensor readings upon their arrival do match with the consumer constraints set on incoming data. The output of the INDP process is in turn again accompanied with the metadata which also contains respective validity intervals checked by the users of fused data.

It is difficult to determine the precise arrival time of data to the INDP node in an ad hoc WSN in advance. For this reason, consumer defined constraints are used to set an upper and lower bounds on the transport and usability time of the sensor readings. When this consumer defined temporal validity interval expires before the sensor readings arrive to the INDP node, the readings are discarded. Similarly, the sensor reading should not reach the consumer before its consumer defined validity interval starts. By setting a lower bound to sensor readings validity, the consumer states that it is not able or ready to use the readings this early. The temporal constraints employed by the consumer node for its computation of situational information are not necessarily related to the validity intervals of arriving data set by producer. The constraints can be stricter or more relaxed depending on the SA application and context (as decided on-line by the consumer node or at design time by the system designer). If the producer defined validity of arrived data satisfies the temporal constraints, it is stored in the consumer node memory, where it remains available so that the processes defined for consumer can select the suitable inputs at the right time.

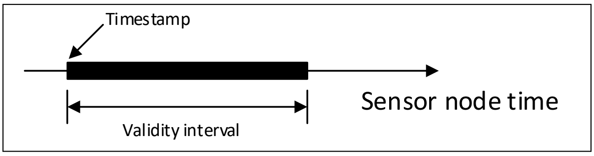


Figure 23 – Timestamp and validity interval

Figure 23 describes a timestamp and a producer defined validity interval for a sensor reading. The timestamp t_s indicates the time instant of the sensing process which produced the sensor reading. This is considered as the time instant when the sensor reading was acquired as it is not always feasible or even possible with cheap COTS technology to create technology which accurately detects the event. This is in more detail explained in 4.3.6. The validity interval $I_{validity}$ indicates the period of time during which the resulting sensor reading is valid. The validity interval for a single sensor reading can be expressed with Equation 2 as follows:

$$I_{validity} = [t_s, t_s + t_{valid}], \tag{2}$$

where t_s is timestamp of the sensor reading and t_{valid} is a length of validity interval on INDP node's time axis. In case the INDP node receives input data from different sources, all the data must satisfy consumer constraints at their arrival.

From the data producer point of view, there are following options for defining temporal data validity interval from time instant of detection:

1. until the next reading is expected to be computed, i.e. according to periodicity,
2. until the value of the observed parameter is not expected to change significantly,
3. until the value of the observed parameter crosses certain threshold, in other words, the data is valid "until changed". This could lead to introduction of a new parameter to subscription: a reporting threshold.

Another question that arises during defining the temporal validity interval is "can validity intervals of sensor readings produced by same sensor observing same phenomena in succession overlap i.e. can validity interval be longer than sensor "refresh rate"? This overlap of validity intervals is definitely possible and depends on application. However, during INDP of data from distributed nodes one must consider the rules of INDP carefully if this previously described overlap of two or more consecutive sensor readings overlap also with validity intervals of sensor readings from other distributed sources. The question is which specific sensor readings can be combined together. Thempster shafer theory and Bayesian methods could be used to select valid readings, provided that there is sufficiently large dataset available, however, as this is not possible in the the context of WSN, then the data from sensor nodes can not be fused or aggregated using these methods. This situation can be handled by defining simultaneity constraint as explained in Section 4.3.5.

From the consumer point of view, the data is valid within the time interval defined by data producer. The fact that data is not valid now, but in the past interval does not mean

that the data is not usable or relevant for SA applications, especially if it can be used to understand/model/predict the current and or the future situations. The temporal validity from consumer's perspective is defined as temporal validity constraint in subscriptions. If this constraint is violated, the data should be discarded (removed from memory buffer). This is why we need to define also validity from the consumer perspective. The consumer may have different rules from producer. Despite being no longer valid from producer's perspective, data could still be usable and relevant in terms of SA.
For consistent INDP, the validity intervals (defined by producer) of inputs must be overlapping as explained in Section 4.3.4. The inputs are not valid now, could well have been valid simultaneously at a certain time interval within temporal constraints of consumer node. Therefore, from the consumer point of view, current thesis defines the temporal validity constraints that define an interval where situation parameter is valid only from the perspective of SA and for inference of certain situation. This temporal validity interval depends on SA consumer requirements.

4.3.2 Spatial validity interval of situation parameters

Similarly to temporal validity, each sensor reading or situation parameter must also have a spatial validity. The spatial validity of the situation parameter defines an area where the parameter is valid. The spatial validity area of a situational parameter may be related to the producer node's location in case the node is mobile or to the specific phenomena observed by the producer node in case parameter is a result of observing the phenomena. For example in case the producer node is a mobile agent, then the spatial validity check before data delivery becomes absolutely necessary. The consumer may have defined spatial constraints specifically for a certain location, but the producer might not yet arrived to this area or might already have moved away from the area. The consumer's spatial constraints determine if the situation parameter is usable for the consumer according to the spatial context. In case the fusion node receives input data from different sources, all the data must be valid at their arrival.

4.3.3 Relative contextual consistency

Relative consistency means that situation parameters used for situation inference of a specific situation must be consistent with each other in required context. For example, when observing a moving vehicle, the heterogeneous distributed sensors in a sensor network might acquire data and produce streams of different parameters each describing either vehicle passing event, its speed, direction or its class. In temporal context all parameters have timestamps and temporal validity. When combining these parameters together in order to understand what is the situation of this vehicle the time and location stamps for specific situation parameters must be consistent.
Lets take for example a situation where:
Sensor $Sensor_1$ reports a situation parameter $S_1 = S_p, S_a, S_t$, where S_p contains a datatype $dt_{VehicleSpeed}$ with a value $speed_1$, S_a denotes location L_1 and S_t denotes time instant T_1 and sensor $Sensor_2$ reports a situation parameter $S_2 = S_p, S_a, S_t$, where S_p contains a datatype $dt_{VehicleClass}$ $class_1$, S_a denotes location L_2 and S_t denotes time instant T_2.

The question now is, how to integrate the situation parameters S_1 and S_2 together to assess the situation of a vehicle with class $class_1$ passing location L with speed $Speed_1$ at time instant T, where L is a combination of L_1 and L_2 and T is a combination of T_1 and T_2. The answer to this question lies in how strict constraints the consumer of this situational information has established. It can be said that both situational parameters are relatively consistent if they describe the same situation. This means that they must satisfy both user established validity and consistency constraints.

It is often a case that an execution of a predefined algorithm can only start when all or at least a certain number of inputs are available. This means that the processing node must wait for inputs from its data sources. If one considers data sources, which are distributed nodes in ad hoc mesh network, this means that before inputs can be used, they must firstly be checked against contextual requirements and secondly against the consistency. The multiple situation parameters used as inputs must have overlapping validity, as explained in Section 4.3.4, or within a certain simultaneity constraint (defined as an interval), as explained in Section 4.3.5. For example, a temporal validity requirement can state that inputs must not be older that N milliseconds and simultaneity constraint can state that the time difference between data from different sources must be within a predefined interval, as explained in next sections.

4.3.4 In-network data processing requires overlapping validity intervals

Validity intervals of individual data elements in streams can be used for grouping data and selecting data elements with overlapping validity intervals. Figure 24 depicts four sensor readings, their timestamps and validity intervals.

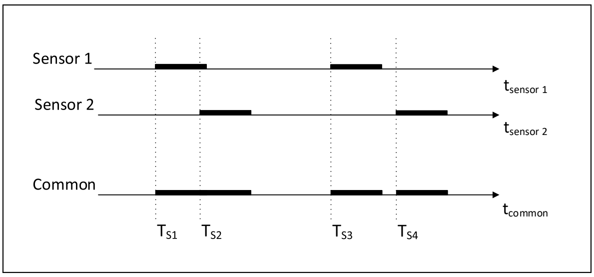


Figure 24 – Overlap of validity intervals.

The t_{S1}, t_{S2}, t_{S3} and t_{S4} denote the timestamps of sensor readings projected on to a common time axis (e.g. a fusion or aggregation node time axis), and black rectangles indicate the respective validity intervals $I_{valid(t_{Sn})}$. It can be observed that sensor reading with timestamp t_{S2} falls within the validity interval of another sensor reading with timestamp t_{S1}. There is a period of time during which both sensor readings are valid and both can be used as inputs for a fusion process. This period of simultaneous validity or an overlapping validity interval can be expressed as $I_{valid(t_{S1},t_{S2})} = I_{valid(t_{S1})} \cap I_{valid(t_{S2})}$. The opposite case can be observed with timestamps of t_{S3} and t_{S4}, where the validity of sensor reading with timestamp t_{S4} does not overlap with the validity of sensor reading with timestamp t_{S3}, thus they should not be used together for detection or identification of a more abstract situation.

In large-scale sensor networks, with INDP capability there can be several distributed sensors that produce data streams for example for a fusion node input. The INDP will provide correct results only when the INDP process takes as an input sensor readings that describe the same situation, which are also valid at the same time, that is, for which there exists a common overlapping validity interval. However, one can also consider a situation where the INDP process, after its execution, has access to data which were valid during their arrival at INDP node, but which validity has expired by the time moment when the actual selection of suitable input data takes place (INDP process may have its own buffers to store input data, which validity may also expire while it is kept). In this case, the validity intervals do not lose their importance. What is important, is that the validity intervals of potential input data from different sources have an overlap. The resulting INDP output data have their own validity interval assigned before the outputted data is transmitted to its corresponding consumer. In general, in case of fusion, the new output data has the validity interval, which is the overlap (intersection) of the INDP input data validity intervals and in case of aggregation (e.g. an average), the output data has validity interval which is union of the input data validity intervals. The validity interval of INDP output data is subjected to same constraints as described previously. Besides overlapping validity the INDP requires also defining the length of simultaneity interval.

4.3.5 Simultaneity interval

The simultaneity interval serves a dual role – it enables to convey and evaluate the actually achieved synchronicity in a network and, if necessary, to compare it with the required synchronicity; and it provides a design parameter for assigning validity intervals for individual sensor readings in order to achieve feasible fusion or aggregation of those readings. In general, the simultaneity interval specifies a time window of tolerance, within which a set of events (e.g. sensor readings or situation parameters) can be considered 'simultaneous' and can be used for INDP processes. It is a period of time that begins with the occurrence of the first of a group of events and ends with the occurrence of the last event of the same group [96]. A simultaneity interval for two sensor readings with timestamps t_{s1} and t_{s2} is expressed as $I_{sim(t_{s1},t_{s2})} = |t_{s2} - t_{s1}|$. For example, if an INDP process receives four sensor readings (events) as inputs with the timestamps of $d_1 = 980ms$, $d_2 = 1010ms$, $d_3 = 875ms$ and $d_4 = 1045ms$, the simultaneity interval for these readings would be $I_{sim(d_1,d_2,d_3,d_4)} = 170ms$. However, in order to consider these readings simultaneous one would have to have defined a specific requirement for a simultaneity constraint. For these inputs the simultaneity constraint would have to be at least larger than 170. If one would have chosen a simultaneity constraint of 150ms, then one of the events either d_1 or d_4 would have had to be considered outside of the simultaneity interval, depending on which event is considered as a first or with higher priority or more relevant for specific situation. The relevance of situation parameters is discussed in 3.5.2. This thesis defines the simultaneity constraint as C_{sim}. The simultaneity constraint is a design goal or rather a requirement for simultaneity of sensor readings (more precisely the situation parameters of observed situations that the sensor readings represent). For example, then looking at the two sensor readings with timestamps t_{S1} and t_{tS2} depicted in figure 24, the correct INDP of these readings requires (in addition to overlapping validity intervals) that the simultaneity interval of the given group of sensor readings satisfies: $C_{sim} \geqslant I_{sim}$. This requirement is independent of the group size, all sensor readings grouped into single I_{sim} according to their timestamps must satisfy C_{sim} in order to be interpreted as simultaneous. In practice, the choice of suitable simultaneity constraint involves several consideration e.g. the application itself, asynchronous sensor reporting intervals and the precision of computed delays of sensor readings. The computation of delay of sensor readings is discussed in Section 4.3.6. The sample application used in this article is the detection of moving vehicles. The choice of simultaneity constraint will influence the precision of the position estimate of the detected vehicle. For example, if Csim=400 ms is chosen, the position of the vehicle is interpreted to be within the area it can cover in 400 ms (given that the speed of the vehicle is known).

4.3.6 End-to-end delays of situation parameters

In order to process the sensor readings in a time-sensitive manner and to align them on a common reference time, the processing node must be able to compute the delays of the arriving data that forms its inputs with a certain required precision. There are two aspects to consider here, first, how the timestamp of the observed situation is computed by the sensor data acquisition process and, secondly, how the delays are computed and projected to the INDP node local time axis.

The timestamp computation problem may not be trivial in the case of low-cost sensor nodes. In ad hoc WSN, it is not feasible that the sensor reading are transmitted very frequently. In most cases, multiple sensor samples, called a frame, are either aggregated (averaged, summed, etc.) or processed into a single sensor reading for the entire frame period. Due to limited computational resources in low-cost sensor nodes, it may not be always feasible to compute the exact time instant of the actual situation from the sampled frame, so a start of the frame is considered as the process activation instant t_s and is used as a creation time instant (timestamp on a sensor node time axis) for sensor readings. Although this does make the modelling and analysis easier, this approach may result in considerable, but bounded error $e \leq t_a$ (t_a is a single sensor process execution period) in sensor reading delay computation. This error must be taken into account when computing the accumulated delay of sensor readings as this affects the comparison of the validity intervals of several readings from different sensors, when projected on to the receiving INDP node time axis and interpreting the INDP results. When the sensor processes support the computation of the exact time instant of the observed situation (for which the sensor reading has been computed), the resulting timestamp for the sensor reading should be updated accordingly.

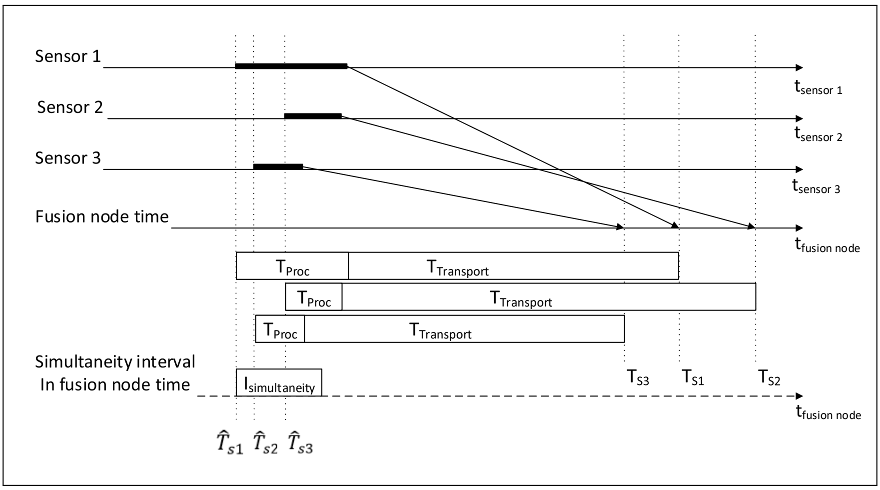


Figure 25 – Temporal alignment and the simultaneity interval.

For the latter problem, a generalised example of distributed sensor communication is presented in Figure 25. Black rectangles on the sensor timelines (t_{S1}, t_{S2} and t_{S3}) represent the duration of signal processing on each sensor node, and arrows indicate the transport times. The packets reach the fusion node at times T_{S1}, T_{S2}, T_{S3} (on timeline of t_{FN}). The fusion node estimates the total transport and processing times of each packet incoming from $S1 - S3$, denoted by rectangles T_{proc} and $T_{transport}$. The event detection times are then aligned to the estimated time instants $\hat{T_{S1}}$, $\hat{T_{S2}}$, $\hat{T_{S3}}$ and compared against the simultaneity interval $I_{simultaneity}$ (on timeline $\hat{T_{FN}}$). Figure 25 illustrates that the processing time and packet transport time for each sensor node may be different, which results in the packets arriving out of order and too far apart to be included in the same simultaneity interval. Therefore, without proper temporal validation it is not guaranteed that the sensor samples characterise the same event. Processing delays originating from sensor platform specifics, clock jitters and drifts, ad-hoc WSN transmission scheduling, limited bandwidth and packet collisions within the network — all can unexpectedly disrupt smooth WSN communication. Proper network management is required to ensure real-time operation of the system as a whole.

Classical methods align sensor data to a common time reference with the help of time synchronisation algorithms [124, 34]. However, applying classical methods, where all sensor readings are collected via a sink node (gateway) to a central cloud server outside of distributed WSN, may lead to significant communication overhead and is not optimal in ad hoc networks. Other methods to align data without global WSN synchronisation include, for example, temporal alignment by utilising causal dependencies [26], where authors use vector clocks. We consider the synchronisation based on vector clocks inefficient because of two specific reasons. First, the size of each timestamp (a vector of timestamps) is proportional to the number of nodes in the network, and secondly, using vector clocks requires increased communication between the sensor nodes in order to establish the causal relations between the sensor readings. Instead of traditional synchronisation methods in WSNs, which can lead to significant communication overhead [34], this thesis takes advantage of existing packet-level delay computation service [88] (for example, implemented in the TinyOS operating systems), which allows to mitigate considerably the timing indeterminism for transmission-related delays (send time, access time and receive time) for a single hop. Its main advantage over other synchronisation methods is its lightweight nature. Each node computes the accumulated delay for the data and passes this temporal information along with the transmitted data. The packet-level delay computation method supported by TinyOS operating system allows the communication stack to automatically convert the sending node local time to the receiving node local time by appropriately modifying the time value within the packet after its transmission is started. The sending node converts the time value within the packet to a delay d_{comp} spent up to that moment since the creation of data and the receiving node in turn can use d_{comp} to compute the data creation time moment on its own local time domain by subtracting its value from the time moment of data arrival. This method does not provide synchronized network time, but provides a submillisecond accuracy for a single hop. Combining this method with time-selective strategy makes it possible to obtain correct results when data are fused from sensors, which readings are produced asynchronously. In other words, neither the clocks nor the actual sampling of the data by distributed sensor nodes are synchronized in any way. In case of multi-hop situation, each forwarding sensor node in the network estimates the time interval d_{comp} between receiving and transmitting data and adds it incrementally to the previous delay (age) of sensor data before forwarding it to the next hop.

4.4 Modelling network nodes and data exchange

This section defines and explains some important concepts for modelling network nodes and data exchange in an ad hoc network for Smart Environments. Concepts such as architecture of the network nodes, modelling processes inside the nodes, how validity intervals and simultaneity interval for stream elements are used are described. The section also describes time-selective data communication strategy for WSN introduced in Publication I and gives a detailed overview of the algorithms for the temporal alignment of data and selection of compatible elements for INDP processes.

The general architecture of a sensor node used during the experiments described in thesis is described on Figure 26.

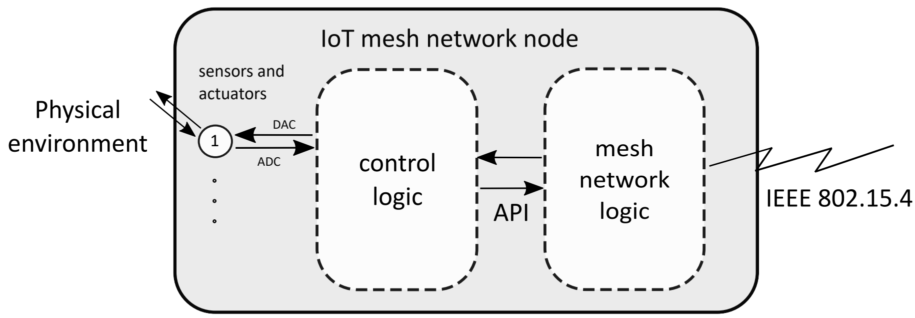


Figure 26 – IoT mesh node architecture.

A network node is modelled as a process which outputs messages containing situation parameters derived from sensor readings. At an abstract level (network level), the computation of situational information by an individual network node can be described as a single process. Although network node process can contain several sub processes including processes for production of different sensor readings, computation of situation parameters, including aggregation and fusion processes and the mediator process.

One example possible set of processes for computing situation parameters from observing a physical phenomena in a producer node can be seen on Figure 27. On the figure five processes are depicted. This set of processes can vary in details depending on the application. A short description of each process follows: 1) the process of raw data sampling takes care of sampling the physical signal (e.g. different sampling and frame rates), 2) the process of raw data processing takes care of signal conditioning, normalisation and application based processing (e.g. fourier transform), 3) the process of situation parameter estimation adds processed data into suitable data structures and takes care of aggregation, interpolation or extrapolation as defined by subscription and adds contextual information (e.g. spatial and temporal information, confidence level, etc.), 4) the process of storage for situation parameters stores situation parameters in a ring-buffer like memory structure and 5) the process of delivery service (a part of a mediator process) takes care of delivery of the messages depending on the network availability and status.

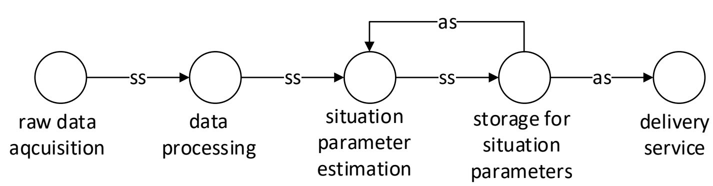


Figure 27 – An example set of processes for production of situational information in a producer node.

The producer node may contain several such chains depending on number of available sensors or other sources for situational information. The actual data delivery for situational information in any network may not always be optimal as desired by situational needs by SA consumer. For example the bandwidth in an ad hoc WSN may not support large amounts of data transmissions, nor frequent reporting of change of situation by sensors. In case the rate of situational information production is higher than the ability to transmit the situation parameters should be aggregated before transmission or abstracted to higher level situation parameter. This is done by the process of parameter estimation process, the latter process may also be capable of interpolating between existing parameters or extrapolating to estimate future parameter values, depending on consumer requirements (defined in subscriptions).

The API-s or rather channels in the model between first four processes (data acquisition, data processing, situation estimation and local storage) are semi-synchronous (semi-synchronous means that the consumer process is started immediately after the producer-process completes its execution). This is self-evident as these steps are usually event driven. Especially, as each process provides input for the next one. Although the data acquisition itself is usually periodical, the monitored physical phenomena and relevant surrounding environment of interest dictates the production of situation parameters. For example if there are no changes in physical phenomena under observation (certain thresholds are not crossed), then there is no reason to recompute the situation parameter (or re-estimate the current situation). In this sense the computation of situation parameters is driven by monitored physical environment. The produced situation parameters are periodically added and stored in a local data storage. This storage is here modelled as an individual process as it plays an important part for entire producer node. It stores a limited amount of history of previously computed situation parameters. The data storage may also be used when new situation parameters are computed. This is depicted on Figure 27 as an asynchronous channel from process of storage for situation parameters to a process for situation estimation. Examples of this feedback can be either an aggregation or extrapolation request. Essentially this local data storage is modelled as a circular buffer. It stores a number of computed situation parameters and overwrites the old parameters when the buffer is filled. Ultimately its size is limited by local physical memory availability. However, the number of parameters stored in the circular buffer data structure is dictated by consumer requirements. Another important function of the local storage for situation parameters is that the delivery service is allowed to access any elements in it, either by index or by contextual constraints. The channel between already produced and stored situation parameters and delivery service is also asynchronous. The advantage of this type of architecture is that the delivery service, being a part of a ProWare mediator, can now apply consumer-defined constraints when requesting situation parameters from the local storage. This means that, as the delivery service is executed only as required by consumer and as the network policy allows, it is possible for the delivery service to always choose only these situation parameters for the delivery which satisfy the contextual constraints of the consumer. Another bonus is that this leads naturally to a case where it is now possible for the mediator to deliver situation parameters on different levels of abstraction, exactly as defined by the consumer. For example by aggregation over a chosen time interval (e.g. by averaging over a certain number of produced situation parameters). However, making data production and delivery processes asynchronous (i.e. decoupled) means that there is a time variable delay already at the producer node level that must be taken into account. It is called a non-transport delay [97] and its modelling is explained in Section 4.4.2.

A formal model for all sub processes is described by using Q model [96]. The execution timeset of a periodic execution of a processes (e.g. for computing situation parameters) is modelled by the Equation 3.

$$T(p) = \{t : t_n = t_0 + n \cdot t_a\} \quad (3)$$

In Equation 3 the variable $t_0 = 0$, $n \in \mathbb{N}$, t_a is the execution interval (e.g. situation parameter computation interval) of a process and $T(p)$ is a timeset of a process executions (e.g. situation parameters estimation process executions). The processes between distributed sensor nodes are considered asynchronous and have each their own timeset and time counting mechanism. For modelling purposes, each activation/execution instant t_n of a process also determines the timestamp (by age) of the data produced by this process. When the produced data is transmitted by the node, its timestamp is updated to reflect the delay between the process activation instant and the actual transmission moment. The practical process of delay computation is described in Section 4.3.6. If computationally feasible, the timestamp is also updated to reflect the estimation of a time-moment of the physical-world situation that is captured by the sensor process. Computing the precise time instant of the situation might not always be trivial due to limited resources of low-cost WSN nodes.

4.4.1 Modelling time-selective data exchange

This thesis uses Q model formalism to describe the mediated interactions between the network nodes. It gives a necessary abstraction layer for describing data delivery mechanism between two nodes. For example, the asynchronous data delivery between the producer and consumer nodes is described by using a channel concept. Consumer node is given access to the data produced by producer node at time instant t. The channel buffers data in a queue organised as a ring buffer and the consumer gets access to a number of sensor readings specified by contextual meta information. This includes intervals for different contextual constraints, such as time, location, confidence etc. By addressing the channel by contextual meta information the data usage between processes is selective, the consumer gets access to data that are bounded within defined constraints. This selective access to data is one of the properties of a mediated interaction. The data stream resulting from data produced by one of the processes is mediated by the channel function and transferred to another process. Formally, for example, when only temporal interval is described as $[\mu, \nu]$, the channel function is expressed with Equation 4.

$$K\left(\sigma_{sf}, t\right) \subset T\left(p_s\right), t \subseteq T\left(p_f\right) \quad (4)$$

The variable σ_{sf} in Equation 4 denotes the channel between producer node s and consumer node f. Variable t denotes the time instant when the consumer requests the available data described by the temporal validity interval $[\mu, \nu]$ (from the consumer's perspective), where Variable μ indicates the oldest allowable age and ν most recent age for valid data. The temporal validity interval in Equation 4, is defined from the consumer's perspective and indicates the number of stream elements conveyed by channel, or rather temporal span of the accessible elements received by consumer process. This time interval is defined by Equation 5.

$$K\left(\sigma_{sf}, t\right) = [\mu, \nu] \quad (5)$$

The time interval $[\mu, \nu]$ in Equation 5 is defined on the time axis of the consumer process. It is easy to get an impression that ν should usually be set to 0 as most recent data is often required. In reality, cases that are relevant for SA applications running in ad hoc networks include both data with different periodicity and with different signal processing intervals and data from distributed sensors with variable delivery delays. In order to cope with different and variable delays of disparate producer nodes, the consumer should be able to dynamically change values of ν and μ in order to fuse and aggregate only contextually matching sensor readings.

An overview of the nodes, processes and related channels is provided on Figure 28. Each node may run several processes, where each process may have its own execution timeset and execution period. Each producer process execution can result in a sensor reading which is conveyed to the consumer process via the channel function. Each consumer process (e.g. an INDP node) establishes a separate channel for each producer (e.g. a sensor node) process. Each channel may have a different interval $[\mu, \nu]$ of the accessible elements and the consumer process has access to the transmitted producers sensor readings according to the channel function.

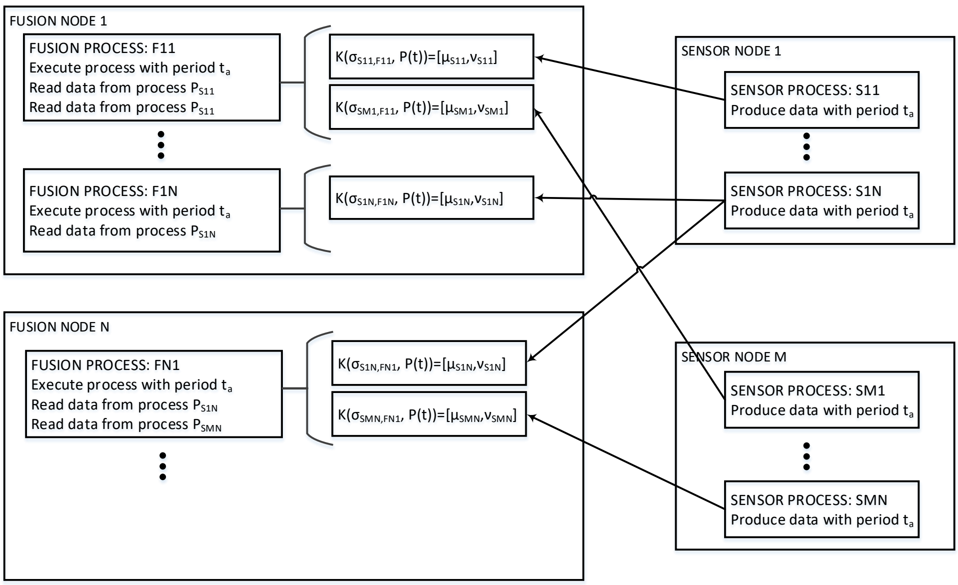


Figure 28 – Nodes and processes.

For example, for the consumer process p_f during its single execution, the interval $[\mu, \nu]$ represents the requirement for the accessible stream elements from producer process p_s. The variables μ and ν are defined by the consumer process and represent respectively the earliest (oldest) and latest (most recent) instants of the producer process output data. Usually $\nu = 0$, is chosen as the most recent possible data is required by the consumer process.

The actual age of the most recent possible data depends on several details. During each execution, the consumer process can read data from several channels, i.e. it can have access to several streams, each from different process. In the asynchronous case, the actual time instant for the most recent possible element for specific channel for the interval $[\mu, \nu]$ is specified by the Equation 6.

$$t = \max_{t_s} \left\{ t_s < t_f + \eta\left(\sigma_{sf}, t_f\right) - \zeta\left(p_s, t_s\right) - \xi\left(\sigma_{sf}\right) \right\}, \tag{6}$$

In Equation 6 the variable $t_s \subseteq T\left(p_s\right)$ is producer process execution time instant, the variable $t_f \subseteq T\left(p_f\right)$ is consumer process execution instant and where $\eta\left(\sigma_{sf}, t_f\right)$ is the length of time interval during which the consumer process receives the data. The variable

$\zeta\left(p_s, t_s\right)$ computes the execution time of the producer process and the channel delay is represented by variable $\xi\left(\sigma_{sf}\right)$. For simplicity the propagation time of a radio packet is considered zero. The formula for finding the most recent possible element from the specific stream of producer readings at specific consumer process execution is explained on Figure 29.

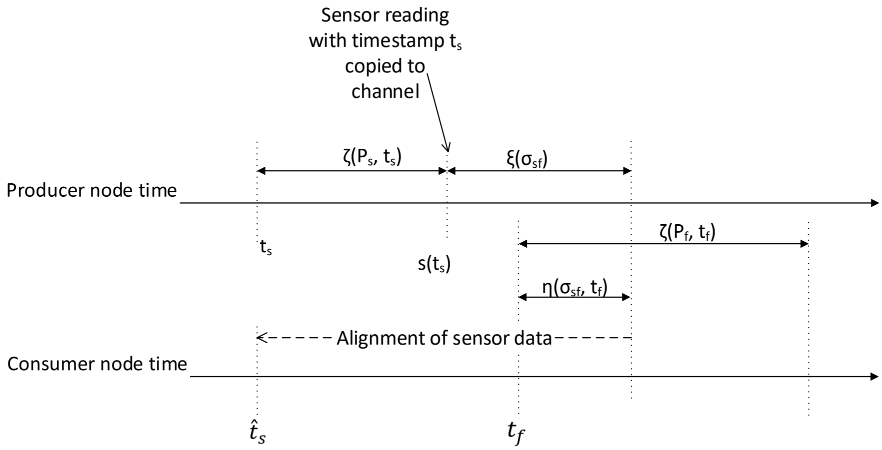


Figure 29 – Computation of most recent available element for the consuming node.

Figure 29 depicts two different time lines, one for the producer process and another for the consumer process. These time lines can be entirely independent of each other as processes are considered asynchronous (this thesis considers autonomous agents). The start time instant of the consumer process execution is denoted on consumer node time line with t_f. After an estimated interval $\eta\left(\sigma_{sf}, t_f\right)$ the consumer process accesses the channel and receives the stream elements. In this case it is assumed for simplicity that $\nu = 0$, current figure does not explain how the rest of the elements are stored in the interval $[\mu, \nu]$ or how these are conveyed to the consumer process. When the consumer process receives the sensor data, it aligns the estimated timestamp of producer's sensor reading $\hat{t}_f$ on consumer node as explained in subsection 4.3.6. In case the consumer process receives data from several streams, all channels are accessed sequentially, until either consumer process has all the data it needs as inputs or a time limit for accessing the channel expires.

In the original Q-model the channel delay $\xi\left(\sigma_{sf}\right)$ is included either in the producer process execution interval $\eta\left(\sigma_{sf}, t_s\right)$ or in the consumer process channel access interval $\zeta\left(\sigma_{sf}, t_f\right)$. Current work considers it necessary to point out the channel delay explicitly. This is due to the reason that in multi-hop networks the importance of the channel delay becomes prominent.

4.4.2 Modelling non-transport delay

In SoS type of Smart Environments it can be, that the processes of situation parameter computation (situation estimation) and process of data delivery in the network nodes are not synchronised. This approach is necessary in large scale and dense WSN networks as otherwise situation parameters computation together with radio transmission driven by physical events would quickly lead to collisions and lost packets. One of the methods, also used in this thesis, to avoid this is to apply hierarchical clustering with time-division media access (TDMA) channel access method as explained in 2.3. In case the ad hoc mesh WSN delivery service for situational information would have to wait (in an organised manner) for its turn, before the actual delivery can take place. The delay caused by buffering either in data producer node or in network nodes during data delivery is called non-transport delay.

In order to understand the non-transport delay, there is a need to abstract away all the other delays in data transport from one process to another. After this, the actual non-transport delay due to periodic execution of asynchronous processes at any consumer process execution instant can be computed by the Equation 7.

$$\varphi_n(t) = t_{consumer} - t_{producer} \tag{7}$$

In Equation 7 the variable $t_{consumer}$ denotes execution instant of consumer process and $t_{producer}$ an execution instant of producer process. The $\varphi_n(t)$ is the non-transport delay. When dealing with asynchronous processes, as depicted on figure 29 the time series graph of resulting non-transport delay takes a sawtooth-like shape. When consumer receives such data streams of data from several asynchronous producers, especially when a multi-hop route is defined over several network nodes, then the resulting arrival of data may seem truly random. The need to correctly compute end-to-end delays and to solve the data alignment problem becomes essential.

4.4.3 Distributed computation in Smart Environments is performed on delayed data

The distributed network nodes can perform current situation assessment only on available data. If one distributed node requires data from another distributed node, a subscription for the required data must be posted. The mediator delivers the subscription to the node who potentially has the required data and in turn the data is delivered back to the data subscriber. The challenge in distributed mesh networks is that the time interval for both subscribing to the data and the delivery is significant in comparison to addressing the local memory. This delay can be reduced to some extent by subscribing for a stream of data, this removes the requesting delay for each individual sensor reading. However, in wireless sensor networks based on 802.15.4 protocol the delays for input data for INDP algorithms can be very unpredictable, even if the time counting in the distributed nodes could be globally synchronised. Another problem is mesh architecture and wide geographical distribution of the nodes, when receiving situational information from remote nodes, the consumer must take possible multi-hop transport delays into consideration. Hence, due to delays occurring in distributed wireless network nodes the INDP (and mist computation) is almost always performed on historical data that is either produced locally or acquired from other distributed nodes and stored in local buffers.

The subscription-based data exchange model described in [107] allows to share the memory requirements between the data consumers and producers. The consumer does not have to acquire and store all sensor readings that are made available by the producers, instead the consumer can subscribe to data with specific temporal constraints and request only for the data that suits its temporal needs. The producers themselves buffer their own data until it satisfies the constraints. This service is provided by the intelligent data mediator (ProWare middleware).

4.4.4 Data buffering during in-network data processing

Embedded devices are typically equipped with very limited memory. For example a typical embedded system such as an 8-bit Atmega128 has 128kB of program memory, a state of the art ARM 32-bit microcontroller could have a program memory size of up to 1024 kB but to minimise the price of the hardware, still in most cases the program memory size is below 512 kB and the RAM size is 32 kB or less. It is clear that network nodes with so little memory are not capable of storing long histories of streaming data from multiple distributed sources. The concept of mediated interactions allows at an abstract level to think of a channel connecting the producer and consumer nodes as a smart mediator agent, a smart channel, which takes care of the services like discovery of data producers, agreements on data exchange, the delivery of data, providing constraints for data delivery and on-line validation of data against the provided constraints while delivering the data. However, in most simple terms, the channel in Q-model can be described as a ring buffer for a stream of data. It provides the data buffering, needed for asynchronous communication. However, as truly asynchronous processes require infinite memory for keeping the messages while the consumer is either not ready or does not yet need the data, the practical question where the data is actually stored remains. Considering the limited resources of WSN nodes the question about storing the memory of the produced streams of data is a very relevant one. It is an interesting side-effect of utilisation of the concept of time-selectivity inherent to Q-model [96] that enables an opportunity to share the memory load between the consumer and producer nodes. The concept of time-selective communication allows the consumer to request the data with specific time interval $[\mu, \nu]$ as explained in Section 4.4.1. By not setting the temporal validity limit for freshness (ν) to 0 for some producers, but having a delay before delivery for the producers with lower end-to-end delays, the consumer is relieved from storing all data before it can use it. Hence, in principle the consumer node could set different temporal constraint to its data producers and level out the delays.

The idea is that usually in case of INDP applications, there are many more producers than consumers and it can be useful to store some of the produced data at the producers' side. Each producer stores a certain amount, defined by the variable ν. In other words the variable ν defines for the producer process the most recent element to be transmitted. If the variable ν is 0, the producer always transmits its most recent sensor readings. Setting a positive value N for ν, effectively orders the producer to delay its most recent data by a time interval N. The result is that consumers only store data they have produced by themselves and data they actually need from their interaction partners. The oldest data to that can arrive at consumer side is defined by variable μ. The value of this variable should be at least larger than end-to-end delay as otherwise the message is dropped on the way because it loses its temporal validity.

One must also consider the available memory in low-cost sensor nodes when defining the validity interval $[\mu, \nu]$ in the subscriptions. If the interval is too short, the data outside the validity interval may be discarded and individual data items would not have overlapping validity intervals, on the other hand if too large validity interval is chosen, the low-cost nodes might not have enough physical memory and the memory buffers for storing situational information might overflow. Another aspect that must be considered here is the variable nature of end-to-end delays in ad hoc DIL WSN. In case the variability in end-to-end delays is too volatile, the above mentioned method may not be effective. This remains a topic of research outside this thesis.

4.4.5 Alignment and selection of compatible elements from streams

The basic idea of the alignment and selection algorithm is to group the available readings transmitted by disparate sensor nodes according to their contextual characteristics (such as temporal validity intervals $I_{validity}$ and/or simultaneity interval I_{sim} and spatial validity polygons) and to use only these selected groups as inputs for INDP processes. The need for such an approach is driven by the problem which arises when several distributed and autonomous ad hoc WSN nodes are used for simultaneous observation to detect situations in real time. Due to the different delays (both processing and networking), the sensor readings used as inputs for INDP algorithms, may not characterise the same real-world situation if naively used in the same order in which they arrived to the network node carrying out the distributed computation.

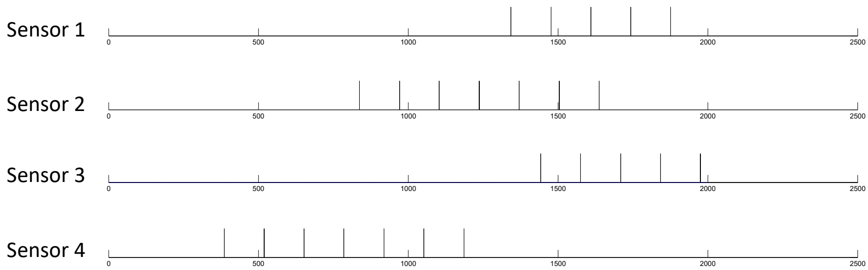


Figure 30 – An example of received and stored data with different delays.

One of the real-world cases, describing differing delays is depicted on Figure 30. On the figure the received stream elements have been projected onto the INDP node's time axis. One can observe that the stream elements on the bottom axis do not overlap with the elements from two of the streams above. However, even in this case, it can be the case that the temporal validity constraints set by consumer node of the stream elements overlap and one can also define a sufficiently relaxed simultaneity constraint, so that a set of four stream elements can be selected and presented to an INDP algorithm as inputs. Figure 31 shows three steps of the alignment and selection process of compatible data from sensor streams. The image a) represents the received stream elements by an INDP node. The arrival order of the stream elements from different sensor nodes is not known in advance as sensor nodes run asynchronously. The incoming stream elements are received by mediator component at the INDP node. The mediator performs the validity check and projects the stream elements to the node's local time axis. The black filled squares in images b) and c) depict stream elements which have been assessed as temporally compatible. When the INDP process executes and requests its inputs, the data alignment and selection algorithm aligns the stream elements from different sensors in the INDP node time axis as depicted on image b).

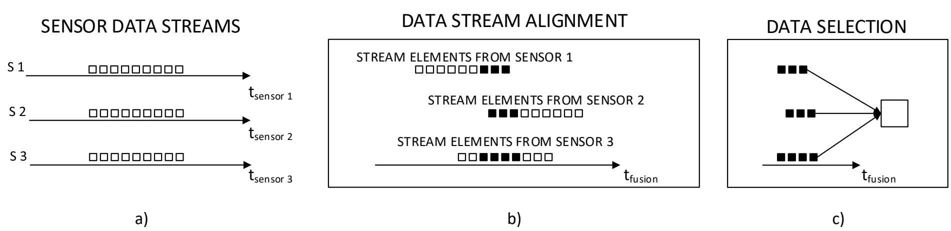


Figure 31 – Three steps of alignment and selection process of compatible data from sensor streams.

The selection of temporally compatible elements from streams for data fusion is depicted on the image c). The process of selection of temporally compatible elements is described by Algorithm 1 in Chapter 4.4.6. The algorithm accepts m number of stream segments as inputs. The minimum number of elements in a segment is 1. The length of a segment depends both on whether the specific data provider is currently providing any data and if so, then with what rate and on INDP algorithm's execution period. Basically if chosen so, and if practical computational power and temporal limitations allow, the algorithm could execute after each single sensor reading arrives from distributed sources. Practically though, it can be more feasible to wait for a number of inputs and then execute the selection algorithm. Each stream segment available for the INDP node S_m contains n number of elements $element_n \subset s_m$. As the INDP node specifies a separate channel function $K(\sigma_{sf}, t) = [\mu, \nu]$ for each communication partner, the requirements for each stream may be different (the exact parameters for each channel function are specified in the data subscriptions made by the consumer node's mediator component (ProWare middleware)).

4.4.6 Alignment and selection algorithm

This section provides a detailed description of the alignment and selection algorithm used in experiments described in Section 5.3. When searching contextually compatible elements across several streams, one stream must be taken as a reference. In order to do that, the incoming streams are sorted by their priority. The criteria for priority can either be relevance of the situation parameters, confidence or fidelity level of the stream elements or also stream update rate or currently available number of elements in the available stream segment. For example, in Publication I, for the sake of simplicity, the stream segment with the least number of elements was chosen as the reference stream. This was feasible as experiments carried out in this work were conducted only with homogeneous input. However, later work conducted by the author referenced came to conclusion to use relevance of situation parameters for deriving the priority of a stream. Relevance of situation parameters is previously explained in Section 3.5.2. The data streams are ordered by their relevance to the situations which are currently relevant for the user. This relevance is in turn considered when assigning priority to data streams. According to our strategy the stream with the highest priority is taken as a starting point and is named $S1$. The idea is to process the elements of $S1$ one by one. For each element $element_{S1} \subset S1$, the alignment algorithm finds temporally the closest element $element_{S2}/subsetS2$ from the next stream $S2$. Closeness is in this case is defined temporally as the time interval between the timestamps of two stream elements. After finding the closest element to $element_{S1}$, a new time instant tw (that is related to given elements) is computed. tw is a weighted average of the timestamps of the identified closest stream elements.

The use of the weights for timestamps is motivated by the desire to take into account the confidence level of the computed delay. For example, stream elements which transport include more hops, resulting in lower precision for computed delays may have lower weights. The obtained tw is then used to find the temporally closest element from the next stream. The process repeats until data elements from all streams have been processed. Each time a new closest element from the next stream is found, a new tw of timestamps is computed from all previously identified elements. This way, the algorithm finds for each $element_{S1} \subset S1$ a set of temporally closest elements across all streams.

In Algorithm 1, this set is denoted as D. The obtained sets of temporally closest elements are then inserted into an ordered array A_{sim}, which is ordered by the simultaneity intervals I_{sim} of the sets in D. The sets D, which simultaneity interval I_{sim} values exceed the simultaneity constraint C_{sim}, are discarded as they are not considered to describe the same situation. The algorithm returns A_{sim}, which contains groups of simultaneous stream elements (data items). It is now up to the design of the INDP process, whether all groups of stream elements are used or only the most recent or most simultaneous set is used.

In practice and in the experiment described in Section 5.3, only set D with the smallest

Algorithm 1: Alignment and selection of temporally compatible elements

Input:

a) $S = \{S_1, ..., S_m\}$, where m is a number of streams

b) C_{sim} – a simultaneity constraint

Definitions and functions:

a) A_{sim} – An ordered array for sets of simultaneous sensor readings

b) $T\,(element)$ – returns the timestamp of a stream element

c) $I_{sim}\,(t_1, t_2)$ – returns a simultaneity interval of a set of timestamps

function align_and_select (S, C_{sim})

1. Sort streams according to priority
2. Choose stream with highest priority $S_1 = GetHighestPriority\,(S)$
3. $Foreach\,(element_{S_1} \subset S_1)$do:
4. declare an empty set D
5. insert $element_{S_1}$ to D
6. $Foreach\,(S_i \subset S)$, where $1 < i \leqslant m$ do:
7. compute $t_w = weighted\_average\,(D)$
8. find $element_{S_i} \subset S_i$, such that $\left|t_w - T\,(element_{S_i})\right|$ is minimal
9. insert found $element_{S_i}$ to D
10. if expression $C_{sim} \geqslant I_{sim}(D)$ evaluates true, then:
11. insert identified set of simultaneous elements D to A_{sim}
12. return A_{sim}

simultaneity interval $I_{sim(D)}$ is used in the data fusion and the other elements in A_{sim} are discarded. This step is needed to simplify the first iteration of the second vehicle detection experiment described in this thesis. The feasibility of passing all sets of D that satisfy the simultaneity and validity constraints to an INDP process depends both on available computational resources and time available for INDP process execution in a practical use case. Executing INDP process more than once, to consume all available inputs, would also produce a more consistent stream of INDP outputs.

5 Experiments and future applications for smart environments

This chapter describes three main experiments and a number of future possible use cases for Smart Environments. The first experiment is conducted in a military context. The experiment describes a use case for military application of a wireless ad hoc sensor network and demonstrates its applicability in a real world scenario. The second experiment investigates bandwidth usage and its dependence on contextual constraints used for the mediator. The third experiment demonstrates that applying selective mediation of interactions for sensor fusion increases its quality considerably. Then an entire section is dedicated for describing project SmENeTe2 and its future outlooks. SmENeTe2 (Smart Environment Networking Technologies) project was a project funded by Archimedes foundation established by Estonian Government. The Section 5.4.2 provides short descriptions of 6 possible future use cases of wireless sensor networks for Smart Environments.

5.1 Demonstration of system operation in military setting

This sub-chapter presents the description and results from a field demonstration for the European Defence Agency (EDA) project IN4STARS 2.0 (Information Interoperability and Intelligence Interoperability by Statistics, Agents, Reasoning and Semantics). The broad goal of the IN4STARS 2.0 project was to enhance information exchange and analysis for ISR (Intelligence, Surveillance and Reconnaissance) applications between multiple (and multinational) stakeholders. The field demonstration included a distributed, unattended sensor network with various sensor modalities (acoustic, motion detection, electro-magnetic and optical) enhanced with data validation and fusion capabilities. An example of an acoustic array sensor is depicted on Figure 32. The purpose of this ground sensor network was to detect the presence of adversary personnel and vehicles, classify the type of the vehicles and track their progress, while at the same time a nearby friendly unmanned aerial vehicle (UAV), equipped with a camera, was deployed to provide visual confirmation of the detected phenomena. UAV would receive instructions for acquiring imagery from sensor network system as described in Publication III.

The sensor network deployed featured a total of 16 sensor nodes: 4 microphone arrays implemented on 8-bit Atmel AVR based platforms, 4 microphone arrays implemented on BeagleBoneBlack (BBB) development boards, 3 proprietary military grade passive infrared (PIR) sensors for personnel detection, 1 proprietary magnetometer sensor, 3 camera sensors, 2 aggregation and fusion nodes and 1 autonomous UAV with a daylight camera. The field experiment was conducted on the grounds of a military base, with the sensors covering an area of approximately 1.5 hectares. The placement of sensor nodes can be seen on Figure 33. Sensor devices need to be aware of their precise locations to enable successful data aggregation and fusion in the network. As the sensor nodes used in the experiment were not equipped with a positioning capability the nodes were deployed manually and GPS coordinates of the positions were acquired with a GPS receiver and loaded into the nodes at the beginning of the experiment via the ProWare interface. Sensor node communication was established using MURP communication modules supplied by the company Thinnect, which have an IEEE802.15.4 compliant 2.4GHz radio and provide mesh networking. The effective communication range of the devices was 60m - 100m.

The demonstration scenario included different military vehicles passing along route A at different times, while area B was monitored to detect human activity (see figure 33). Upon detection of activity by sensor network, the UAV would be deployed to take pictures of route A or area B as required. Four different slow moving military vehicles were used: a light patrol vehicle, a light utility truck, a heavy truck for personnel and an armoured

Figure 32 – An acoustic sensor node with vehicle used in experiment.

personnel carrier. The speeds of the vehicles, when driving through the sensor network, ranged from 10km/h to 35km/h. The distances between the sensor nodes and tracked vehicles varied from 3 meters to 20 meters. All microphone array sensors, one PIR sensor and the magnetometer were placed along route A to detect vehicles, while other PIR and camera sensors monitored area B.

All sensor nodes were capable to perform initial signal processing and data analysis. PIR sensors detected motion and nearby camera sensors took pictures according to motion events received from PIR sensors. Acoustic sensors determined the direction to sources of noise and attempted to classify the source (in this case the four different vehicles). Aggregation and fusion nodes combined the individual direction estimates received from acoustic sensors to distinguish real phenomena (and establish their precise location) from random noise.

The set of situations that the WSN can detect, can formally be described using the notation of situation parameters referenced in Section 3.5.1. Basic situations are implicitly

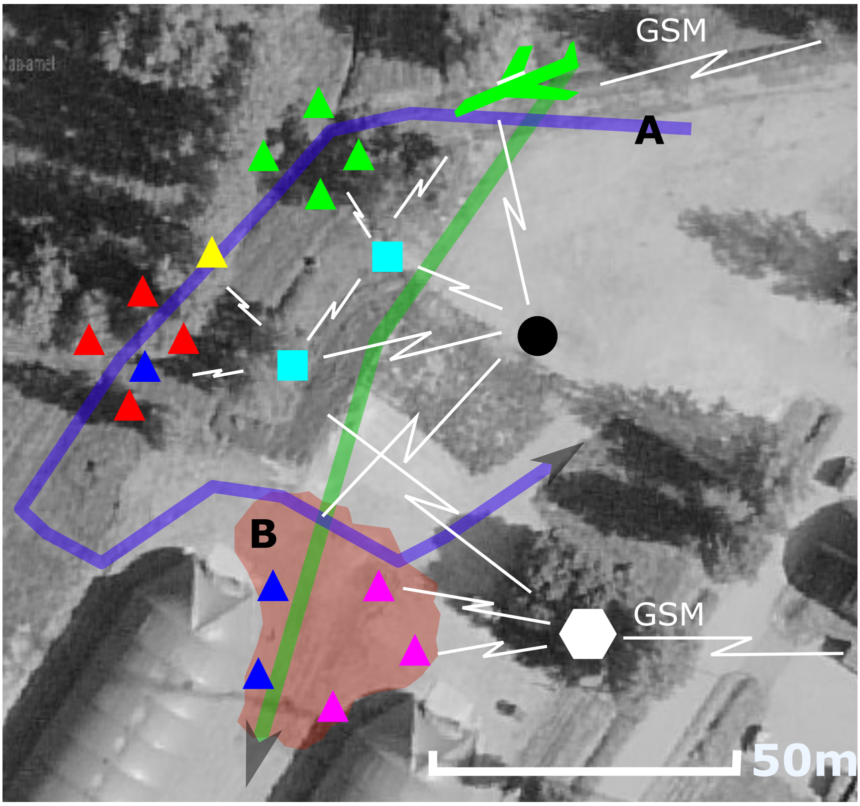


Figure 33 – Sensor node placement (green triangles - BBB acoustic sensors; red triangles - 8-bit microcontroller acoustic sensors; dark blue triangles - PIR sensors; pink triangles - camera sensors; yellow triangle - magnetometer; light blue squares - fusion and/or aggregation nodes; black circle - tablet user; white hexagon - gateway node; purple line - route A of military vehicles; pink area - monitored area B; green line - route of UAV.

defined for all sensor read-outs, while higher level situations must be defined based on the available basic situations and application needs. For example, a situation describing a vehicle's position (data type 'vehicle' with a location tag) is a sample of a higher level situation, one that is created by fusing basic situations that are only characterised by a vector tag (the angle), $S_{angle} = \{S_p, S_t, S_a\}$ in this case. The fusion function would be the function that calculates the intersection points of all beams (beams formed based on angle value S_p and sensor location S_a), while considering temporal compatibility S_t of the sensor measurements. Other higher level situations are created similarly, or by the aggregation technique, e.g. $S_{vehicle} = \{S_p, S_t, S_a\}$, where $S_p = \{S_{location}, S_{mag}, S_{class}\}$ and S_{mag} and S_{class} are respectively magnetometer sensor readouts and classification results, or $S_{hostiles} = \{S_p, S_t, S_a\}$, where $S_p = \{S_{pir1}, S_{pir2}, S_{camImage}\}$.

The data produced by the sensor network was accessible in two ways. Firstly, autonomous friendly military units in the vicinity could subscribe to sensor information via rugged military tablets with the specific user interface installed. Secondly, a remote database server was set up for far-away stakeholders (e.g. analysts from friendly nations). Tablet users were able to access the sensor network directly through wireless link, through an attached MURP device, and/or through a (GSM) gateway, while the database was connected to the WSN through a GSM gateway. The database is not a necessary component of the system (since local users can access the network directly), in the experiment it was used to collect sensor-data for post-experiment analysis and to supply other subsystems of ISR with input data.

According to the scenario, PIR and camera sensors would detect hostile activity in area B and notify (with pictures of detected events) a remote command centre through the network gateway. A friendly military unit was then sent to investigate the situation. Once it reached the vicinity of the sensor network, it started receiving the latest data about detected events directly to its tablet device. While investigating the situation, additional information would be received from acoustic and magnetometer sensors that would warn the military unit of approaching vehicles along route A. The acoustic sensors determine the location of the vehicles and try to classify them. This early warning capability enables the friendly military unit to retreat to a safe distance and order an UAV to come and survey the new activity. The UAV can, in principle, communicate with sensor nodes, once it is in communication range, and adjust its mission (e.g. adjust the area to be surveyed) to the latest information as described in Publication III. However in this experiment this was not demonstrated. The requirements and challenges of UAV and ground sensor network cooperation have previously been described in Publication III. The UAV used in the experiment is a small self-built tactical fixed wing aircraft. It weighs around 2 kg, has a wingspan of 1.5 m and its average flight time is 40 minutes. The top speed of the UAV is around 100 km/h. The UAV is depicted on Figure 34.

Figure 34 – UAV used during the experiment.

The experiment demonstrated a concept where signal processing, data analysis, distributed aggregation and fusion are performed inside the network by sensors or other special nodes and that users access this information directly, when in vicinity, over physical links and through service agreements established automatically at run-time based on existing information needs.

5.2 Moving vehicle detection experiment 1

The experiment demonstrated the use of temporal and spatial constraints for sensor data validity to manage the collection and exchange of situational information while taking advantage of both ISDP and INDP algorithms. More specifically the experiment evaluated how using the optimal constraints for ISDP and INDP benefits network communication by reducing communication loads and how it increases the dependability of overall sensor network computation results. Only one type of sensor nodes (microphone array sensors nodes) was used in this experiment. In addition, separate network nodes were used for fusing the data received from sensor nodes. The goal of using homogeneous sensors was to simplify the experiment and demonstrate the benefits clearly. The task of the sensor network was to detect sound emitting objects and estimate their location based on acoustic information collected by microphone array sensor nodes.

Each individual microphone array sensor alone can estimate a geographical bearing (direction) to a sound source from its position, but cannot effectively determine the distance to it, and therefore also the location of the sound source. Each microphone array sensor consisted of 6 microphones, the sampling speed of each microphone was set to 20KHz. The detailed description of the computation of the direction to the sound source - an AoA, and respective time difference of arrival (TDOA) method, is described in Publication II. A location estimate can be computed, however, by several sensors, which monitor the same area, by combining their direction estimates. This combination process (sensor data fusion) is depicted in Figure 35.

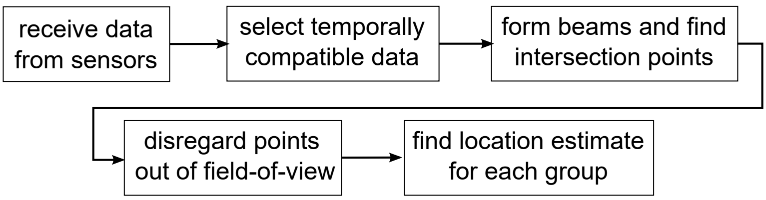


Figure 35 – Estimating the location of sound source.

The Figure depicts five steps for estimating the location of the sound source. While using this method for current experiment, the same approach of ensuring compatibility of input data can be used in wider context for other fusion and INDP methods. First, data are collected from all sensor nodes, which have detected a sound event. The data include the measured direction estimate - the AoA of the sound (a geographical bearing) and metadata, such as the location of the sensor node (geographical coordinates), the timestamp indicating the delay (or age) of the direction estimate and the sensing range of the sensor. Based on the age of each direction estimate, compatible sound event instances are found and analysed together. Next, the fusion process forms AoA beams along all the direction estimates and computes intersection points of these beams. Due to the discrete nature of AoA calculation procedure and other inaccuracies of input data, it is highly unlikely that the beams will intersect in a single point. Rather, a cluster of intersection points emerges and the dispersion or scattering of this cluster determines whether the result should be considered a valid location estimate or not. From this cluster, a single geographical position can be computed, which is a weighted average of the intersection points in the cluster. It is also checked that intersection points fall within the field-of-view of the involved sensors. Intersection points that are out of the range of the sensors are not considered.

This setup did not decrease generality, as all the essential ad hoc network characteristics that have been described so far, including in-sensor signal processing, in-network data fusion, subscription based sensor discovery and tasking and data validity checking were present in this experiment.

The three main claims that the this experiment should validate or refute are:

1. utilising a consistent system of contextual (e.g. spatial and temporal) constraints within the network is necessary for correct distributed data fusion and aggregation,

2. temporally and spatially consistent fusion enables to eliminate some of the false positive results of individual sensors therefore improving the quality of the end result, and

3. in-network fusion and aggregation reduces the number of packets sent to end-users (e.g. a database or any other user).

The goal of the experiment was to show that without any contextual validity constraints (or with very loose constraints) the fusion process (in this case object location estimation) will produce a lot of erroneous results (either false negatives or false positives). However, as the goal of the experiment was not to estimate the effectiveness of the fusion algorithm but the effects of mediated interactions on the system level quality of the entire network. The standard metrics, such as precision and recall, known from the field of machine learning and data fusion for characterising data processing algorithms are not well applicable for the current experiment. Instead, the precision in form of the area of computed positions (explained later in this Chapter) and either false negatives or false positives are chosen for evaluating the results. The fusion algorithm used in current experiment was previously developed and validated in laboratory conditions with high quality synthetic data and its analysis falls out of the scope of current thesis. So, without any contextual constraints (or with very loose constraints), the quality of the source data will be lower. which will affect the INDP processes in the network. This happens because the received data from sensors will be accepted and fused even if they are incompatible. It is also expected that the experiment shows that sensors on their own may detect many objects that are not actually there (false positives) due to environmental disturbance such as winds or heavy rain. In these cases the proper fusion processes can eliminate some of the false positives by fusing only contextually (e.g. spatially and temporally) compatible sensor data. Finally, a lots of sensor data messages (packets) were expected to being sent to fusion node, but much fewer fusion result messages (ideally only correct positive results) were expected being sent to the subscribing end-user. Sending fewer messages from the edge of the network saves node energy and network bandwidth.

For the experiment eight microphone-array sensor nodes (based on BeagleBoneBlack) were used to record 30 minutes of acoustic signals by the side of an urban street with moderate traffic. The same sensors were then set up in laboratory conditions where they, instead of recording live signals, now read the previously recorded acoustic data and treated it as if it were directly received from their ADC modules. Using this setup enabled repeated replay of the same 30 minutes of situations with different network and data constraint configurations and to compare the results. In order to compare fusion results to real objects it is necessary to know when vehicles passed through the area monitored by sensor network. During the experiment a video-camera also recorded the passing of each vehicle and later this video was analysed to count all vehicles and record their times of occurrence. This was done using the object counting software described in Publication II and the occurrence times were manually re-checked.

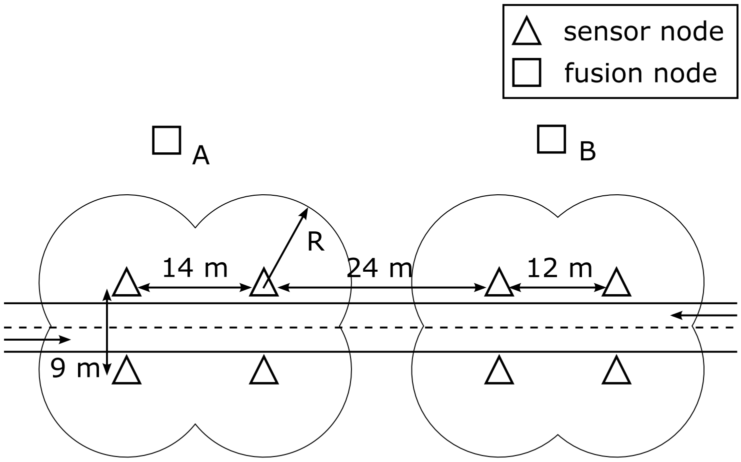


Figure 36 – Sensor node placement for vehicle detection.

Sensor network node layout is depicted on Figure 36. Two fusion nodes A and B were used with node A receiving messages from the four sensors on the left and node B receiving messages from four sensors on the right. The four sensors on the left are referred to as cluster A and the sensors on the right as cluster B. Sensor nodes were placed next to the street in order to detect passing vehicles. A total of 92 vehicles, of which two where busses, two were motorcycles, and the rest were cars, passed by the sensors during the 30 minutes. The speed limit at this stretch of the street was 50km/h.

The sampling speed for each microphone was set to 20kHz and measurement frame length, used in AoA processing, was 136.5 ms, i.e. 2730 samples per frame. As a result, approximately seven AoA calculations were done per 1 second. The results were sent to fusion nodes at an interval determined by the data subscription agreement between sensor and fusion nodes. The bandwidth experiment consisted of four different experiment runs. Each run with different parameters.

Table 1 – Message sending intervals and temporal and spatial constraints for four different experiment runs.

Experiment id and setting					
no.	Experiment name	sensor message sending interval	fusion message sending interval	age of sensor data	FOV circle radius
1	loose constraints	2s	3s	7s	40m
2	extreme constr.	2s	3s	1s	9m
3	optimal	2s	3s	3s	9m
4	high sending interval	1s	2s	3s	9m

The parameters for different experiment runs can be seen in Table 1. An explanation of the table columns follows. The sensor message sending interval of N seconds means that a sensor node will buffer AoA results for the last N seconds and at send-time only the latest valid result will be sent. Meaning only single latest sensor reading that satisfied the constraints. An alternative, not used in this experiment, is to send all buffered results at send-time and have the fusion node select which results it wants to use. On the other hand, most of the AoA calculations end with a negative result, meaning that no particular object could be detected. Negative results are filtered out by the validation process using the defined constraints. In case there are no positive results buffered that satisfy the defined constraints at the message sending time moment then, then nothing is sent to fusion node.

In order to monitor what happens with the network during different experiment runs all network nodes logged their activity. All nodes wrote different log messages to their serial port and several single-board computers (Raspberry Pi 2) collected these messages and time-stamped them upon arrival. The computers kept their own clocks synchronised via the network time protocol (NTP), so that all log records would be comparable (note that the clocks of wireless sensor network nodes themselves were not synchronised). Out of the all logged activities and results of network nodes, the most relevant log entries from the perspective of analysing and optimising the communications were:

1. sensor node message sending times
2. sensor AoA result values in these messages
3. sensor message receive times at the fusion node
4. fusion execution time instances and results

The bandwidth experiment consisted of four separate sub- experiments (see Table 1). The results of the experiments are presented in next chapter. The experiments differ by altering four selected input parameters, two that change message sending intervals of the sensor and fusion nodes and one for temporal and one for spatial validity constraints. Sensor sending interval, fusion sending interval and age of sensor data are all measured in seconds. For the experiments the spatial constraint set for data was defined as an area (the field-of-view) encircling each sensor and the radius (R on Figure 36) of this circle is set in meters. Formally, these four parameters, are related to each other by mathematical descriptions provided by Q-model [96] and thoroughly explained in Chapter 4.4. The actual time model of the Q-model [96] is more complicated, including among other features channel delays, execution time and start instance indeterminacy, etc., but these are not considered in this experiment. Experiments number 1 and 2 were performed with respectively very loose and very strict temporal and spatial constraints on sensor data, while message sending intervals were left unchanged. Altering data constraints should have considerable effect on fusion results and the successful positioning of vehicles. Experiments number 3 and 4 were performed with moderate and high message transmit rate (shorter sending intervals) for both sensor and fusion nodes. Data constraints in these two cases were left to what had been previously found as optimal values for good vehicle positioning. It was expected that high message transmit rate would cause more packet collisions and packet loss, which in turn would disrupt overall operation and the quality of results.

5.2.1 Results of the experiment

This sub-chapter presents the results of the first vehicle detection experiment and overall efficiency of the sensor network to detect vehicles. The overall results are presented in Table 3. Statistics of sent messages can be viewed in Table 2. Either cluster of sensor nodes has also been reviewed separately, because all sensor nodes were dedicated to their appropriate fusion nodes and no messages were sent between clusters (however, the single communication channel still had to be shared by both clusters). Generally cluster A performed better than cluster B in all experiments by detecting vehicles more precisely and by giving less false positives (i.e. successful fusion results, when there actually is no vehicle near the sensor network). This is a matter for the future research, as it is not clear what are the exact reasons for this. False positives and false negatives were reported by both clusters and are unfortunately inevitable unless additional sensors (e.g. a magnetometer) are added to the given network. This is understandable, as microphone-array sensors are intended to locate sound emitting objects, but not distinguish between vehicles and other environmental noise. During earlier testing, a case was documented, where the noise of a passing aeroplane fooled the sensor network to give consecutive false positives, the same can be caused by winds, heavy rain etc.

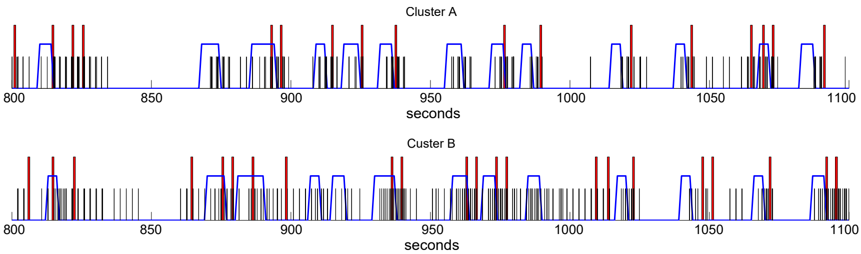


Figure 37 – Sensor and fusion computation results vs vehicles passing (tall red bars - fusion node results; blue trapezoids - vehicle passing; low narrow black stripes - sensor node results

Figure 37 depicts events that happened during the experiment. Data from experiment number 3 is used to depict the instances of successful fusion calculations (tall red bars), the instances when sensor nodes send messages (narrow black stripes) and time intervals when a vehicle was near sensors (blue trapezoids). Vehicle occurrence times are acquired from recorded video and plotted with 1 second granularity. Trapezoids with shorter width typically represent single vehicles and trapezoids with longer width either slow moving vehicles or several vehicles passing in close succession. The vehicle events depicted in either cluster are not precisely aligned, the shifts are caused by different directions and speeds of vehicles. The excerpt reveals that occasionally false negatives happen, i.e. both clusters fail to detect a vehicle (e.g. after 850 second for cluster A and after 900 second for cluster B), but between the two no vehicle is left undetected. It also shows false positive fusion results (e.g. at around second 825 in cluster A and just before 900 second in cluster B). The fact that fusion instances occur a few seconds after vehicles is because sensor and fusion node execute periodically at discrete intervals (see Table 1).

One of the three main claims of this experiment was that utilising a consistent system of contextual constraints within the network is necessary for correct distributed data fusion and aggregation. To prove this claim, the experiment assessed the effect of using different spatial and temporal data constraints on the fused sensor data. The results show that choosing appropriate constraints to suite the task at hand has major impact on the efficiency of the system. Comparing the results of experiments number 1 and 2, that differed only by the constraints set for fusion input data (see Table 1), the difference between the number of successful fusion results is greater than twenty times. Table 3 shows the number of unsuccessful fusion results (fusion results that were disregarded) because of either temporal or spatial mismatch of sensor data. Temporal mismatch of data disregards most of these unsuccessful fusions because it is the first constraint that is checked, data that pass the check are only then submitted to spatial validity checking and additional cropping. The spatial constraints are also checked by fusion algorithm while it computes the intersections of the beams formed on AoA's provided by individual sensors. Both experiments 1 and 2 represented extreme cases of data constraint usage, showing that with very loose constraints, there will be more successful fusion operations including more false positives and that with very strict constraints there will be fewer fusion operations and less real objects detected. An optimal set of data validity constraints (determined empirically during the course of experiments) was used in experiment 3. In this experiment the number of successful fusion operations was reasonable considering the total number of vehicles (92) and the number of false positives is in-between those of experiments 1 and 2. What is important is that when combining the results from both clusters only two vehicles were able to drive by undetected in experiment 3. Unless sensor technology itself is improved, this is one of the ways to deal with false negatives, i.e. by adding sensors (with different modality) and considering the results of more sensors.

Table 2 – Messages sent and received.

exp.	sensor node sent messages			fusion node	
	total	vehicle	no vehicle	received	sent
Cluster A					
1	1097	681	416	1084	224
2	1097	683	414	1091	10
3	1116	691	425	1089	122
4	2180	1341	839	2127	196
Cluster B					
1	1219	577	642	1208	264
2	1226	590	636	1217	10
3	1220	579	641	1219	123
4	2433	1163	1270	2406	210

The second goal of the bandwidth experiment was to see how many AoA results individual sensors produce and how many of these lead to successful fusions. Table 2 presents the total number of messages sent to fusion nodes by all sensors in both clusters. Each sensor message contains a new AoA value, so the number of messages reflects the total number of valid AoA results produced by sensor nodes. Comparing message send times with the all the times when a vehicle was in the field-of-view of sensors, reveals that about 40% of messages for cluster A and roughly half of messages for cluster B occur when no vehicle is present (see Table 2). This shows that the environment is noisy and a lot of sound sources are detected that are of no interest. Fusion improves the end result and decreases false positives by applying constraint checking and eliminating for example isolated AoA instances of individual sensors and concurrently produced random AoA instances of multiple sensors. From the Table 3 we can see that for experiment number 3, the ratio of false positives and correctly detected vehicles improves to 30% / 70% for cluster A and 44% / 56% for cluster B (consider Table 3 successful fusions). The fact that sensor nodes send AoA messages when there are no vehicles in their field-of-view and that fusion nodes mostly don't fuse these results, is also evident from the experiment time-line depicted on Figure 37.

The third and final goal was to determine the difference of bandwidth usage between forwarding all sensor messages and forwarding only fusion result messages for the higher SA level (e.g. to an operating military unit, a remote database, etc.). In all four experiments

Table 3 – Fusion results for both fusion nodes.

exp.	total	unsuccessful fusion		successful fusion		
		temp. mismatch	spat. mismatch	false positive	vehicle	total
Fusion node A						
1	342	91	27	55	169	224
2	180	170	30	2	8	10
3	285	113	50	36	86	122
4	393	141	56	60	136	196
Fusion node B						
1	418	98	56	100	164	264
2	303	283	10	7	3	10
3	368	154	91	54	69	123
4	501	162	129	92	118	210

the amount of fusion messages sent was approximately ten times less than the number of sensor messages sent (see Table 2). However, what is more important than the total amount of messages sent, is when they are sent. It is possible to see on Figure 37 that near vehicle passing times the number of sensor messages increases (sections of narrow black stripes get denser). This is because all sensors detect the presence of vehicle and want to use (the shared) communication channel at the same time. In our small sensor network of 8 sensors this did not cause a problem, not even for experiment number 4, where message sending intervals where changed, such that sensor nodes sent AoA results (when they had any) at an interval of 1 second. However, this would cause problems in larger networks that need to detect real-time events in real world scenarios. While the experiment was able to show that by utilising in-sensor signal processing and in-network fusion it is possible to limit the total number of messages generated by a sensor network, the experiment was not able to demonstrate significant packet collisions and congestion of our network at all. An experiment with either a larger number of nodes or shorter message sending intervals is probably needed to demonstrate this.

In conclusion the four experiments demonstrated the benefit of using in-network distributed data fusion to increase the dependability of sensor network results and to decrease the amount of data forwarded to end users. It was also shown that data validity checking (at least against temporal and spatial compatibility), is essential for correct data fusion. We expected to see congestion at network usage peaks, but did not succeed in creating situations where network becomes congested by an overflow of sensor messages. Since the minimum message sending interval of 1 second (see experiment 4) and cluster size of approximately 10 nodes is sufficient and reasonable for SA monitoring applications, the finding of communication breaking point of the wireless sensor network was not attempted at this moment. In general the sensor network was able to fulfil its task of detecting passing vehicles, although a considerable number of false positive results were also produced. This shortcoming can further be improved by adding additional sensors to the network.

5.3 Moving vehicle detection experiment 2

The purpose of this experiment was to demonstrate the operation of on-line alignment of stream elements for distributed sources and selection of contextually suitable elements as inputs for data fusion algorithm. In short a demonstration of the positive aspects of mediated interaction. For this experiment, the same setup was used as in the previous experiment. Only the sensor node sending period was set to 1000ms. In between the sending periods, the 7 sensor readings that the sensor node was able to sample covered 955.5ms, the sampling of frames (sensor process) was asynchronous with the sending period. At the end of each sending period, the sensor node assembled the available readings, which satisfied fusion node's validity constraints, into a batch of a single payload and transmitted it to the fusion node. In order to process all received readings, the fusion process execution period was also chosen to be 1000ms. The different execution times of sensor, sending and fusion processes for the experiment setup are illustrated in Figure 38. The delay arising from periodic activation at any fusion process execution instant can be described by formula $\varphi_n(t) = t_f - t_s$. The maximum delay due to periodic asynchronous processes with given settings can be up to 2000ms. The actual transport time depends on uncertainties induced by ad hoc network and environment. Considering maximum delay, the validity constraint for sensor readings in this experiment was chosen to be 2000ms.

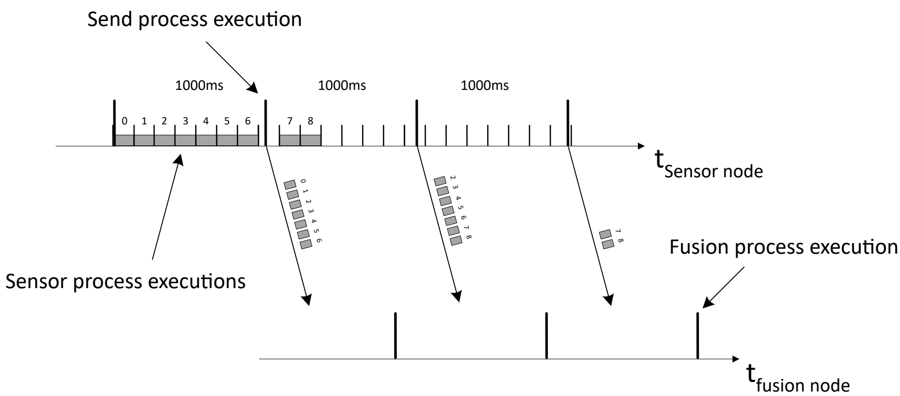


Figure 38 – The figure illustrates how number of transmitted values depend on the validity constraint. Note that all processes are asynchronous.

However, when no vehicles are near the sensors, the AoA calculations end with a negative result (if not considering noise from environment), meaning that no vehicle is detected, in Figure 38 these cases are illustrated as empty slots at the sensor process executions. Negative results are never sent to the fusion node. If previous AoA estimation results which are still valid at the sending time and no new AoA estimations have been computed, then the old results (within the validity constraint of 2000ms) are retransmitted to the fusion node. This means that some sensor readings could be used more than once by the fusion process. When validity time of buffered readings expires and there are no new positive results nothing is sent to the fusion node.

As already described during previous experiment, due to the discrete nature of AoA calculation procedure and other inaccuracies of input data, all the beams will very seldom intersect in a single point during the fusion process. Rather, a cluster of intersection points emerges and the scattering or dispersion of this cluster determines whether the result should be considered a valid location estimate or not. From this cluster, a single geographical coordinate can be computed, which is a weighted average of the intersection points in the cluster. It is also checked that intersection points fall within the field-of-view of the involved sensors (spatial check during the fusion process). Intersection points out of range of the sensors are not considered.

The resulting cluster of valid intersection points provides a basis for analysing the effectiveness of the fusion process. When the inputs to the fusion node are not acquired simultaneously, the resulting cluster of intersection points is more scattered as depicted in Figure 39(a).

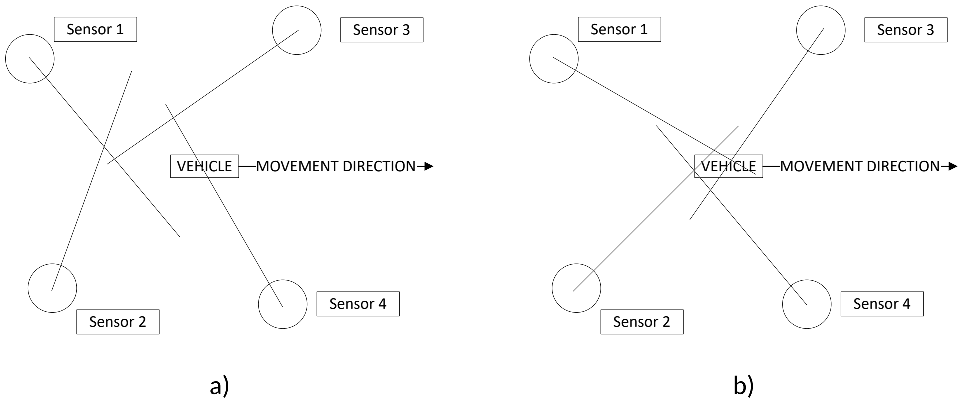


Figure 39 – Fusion result without data alignment (Figure a) and expected improvement with data alignment and selection (Figure b).

Figure 39(b) illustrates how applying time-selective data fusion strategy is expected to lead to improved fusion precision. In this case, provided the fusion node has access to streams of sensor readings that cover the vehicle passing, the alignment and selection algorithm should be able to select more compatible inputs for fusion.

In order to compare the experiment results, two separate parameters are used for analysis. These are simultaneity interval I_{sim} of the fusion algorithm inputs and the area of location estimation S_{loc}. The simultaneity interval I_{sim} of the fusion algorithm inputs describes the temporal dispersion of the computed delays of the sensor readings used as inputs. The second parameter, the area of the location estimation S_{loc} is a rectangular area covering the cluster of intersection points formed by AoA vectors provided by the sensors. The S_{loc} is a way to assess the scattering (or dispersion) of the intersection points. If the cluster of intersection points is more scattered, the rectangular area is larger and vice versa. The actual position estimation of the noise source is computed by taking a weighted average of all the intersection points. Our hypothesis is that there is a correlation between the I_{sim} and S_{loc}. The lower I_{sim} should result in smaller S_{loc}.

In order to monitor what happens in the network during different runs of experiment, the same setup as in previous experiment was used. All sensor and fusion nodes logged their activity by writing different log messages to serial port. This way the execution timesets for all processes, delays and other temporal parameters which cannot be otherwise extracted from the wireless processing environment could be recorded for analysis. Several single-board computers (Raspberry Pi 2) collected these messages and timestamped them upon arrival. The single-board computers kept their own clocks synchronised via the network time protocol (NTP), so that all log records were comparable (WSN nodes themselves were not synchronised).

The experiment runs were carried out the same way as the WSN would have been deployed in real world by replaying the recorded data streams at every sensor node. The WSN nodes used their radio transceivers to exchange the data as they would if they were deployed in the field. The two different configurations of experiments are listed in Table 4.

Table 4 – Experiments, their configurations and parameter varied.

no.	Experiment configuration name	Varied configuration parameter	value (ms)
1	Most recent data first	Validity constraint	2000ms
2			1500ms
3			1000ms
4			800ms
5			500ms
6	Temporal alignment and selection	Simultaneity constraint	1000ms
7			800ms
8			600ms
9			400ms
10			200ms
11			100ms

The configuration of first set of experiment runs was about using the most recent data first. This configuration did not use the temporal alignment and selection algorithm. The purpose of the experiment was to demonstrate the naive version of data collection from WSN, where each sensor node periodically transmits a result to the fusion node, which consumes the data in their order of freshness.

The configuration of second set of experiment runs applied the temporal alignment and selection algorithm, so that the temporarily compatible input data for fusion algorithm was selected from available inputs according to the similarity of the computed delays. This experiment configuration requires that the fusion process at every execution has access to several sensor readings or a segment of a stream of sensor readings from each sensor process. In the current experiment, due to the limited memory in the fusion node, a solution was implemented where instead of storing the stream elements on fusion node, the sensor node transmits a batch of readings in each of its packets. As the maximum length of IEEE 802.15.4 physical layer frame is 127 bytes, it was possible to transmit a maximum of 7 sensor readings (accompanied by appropriate metadata) in a single batch.

Both experiment configurations compute the delays of sensor readings using the same method as described in Section 4.3.6. The only difference is how the delay information is exploited. Without the alignment and selection algorithm, no simultaneity constraint is applied and the delays of sensor readings are only checked against validity constraints (the same validity constraint is applied both on sensor node before transmission and on fusion node side upon receival of data). The sensor readings with longer delays, which did not satisfy the validity constraints were not used for fusion. With the alignment and selection algorithm the inputs are projected and aligned to fusion node time domain and only temporally most compatible inputs are selected and passed to the fusion algorithm, provided they satisfy the simultaneity constraints. During all experiment runs, all execution periods for both sensor and fusion processes were set to 1000ms. The data validity intervals for fusion inputs were subjects to different validity constraints during first experiment and during the second experiment the upper validity constraint for fusion inputs is fixed to 2000ms.

The difference between the two experiment configurations can be better understood when looking at the Figure 40. This Figure explains why this type of alignment and purposeful selection of stream elements is needed. The Figure depicts a sample set of sensor

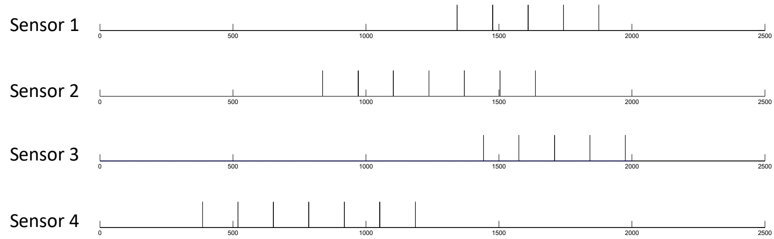


Figure 40 – An example of received and stored data with different delays.

streams as inputs for fusion process. In the figure, the streams from different sensor nodes have been projected onto fusion node time domain and aligned according to their respective delays. If the fusion process starts to consume the sensor readings by the most recent data from each stream, then the length of simultaneity interval I_{sim} of the resulting set of inputs will be more than seven hundred milliseconds.

On the other hand, if the fusion process is allowed to select temporally suitable elements, the value of I_{sim} is significantly reduced.

5.3.1 Results of the experiment

This section presents the results of a total of 11 experiment runs. The results for the first 5 runs are presented in Table 5. During these experiment runs, the temporal alignment and selection algorithm and simultaneity constraint were not applied. The results of the application of temporal alignment and selection algorithm on stream elements and the use of different simultaneity constraints are presented in Table 6. For both tables column no. 4 contains measured average simultaneity intervals for fusion inputs (a measure of temporal consistency of inputs) and column no. 5 contains average rectangular area of intersection points, which represents the precision of fusion result (a position of a passing vehicle). The results for I_{sim} and S_{loc} are averaged for each experiment, which is 30 minutes. The next column presents the number of completed fusions (successful fusion means that a position that satisfied spatial constraints was computed) and the last two columns show how many of the fusion results were false negatives and false positives. A false negative is a vehicle that was undetected and a false positive is a computed position where there were actually no vehicles present. Finding of false positives and negatives was possible, because the time intervals when a vehicle was in range of the sensors were recorded during the original field experiment.

Table 5 – Results without the alignment and selection algorithm, with application of validity constraint.

no.	validity constraint (ms)	Simultaneity constraint (ms)	*Average* (I_{sim}) (ms)	*Average* (S_{loc}) (ms)	Computed positions	False negatives	False positives
1	2000ms	not applied	947.5ms	34.4m^2	500	1	31
2	1500ms	not applied	700.8ms	31.7m^2	348	17	14
3	1000ms	not applied	506.9ms	22.2m^2	188	19	3
4	800ms	not applied	344.0ms	11.4m^2	90	47	0
5	500ms	not applied	140.0ms	11.3m^2	10	87	0

During the first five experiments presented in Table 5, the fusion node consumed the arrived inputs as most recent data first, in the same order as they arrived. The fact that the average simultaneity interval is relatively long can be explained by periodic and asynchronous execution of ad hoc WSN nodes. Furthermore, the allowable age for the most recent available sensor reading for transmission depends on the validity constraint. With validity constraint being longer than sensor process execution period, the sensor node was allowed to transmit or retransmit older values. Experiment runs 1-5 show that when the value of the validity constraint is reduced, the values of I_{sim} and S_{loc} improve. However, the number of false negatives quickly rises. The lower values of validity constraint filter out the sensor readings with longer delays. This does not improve the fusion reliability as with lower values of validity constraints more cars are left undetected. The effect can be explained by Figure 41, which presents an histogram from experiment 1 with measured sensor delays by fusion node A. During this experiment the validity constraint of sensor readings was 2000ms, meaning the sensor was allowed to retransmit the valid readings if there are no newer readings. The figure is illustrative as it depicts the delays without the application of temporal constraints.

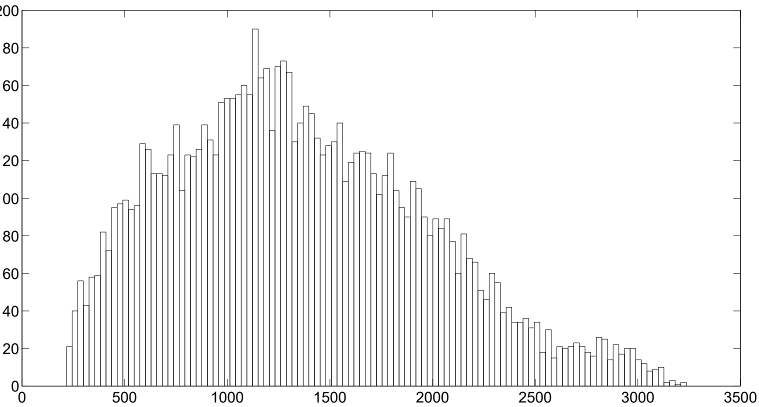


Figure 41 – Age of sensor readings in milliseconds as measured at the consuming (fusion) node.

The average for all communication delays of sensor readings received by the fusion node is 1357.7ms. Altogether the fusion node A received 9157 sensor readings. It can be observed that (due to periodic execution) the majority of the readings fall into an interval between 500ms to 2000ms. The reason why there is ca 200ms delay before the first readings arrive to the fusion node must be, in addition to the sensor processing time, fusion node's asynchronous and periodic execution (this can be explained by the fact, that only the sensor reading production is started synchronously at the beginning of the experiment, the fusion nodes were started in no specific order before the experiment). The readings that have been delayed more than 2000ms are most likely the ones that were retransmitted due to the fact that no new valid readings were available. The theoretical maximum of a delay due to periodic execution and retransmission can be up to 3000ms (validity time added to delay caused by periodic execution of processes). Longer delays must have been caused by network and environment induced uncertainties (or other real world unpredictable causes).

The rest of the experiment runs (6-11) in Table 6 show how averaged values for simultaneity interval and area of location estimation were influenced by alignment and selection algorithm together with different values for simultaneity constraints.

The experiment indicates a correlation between the area of average location estimation and the simultaneity constraint. The lower the simultaneity constraint, the smaller

Table 6 – Results with the alignment and selection algorithm, with application of simultaneity constraint.

no.	validity constraint (ms)	Simultaneity constraint (ms)	*Average* (I_{sim}) (ms)	*Average* (S_{loc}) (m^2)	Computed positions	False negatives	False positives
6	2000ms	1000ms	208.3ms	22.7m^2	446	0	14
7	2000ms	800ms	180.8ms	21.2m^2	428	0	12
8	2000ms	600ms	147.5ms	19.8m^2	404	1	11
9	2000ms	400ms	86.1ms	17.6m^2	342	2	8
10	2000ms	200ms	58.9ms	17.1m^2	284	7	3
11	2000ms	100ms	6.0ms	16.7m^2	200	20	2

the area i.e. the precision of position estimation improves. However, the same side effect as during the first five experiments without the temporal alignment and selection algorithm is present. Stricter simultaneity constraint filters out the actual vehicle detections with lower precision (larger values of S_{loc}). For example, the usage of simultaneity constraint 100ms leaves 20 vehicles undetected (false negatives).

Experimental results of the two different experiment configurations (most recent data first vs time-selective strategy) clearly show that time-selective approach achieves considerably better results than the configuration which uses only validity constraints and prefers the most recent data first.

Choosing a good criterion for WSN performance estimation is not trivial. One possibility is to use accuracy as a criterion. In statistical tests, accuracy can be measured by Equation 8.

$$Acc = \frac{(TP+TN)}{(TP+FP+FN+TN)} \tag{8}$$

In Equation 8 the Acc stands for accuracy, TP for True Positives, TN for True Negatives, FP for False positives and FN for False Negatives. It can be seen that the accuracy is increased if either false positives or false negatives or both are decreased. However, for these experiments the low number of false positives and low area of computed positions (S_{loc}) was considered as the most important outcomes. The number of false positives should be low as the false alarms are not desired in order to avoid the false positions computed based on false alarms. Considering the results of all experiments carried out, a number of 3 false positives is chosen as the maximum acceptable value. The area of computed positions S_{loc} is used as a vehicle tracking precision by the system (whole network, fusion algorithm, alignment and selection algorithm are viewed together as a system). The other outcome parameters to be considered are the number of computed positions (successful fusions) and the number of false negatives.

With the first experiment configuration (the most recent data first approach), the best results are with validity constraint being 1000ms, which is the first threshold, where the number of false positives is 3. However, the number of false negatives is too high, 19 false negatives out of 92 vehicles leaves 20.7% of vehicles undetected. In total this leaves only 73 vehicles detected with 185 correct positions. The average precision of positions was $22.2m^2$.

The second configuration shows much better results. The outcome of simultaneity constraint of 200ms gives 3 false positives and less than 7.6% of false negatives. In total, 85 vehicles of 92 where detected with 284 correct positions. The average precision of positions was $17.1m^2$. It can be noticed that the average S_{loc} is getting more stable after the simultaneity constraint of 400ms. This indicates that more precise average position is difficult to achieve (the reason for this could be that the frame start is chosen as timestamp for sensor readings, not actual sound event).

In conclusion, with the same number of false positives in experiment runs no. 3 and no. 10, the second experiment configuration with time-selective algorithm showed significantly less false negatives (decreased by more than 6 times). Experiment run no. 10 also improves the $Average(S_{loc})$ by 23.0%. When using more strict constraints in either configuration (with runs 4, 5 or 11) the constraints start to filter out too many detections. It can be concluded also that the constraints were too relaxed in runs 1, 2, 3, 6, 7, 8 and 9. However, the task of this experiment was not to find best configuration but to demonstrate the necessity of time-selective handling of input data for in-network processing, which was clearly confirmed with the experiment.

5.4 SmENeTe2 future use cases and applications for an urban Smart Environment

The specific objectives of this chapter are to provide a short overview of the implemented sensor network (the test-bed) and to describe some cases where extending the ProWare with additional functionality is needed for implementing Smart Environment applications. This chapter also demonstrates that the developed concepts are applicable on the larger scale.

5.4.1 Short overview of the SmENeTe2 sensor network

The SmENeTe2 project has resulted in installation of approximately 900 sensors on the street light poles of Tallinn city, each sensor is equipped with a power storage cell and a solar panel. The computational platform of sensor nodes for the Smart Environment use cases are based on energy efficient 32 bit SiLabs Mighty Gecko platforms. The nodes' software platform utilises FreeRTOS as underlying RTOS functionality provider and ARM CMSIS RTOS abstraction layer as API). As in previous experiments more powerful platforms were used for computationally complex algorithms, in the use cases, only SiLabs platforms are used. The preliminary results of the experiments and review results of the use cases were applied as an input to draft a revised architectural design for the ProWare middleware to support Smart Environments. The network consists of four different types of sensor modules (SM):

1. SM1 - Environment sensor (weather and air quality)
2. SM2 - Microphone array sensor (noise level and direction to the noise source)
3. SM3 - Microwave radar sensor (movement and traffic density)
4. SM4 - Simple microphone sensor (noise level)

All sensors also include capability to monitor several internal aspects, such as internal CPU temperature, battery temperature, battery voltage, current consumption, solar panel voltage level and vibration (all sensor modules are also equipped with an accelerometer). The function of the sensor network is to collect data both from the environment and about the city traffic flows. In addition to usual data collection and processing tasks the information provided by the network forms the basis for Smart Environment applications that are being designed and partially are under development. Smart Environment applications differ from conventional data collection in that the information should be locally available, dynamically renewable and can, in principle, be consumed for decision-making to foster rapid solution of many issues in the fields of transport, communication, security and many others. Secondary purpose of this network is to provide interesting research cases for scientists at Tallinn University of Technology. The different cases comprise for example both long lines of sensors along main boulevards and true mesh network in a central area where at places the sensors are deployed in a grid-like fashion. It can be said that this network has a potential to turn a city into a Smart Environment for its inhabitants.

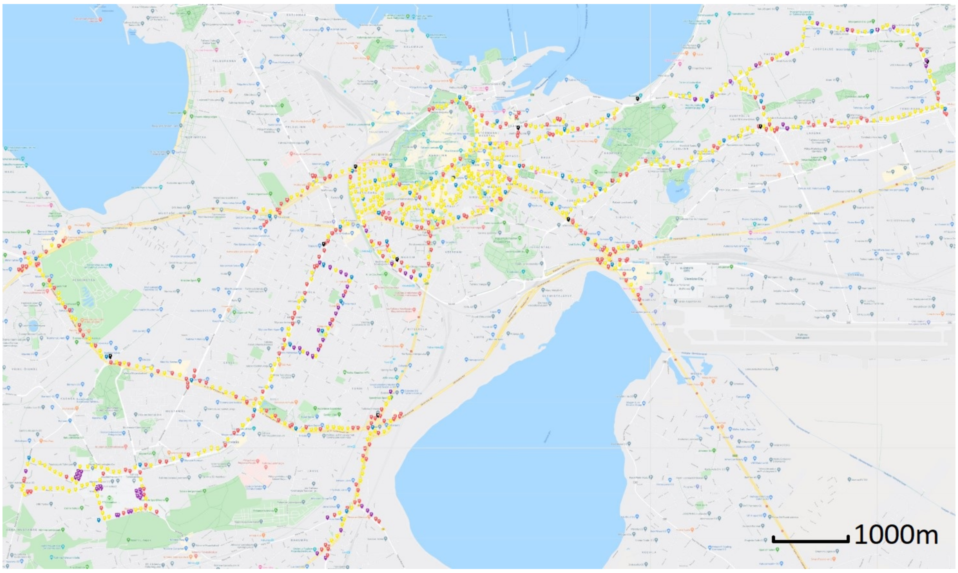


Figure 42 – Map of the SmENeTe2 wide-area sensor deployment. The purpose of the image is to give an overview of the scale of the deployment. Blue icons depict sensors of type SM1, purple icons depict sensors of type SM2, red icons depict sensors of type SM3 and Yellow icons depict sensors of type SM4. The black icons depict the gateway nodes.

5.4.2 Smart Environment possible future use cases

Below some of the use cases are described that have not been implemented yet while this thesis is being written. These examples give an idea of the future applications of urban Smart Environments.

5.4.3 Case 1. Example of traffic flow estimation with microwave radar sensors

One example application of Smart Environment is traffic flow behaviour estimation. Microwave traffic sensors of type SM3 deployed in succession along the street can be used for example to estimate how traffic flow is behaving. Is it slowing down, is it speeding up, has it come to complete stop or has its behaviour been stable over a longer period. The algorithm for the sensor is described in Publication VI; MW sensor of type SM3 uses Doppler effect to compute the speed of passing vehicles. The passing event is computed by measuring increase of energy of signal frequency spectra. This data could be used for real-time traffic overview applications for drivers. So that drivers could be able to choose faster routes.

5.4.4 Case 2. Example of energy efficient use of microwave radar sensors

Microwave radars consume considerably more power than other typical Smart Environment sensors. The example sensor used in SmENeTe2 project is MDU2750L. It is a 9.90GHz motion detector unit utilised as a doppler radar. The peak operating power consumption of this sensor is 25mA. This is a lot, considering the available battery capacity of the sensor unit being 6000mAh. In continuous use, this sensor would drain the battery in 240 hours. Adding the power consumption of the platform lessens the sensor up-time even more, however, specific calculation is out of the scope of this thesis as the expected lifetime of the whole sensor node running only on battery is one month, this is $31 days * 24 hours = 744 hours$.

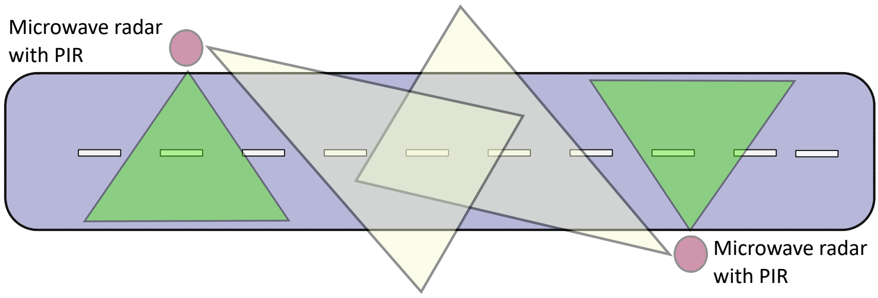


Figure 43 – Cooperating microwave radars with PIR.

One possible solution to the power consumption challenge is to utilise INDP, applying a cooperative solution depicted on figure 43. In this use case two sensor nodes are placed along the rode on opposite sides. Sensor nodes are denoted with red rectangles and are equipped with two sensing units. The first sensing unit of a sensor node is microwave radar MDU2750L, it is deployed towards incoming traffic at a 30 degree angle, depicted with yellow triangle. The second sensing unit is a passive infrared sensor, deployed perpendicularly towards the road, depicted with green triangle. For motion detection, a passive infrared sensor of type EKMC1601112VZ is used. This sensor has a maximum power consumption is 170μA, which is several magnitudes less than the radar sensor. Both sensing units are integral parts of the sensor node. The fields of view are denoted as large triangles, the radars field of view is coloured yellow and passive infrared field of view is coloured green.

In normal operating mode, the microwave radar measures the speed of approaching vehicles and also detects the time instant when the vehicle passes the sensor. The task of the PIR sensor is to confirm that the vehicle is passing the sensor node. However, in case sensor nodes are allowed to exchange data produced by movement detection measured by passive infrared sensing units, they can switch off their radars while there is no traffic. This would save a considerable amount of energy, especially during night time, when traffic flow is less dense.

5.4.5 Case 3: Microphone array sensors for positioning the noise source

Each single Microphone array sensor (SM2) produces a stream of sensor readings where each reading represents an angle to the noise source (AoA). When several sensors of type SM2 are monitoring the same area, a position of an acoustic phenomena e.g. a passing vehicle can be computed. Stream elements (sensor readings) from several sensors can be temporally aligned by the ProWare middleware (Proactive smart mediator) and used as inputs for computing the position of the noise source. The examples of this type of use cases have also been described in Publication II and in this thesis in sub-chapters 5.2 and 5.3.

5.4.6 Case 4: Microwave radar sensors for computing the length of a queue of vehicles in front of traffic junctions

Each single SM3 sensor produces a stream of sensor readings, where each reading represents a speed of a passing vehicle. Lets suppose that several SM3 sensors are monitoring a stretch of a lane, for example few hundred meters just before a cross junction. By using data about passing vehicles speed and count, an estimate can be computed for the length of a growing queue of vehicles waiting for the green traffic light. Locally each sensor computes an average speed and frequency of passing vehicles over a certain time interval, e.g. 2-3 minutes. Streams of these data from several SM3 sensors can be temporally aligned at the fusion node. The aligned readings from different sensors with certain simultaneity criteria can be used as inputs for computing the length of queue of vehicles in front of a traffic junction.

5.4.7 Case 5: Complementary microwave radar sensors

Each single SM3 sensor produces a stream of sensor readings where each reading represents a speed of a passing vehicle. Sometimes the quality of this stream of speed data is degraded (is intermittent in time and has occasional caps), especially when vehicle is still far from sensor. It may be possible to compensate this by another SM3 sensor placed on other side of the road, so that there is an overlap of the field of views of both SM3 sensors. The streams of data from both sensors, temporally aligned, could give a better estimation of vehicles speed.

5.4.8 Case 6: Traffic related sensors give input to traffic signs (traffic signs adapt to situation)

Vehicles approaching blind intersections could potentially receive warnings on incoming traffic via LED traffic signs. Traffic sensors placed along the road behind the blind intersections would provide this data upon a request. The problem is how would sensors know that they need to provide data for the traffic sign. The solution would be that the sensors detecting approaching car would make a request for data from sensors behind the blind corner. Potential challenges: the sensors detecting the approaching traffic need to reach a consensus that there is a traffic.

6 A middleware design for Smart Environments

This chapter gives an overview of a middleware design provided for SE. While preparing this thesis, the author worked with a Smart Environment networking technologies design project (SmENeTe2). Both the experiments described in Chapter 5, the theoretical work presented in the thesis and the work done during SmENeTe2 project has led to these suggestions for the middleware design. SmENeTe2 project was an applied research project ordered by a private company Thinnect OÜ and supported by Archimedes Foundation. The project was carried out between July 2017 and December 2019. During the project, the author together with the research team from Research Laboratory for Proactive Technologies (ProLab) and Thinnect OÜ engineers, who all worked towards the three main goals of the project:

1. Validate applicability of Thinnect patented mesh networking layer for scalable Smart Environment applications and to recommend changes to the networking layer based on these application requirements.
2. Extend the proactive middleware - ProWare that had been developed at ProLab to offer the functionality required for Smart Environment applications.
3. Develop functionality for in-situ diagnostics of wireless network with cloud based support.

This sub-chapter gives a brief overview how theoretical and experimental work in this thesis is applied for achieving the second goal of the SmENeTe2 project. The second goal of the project is directly related to the thesis as both the theoretical work and practical results of the experiments can be applied for achieving this goal. ProWare is modelled as a proactive and smart mediator agent that utilises theoretical concept of mediated interactions to support Situation Awareness applications [95]. Earlier works have described specific functionalities of ProWare: computation of situation parameters [108], data encoding format [95], subscription based data exchange [111] and on-line data validation capability [107]. This thesis has continued this line of work and, in order to enable Smart Environment applications, contributes with the functionalities of processing of streams of situation parameters as described in Publications III and I, time-selective data processing as described in Publications II and I and time-selective alignment of stream elements from distributed sources as described in Publication I.

6.1 Description of the work on extensions to the ProWare

This section describes the extension of existing "ProWare" solution to support Smart Environment applications (that can be also used in the context of the SmENeTe project). ProWare is a multi-functional middleware, a communication layer that ties together all the heterogeneous sensors, provides an environment for data exchange and enables distributed fusion and aggregation in wireless sensor networks. The focus of ProWare properties is on a capability of dynamic configuration of temporal and spatial properties of interactions between wireless nodes.

In order to offer the functionalities required for Smart Environments applications, it became necessary to enhance the effectiveness of in-network computation. ProWare is a middleware for IoT nodes which enables to build distributed service-based application implementations. A high level description of ProWare can be created by listing ProWare services:

1. Producer discovery

2. Agreements on data exchange
3. Agreements on data delivery
4. On-line data validation

The contributions of this thesis developed for ProWare are based on SA requirements described in 2.4.1 for Smart Environment applications. While the old version of ProWare enabled execution of a service-based application scheme on embedded nodes, the functionality was still limited. The major deficits that were identified by the SmENeTe2 project team, were:

1. The production of data was coupled to the data delivery service
2. The application of time selective data processing and temporal constraints was not tested in real applications
3. The middleware did not support temporal consistency check of data across multiple streams
4. The middleware did not enable complex (multi-agent) embedded application schemes that require temporal alignment of data

All of these functionalities are required in Smart Environment applications. An oversimplified structure of the middleware that supports all these missing functionalities is depicted on Figure 44. Explanations for each functionality is provided in the following sub-sections.

6.1.1 Decoupling the data production and delivery

Contemporary Smart Environment applications include more demanding sensing capabilities that include much more complex data processing than just generating a certain situation parameter, such as a temperature reading or a voltage level. The new types of low cost and low power sensors that have become available (e.g. microwave radars as described in Publication VI, video based object counters and acoustic arrays as described in Publication II) demand raw data processing, which time-wise can take several hundreds of milliseconds as described in Publication II. In this case it is no longer reasonable to keep sensor data production coupled with the delivery service, as the data producer process cannot predict when the data delivery process is available. To solve this, this thesis suggests that after receiving the respective subscription, the data producer process should start running continuously i.e. executing and providing data periodically without any pauses. The produced situational parameters about phenomena of interest are buffered into suitable data structures (e.g. same way as the producer process's sensor readings are buffered in a ring-buffer like memory structure) and each time instant, when the delivery is about to start, ProWare middleware should be able to access the already produced and buffered data, compliant to consumer contextual constraints, and to select the suitable sensor readings for the delivery. If more than one available sensor reading satisfies consumer contextual constraints, then the readings are aggregated into arrays. Due to the nature of the ad hoc mesh network, an array type of an element makes sense. It is not feasible to send each reading as a separate packet, as low utilisation of payload does not increase throughput. The selection of the suitable data by the middleware is carried out when the delivery process becomes available. The asynchronous channel between data delivery and data production processes is described earlier on Figure 27. This decoupling of the delivery and production results also in one of the main changes in

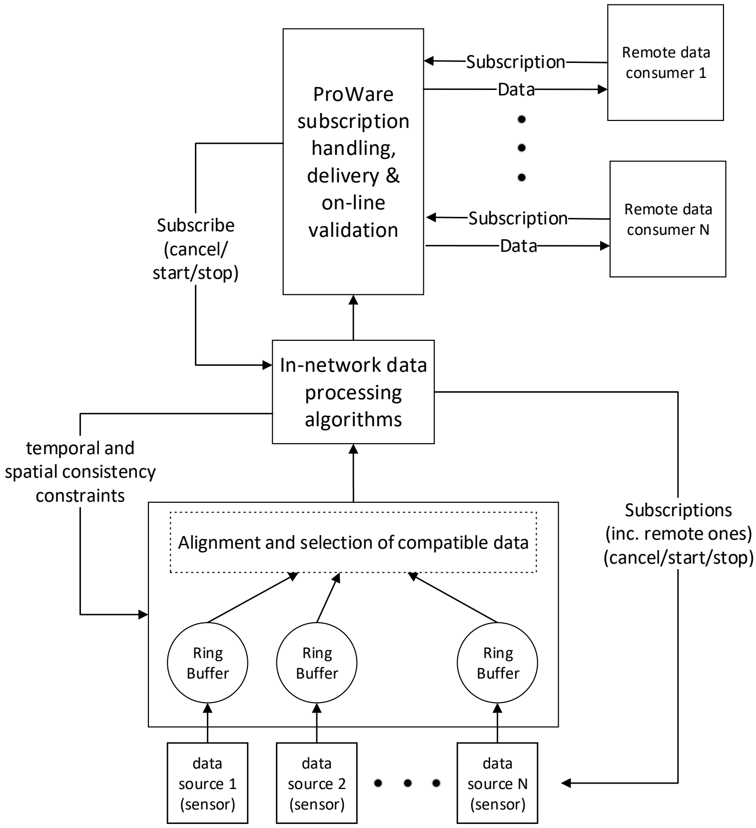


Figure 44 – In-network data processing with ProWare middleware.

ProWare architecture, it is now necessary to have separate modules for incoming and outgoing streams. On Figure 44 the decoupling is depicted in the form of ring-buffer memory structures. Note that there is only one box containing INDP algorithm on figure. In reality, there could be more than one (the number depends also on physical limits, such as memory and processing power) algorithm being executed. The same way as there can be several data sources, there can also be several INDP algorithms implemented, they can be either have independent clients or there can be higher level algorithms implemented which in turn again assume memory buffers and data alignment algorithms.

6.1.2 Support for stream-like data handling in DIL ad hoc network

As described in previous section the sensor signal sampling, processing i.e. the production of data and its delivery to recipient should be decoupled into separate asynchronous processes already on the provider side. This is also described on Figure 44. On the Figure it can be seen that the data sources, which can be both local sensors or data streams from distributed remote sensors feed data to ring-buffer like data structures. The next level process can in turn at any time moment read data from the memory (provided that buffer is filled with the data and reading and writing to same memory address do not happen at same time). This gives the impression of working on a stream on both producer and consumer sides. Abstracting to a stream makes it easier to define stream operations such as queries, selectors that can be used to extract (choose) data from the stream. However, it is not formally correct to call it a stream, as the asynchronous decoupling described in previous section results only in a stream-like data production by network nodes. It is called "stream-like", because, even if the node produces periodic data which is guaranteed to stay within the constraints described by the subscription (via on-line validation service), some of the data can be missing as the execution frequency of the delivery service might either not be frequent enough to transport all the data produced, the packet structure or communication protocol may not support the transport of all data produced by the sensor node or packets may be lost in transit. In this way the consumer of the data receives a Disrupted Intermittent and Limited (DIL) stream, which elements can further be occasionally variably delayed by self-organising network structure or simply lost due to disrupted communication channels. The DIL properties of the network, addressed in section 2.3, make it unreasonable to schedule the data delivery coordinated by data production. It is far more reasonable to use asynchronous communication approach and to buffer a number of recent sensor readings and let the ProWare middleware choose suitable readings from buffers according to the temporal constraints set by consumer.

6.1.3 Data delivery policy

The suitable data delivery policy should be chosen so that it supports SA information collection and exchange in Smart Environments applications. Meaning, one should not view the selected policy from the perspective of the single pair of consumer and producer, but from the perspective of the whole part of the network involved in the computations. The description of the data delivery policy should be abstract enough for coping with the properties of ad hoc mesh network used for Mist Computing. The data delivery policy should depend on at least two parameters: the contextual validity constraint and the priority (or relevance) of the stream. The contextual validity constraints, described in 4.3.1, defines which readings are valid for delivery in dimensions like time, location, etc. The priority of the stream defines the order in which the streams are handled (some streams are more important e.g. link quality diagnostics or theft alert). In case of time-selective INDP the priority of the stream defines which stream is used as a reference stream when selection of compatible elements from different streams is carried out, as explained in creater details in sections 3.5.2 and 4.4.6.

6.1.4 Priority of a stream

In general the priority of the stream is defined in the subscription. As it is the client that knows which data is more important for inference or description of a specific situation. For example, a high quality classification result may have higher priority than a movement detection. In addition to the stream priority defined in a subscription by client, there must be additional rules concerning priority of data inside a stream. Some simple rules are established:

1. The most receent data from the interval $[\mu, \nu]$ should always be delivered first.
2. Queued data is only delivered if possible.
3. If whole interval $[\mu, \nu]$ does not fit into the single packet, it is fragmented and delivered in several packets.
4. However, if current transmission (delivery service) window closes before all fragments are delivered, the validity of the available data in buffers is re-evaluated when next transmission window opens.

5. It can be that some readings that could not be delivered in the previous transmission window have now lost their validity. These data can also be transmitted but only in case there are no other valid data or data with higher priority queued for transmission.

6.1.5 Time-selective data processing

Time-selective data processing forms a major part of the concept of mediated interactions used in current work. The first experimental implementation of the middleware that supports time selective data processing in ad hoc sensor networks for SA applications is described in Publication II, where author of the current thesis together with Johannes Ehala describes an implementation of INDP and analysis in wireless sensor networks (WSN) for creation of situation awareness. The work described follows ProLab's theoretical work conducted over several decades on managing data integrity, consistency and validity when data for processing are selected from multiple heterogeneous sources. Although the article does not explicitly mention the time-selective data processing the work described in that article fits naturally with the essence of time-selective data processing and also with the concept of mediated interactions. Specifically in that article it was demonstrated that aggregation and fusion of data received from various different sensors benefits considerably from applying middleware supported checking that the data used for processing is valid and relevant. Aggregation and fusion are performed at run time by dedicated nodes and results are made available to appropriate stakeholders. This contrasts the widespread practice of collecting all sensor network data to central database and performing analysis only after collection. This approach is relevant for SA systems where sensor data interpretation must be made operatively during run time already within the network and results must be made available immediately. The handling of INDP and local situational information exchange is absolutely necessary for Smart Environments as the whole idea of the Smart Environments is that the inhabitants occupying it can be in interaction with the environment and can extract useful information from it in real-time. These are the same expectations that one has for SA applications, which is natural as Smart Environments in most cases are also built to enhance SA of its occupants as was explained in the beginning of the thesis. The results from experiments described in current thesis show that time-selective data processing (on-line checking of validity and relevance by the proactive middleware ProWare) based on temporal and spatial validity constraints on data can have a considerable effect on in-sensor data processing. The difference between having very loose or very strict temporal and spatial constraints resulted in approximately a 20-time difference in the bandwidth usage of the information generated by the sensor network as described in Chapter 5.2. This makes it clear that both the SA and Smart Environment applications can be greatly effected by changing temporal constraints which is possible when applying time-selective data processing. The formal approach to time-selective data processing is explained in section 4.4.1.

6.1.6 Temporal alignment of streams for Smart Environment applications

The temporal alignment of streams is directly related to time-selective data processing. The basic idea of stream alignment is that the data consumer gets access to temporally overlapping segments of data elements from all streams. The streams are aligned on INDP node's time axis. In case the data consumer and data producer are on disparate platforms, which clocks are not synchronised, then the timestamps of stream elements are estimated as described in sub-chapter 4.3.6 . In practical applications, as described in experiments in Chapter 5, it was shown that not all streams available are needed for INDP. In case of

WSNs, the data sources have often been deployed with a redundant policy, thus only a sufficient number of overlapping stream segments might be required. The overlap of the streams can be achieved by 1) either a setting longer intervals for windows of $[\mu, v]$ or 2) adjusting the limits of windows differently for each stream until they have a sufficient overlap (data consumer can find mutually compatible data that satisfy contextual constraints defined in subscriptions). The latter assumes, that end-to-end delay is roughly known. Setting longer intervals is problematic, because there are limits to how much data can be stored in the memory of low-cost embedded devices. Adjusting the windows of $[\mu, v]$ for individual data producers is more feasible as it can be done on the basis of known or estimated end-to-end delays. The estimated end-to-end delays are estimations based on on-line measurements of message delivery time during the INDP. However, a new problem emerges in the case where the consumer node attempts to set different windows to producers in order to achieve temporal alignment of all incoming streams with the stream that has the highest end-to-end delay. In order to provide only data that satisfies the required temporal constraints given by interval $[\mu, v]$, the data providers with lower end-to-end delay will have to buffer more data before transmitting (explained in 4.4.4). By choosing the stream with highest end-to-delay as a reference also results in the increase of the overall delay of INDP algorithm.

Cases where "alignment" of sensor data is required:

1. Heterogeneous sensors with different production periods
2. Several sensors with similar production periods but with intermittently missing readings
3. Sensors with random wake-up times
4. Event based sensors

Another function of temporal alignment is finding temporally closest or compatible elements over several stream segments. This is done on the basis of stream priority or (relevance of situation parameters). The streams are ordered by priority and the stream with the highest priority is chosen as a reference. In short, the alignment algorithm loops over the elements from reference stream and finds temporally closest elements from other streams. This procedure is thoroughly explained in Chapter 4.4.1. The sets of data elements from different streams, which satisfy with the consistency check can be used for INDP algorithms.

In addition to temporal alignment some INDP algorithms require that input data is temporally consistent. The temporal consistency of two data elements means that in temporal domain they can be assumed to describe the same event. To ensure consistency between data elements received as inputs from disparate sensor nodes, this thesis suggest to use a simultaneity constraint, specified in subscription for INDP data. If this simultaneity constraint is satisfied, it provides an approximation of causal relations between the situations observed by remote sensors.

7 Conclusion

There is major a trend towards Edge Computing (EC) i.e., moving the computational intelligence and data storage capability closer into the physical and social environments. The Cyber-Physical Systems that are used as IoT devices are embedded by thousands in Smart Environments (SE) where their main purpose is to provide situational information that can be used to update the Situation Awareness (SA) of both humans and other artificial agents for effective decision making. However, distributed wireless ad hoc and mesh IoT devices forming a Systems of Systems (SoS) in a SE do not communicate synchronously, leading to a situation where devices that perform in-network data processing, receive data from distributed sources that may be out of order and inconsistent. Furthermore such systems exhibit also emergent behaviour, i.e., behaviours that were not considered during design time, leading to unpredictable delays and production of invalid situational information. Building Situational Awareness with inconsistent or invalid situational information leads to incorrect decisions. To solve the temporal ordering and inconsistency problems of data, this thesis analysed existing timing approaches and offers a novel solution in the form of mediated interactions. This thesis also argues that collection and exchange of situational information for SA applications requires an understanding of what is the SA and hypothesises that this understanding helps to build better systems for collection and exchange of situational information. After providing an overview of the concept of SA, this thesis identified three aspects where SA theories can improve the data collection and exchange in SEs:

1. moving intelligence closer to the network edge,
2. basing collection, exchange and processing of data on SA needs and
3. making use of models and architectures of cognitive information processing from the field of Situation Awareness to advance and control the information collection and exchange for in-network processing.

For the first aspect, this thesis made use of the concept of situation parameters. The high level situations of interest can be decomposed into hierarchical parts and presented as situation parameters. The values for these parameters can in turn be produced by in-sensor data processing in individual nodes and composed into higher level situation parameters via in-network data processing algorithms within the network, close to the observed situation or physical phenomena of interest. The approach to decompose the high level situations into smaller parts, enables also to design much simpler algorithms for the individual low-cost network nodes. This composition of higher level situations already within the network, however, is not trivial, and is supported by the second aspect. For the second aspect, this thesis applies the concept of Data To Decision (D2D), which in turn is implemented through subscription controlled mediated interactions. The subscription controlled mediated interactions are used to relay only relevant, consistent and valid information which is required for SA, meaning both data collection and exchange involves only valid data that is used to generate relevant situational information driven by SA needs.

The third aspect involves using models and architectures from the field of SA. Two of the most well known models of SA are Mika Endsley's three level SA model and model of Distributed SA (DSA) provided by Nevil Stanton and Paul Salmon. In Endsley's three level model perception is a process which allows to distinguish relevant information for achieving situation awareness, comprehension in turn requires combination or integration of multiple perceived pieces (situation parameters) into a meaningful and operationally relevant information and projection is the highest level of Endsley's model of SA, it involves the ability to use the past situation parameters (memory) to estimate how they would evolve in near future. The DSA model by Nevil Stanton and Paul Salmon in turn describes SA as distributed situation awareness of a system that emerges through interactions between system components. This approach is well in line with the concepts developed in this thesis. One of the aspects addressed in the thesis is the changing topology and time variant delays of the environment where collection and exchange of situational information occurs. This necessitates the use of mediated interactions for creating consistent and valid SA. The more detailed analysis of the usability of SA models and architectures remains future work.

The thesis also defined the requirements that SA applications set for the SE and demonstrates through three main experiments that the proposed methods satisfy the given requirements. The proposed methods have also been validated with the customer during a WSN demonstrator carried out during a military exhibition Purple Nectar. This demonstrator is briefly described in the introduction of the thesis. The thesis also gave a short outlook for future applications for SE as described in 6 use cases that are under construction as of time of writing this thesis. Finally the thesis concludes with a number of suggestions for the enhancements of current middleware solution for future SE.

Applying mediated interactions could lead us one step closer to a SoS that can achieve its goals while adapting to changing environment. Such Sos is able to manage emergent behaviour, while adapting to (changing) user SA requirements and also while coping with the dynamically joining and leaving components and while at the same time continuously providing required services and successfully avoiding overloading the network with data collection tasks.

7.1 Future Research

This thesis has covered a wide range of topic while making use of a few different concepts from different disciplines, such as Situation Awareness, systems engineering, embedded computing, distributed computing, etc. This opens up also a multitudinous number of different challenges for future research. For example, the SmENeTe use cases described in 5.4.2 involve several interesting research challenges. Also the actual implementation of solutions described in current thesis in a WSN is far from trivial and leaves many technical challenges to be solved. To provide a small list of possible future research topics could be:

- Analysis and methods of estimation of time-variable end-to-end delays in mesh networks.
- Dynamic adaptivity to changing end-to-end delays by adjusting data processing and subscription rules with different contextual constraints.
- Analysis of sharing of memory buffering between producer nodes and consumer node while dynamically adapting to end-to-end delays.
- Methods to detect emergent behaviour as it is appearing, i.e., online in real time.
- Analysis of applicability of models and architectures known from the field of Situation Awareness for improving the quality of collection and exchange of situational information in Smart Environments.

The author of the thesis believes, that applying SA theories together with concept of me-diated interactions, based on a prototype of a multi-stream interaction centred model, for improving the Situation Awareness of both human and artificial agents via SE will offer many interesting challenges ahead.

List of Figures

List of Tables

References

[1] F. Adelantado, X. Vilajosana, P. Tuset-Peiro, B. Martinez, J. Melia-Segui, and T. Watteyne. Understanding the limits of lorawan. *IEEE Communications magazine*, 55(9):34–40, 2017.

[2] E. Ahmed, I. Yaqoob, A. Gani, M. Imran, and M. Guizani. Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 23(5):10–16, 2016.

[3] I. Al-Anbagi, M. Erol-Kantarci, and H. T. Mouftah. A survey on cross-layer quality-of-service approaches in wsns for delay and reliability-aware applications. *IEEE Communications Surveys & Tutorials*, 18(1):525–552, 2016.

[4] N. F. Ali, A. M. Said, K. Nisar, and I. A. Aziz. A survey on software defined network approaches for achieving energy efficiency in wireless sensor network. In *2017 IEEE Conference on Wireless Sensors (ICWiSe)*, pages 1–6. IEEE, 2017.

[5] P. W. Anderson. More is different. *Science*, 177(4047):393–396, 1972.

[6] C. André, J. DeAntoni, F. Mallet, and R. De Simone. The time model of logical clocks available in the omg marte profile. In *Synthesis of Embedded Software*, pages 201–227. Springer, 2010.

[7] S. Astapov, J. Berdnikova, J. Ehala, **J. Kaugerand**, and J.-S. Preden. Gunshot acoustic event identification and shooter localization in a wsn of asynchronous multichannel acoustic ground sensors. *Multidimensional Systems and Signal Processing*, 29(2):563–595, 2018.

[8] J. C. Baeten. A brief history of process algebra. *Theoretical Computer Science*, 335(2-3):131–146, 2005.

[9] A. J. Bahm. Five types of systems philosophy. *International Journal of General System*, 6(4):233–237, 1981.

[10] M. Bahrepour, N. Meratnia, and P. J. M. Havinga. sensor fusion-based event detection in wireless sensor networks. In *Proceedings of the 6th annual international mobile and ubiquitous systems: networking services (MobiQuitous)*, pages 1–8. Toronto, ON, Canada, July 2009.

[11] R. Bahroun, O. Michel, F. Frassati, M. Carmona, and J.-L. Lacoume. New algorithm for footstep localization using seismic sensors in an indoor environment. *Journal of Sound and Vibration*, 333(3):1046–1066, 2014.

[12] H. Barringer, A. Goldberg, K. Havelund, and K. Sen. Rule-based runtime verification. In *International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 44–57. Springer, 2004.

[13] J. Barwise and J. Perry. Situations and attitudes. *The Journal of Philosophy*, 78(11):668–691, 1981.

[14] A. Bauer, M. Leucker, and C. Schallhart. Model-based runtime analysis of distributed reactive systems. In *Australian Software Engineering Conference (ASWEC'06)*, pages 10–pp. IEEE, 2006.

[15] S. Beddar-Wiesing and M. Bieshaar. Multi-sensor data and knowledge fusion–a proposal for a terminology definition. *arXiv preprint arXiv:2001.04171*, 2020.

[16] G. Berry. Scade: Synchronous design and validation of embedded control software. In *Next Generation Design and Verification Methodologies for Distributed Embedded Control Systems*, pages 19–33. Springer, 2007.

[17] J. Boardman and B. Sauser. System of systems-the meaning of of. In *2006 IEEE/SMC International Conference on System of Systems Engineering*, pages 6–pp. IEEE, 2006.

[18] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu. Fog computing: A platform for internet of things and analytics. In *Big data and internet of things: A roadmap for smart environments*, pages 169–186. Springer, 2014.

[19] R. C. Carrano, D. Passos, L. C. Magalhaes, and C. V. Albuquerque. Survey and taxonomy of duty cycling mechanisms in wireless sensor networks. *IEEE Communications Surveys & Tutorials*, 16(1):181–194, 2014.

[20] P. Caspi and N. Halbwachs. A functional model for describing and reasoning about time behaviour of computing systems. *Acta Informatica*, 22(6):595–627, 1986.

[21] C.-T. Cheng, H. Leung, and P. Maupin. A delay-aware network structure for wireless sensor networks with in-network data fusion. *IEEE Sensors Journal*, 13(5):1622–1631, 2013.

[22] M. Clark, X. Koutsoukos, J. Porter, R. Kumar, G. Pappas, O. Sokolsky, I. Lee, and L. Pike. A study on run time assurance for complex cyber physical systems. Technical report, AIR FORCE RESEARCH LAB WRIGHT-PATTERSON AFB OH AEROSPACE SYSTEMS DIR, 2013.

[23] B. W. Clifton, S. Hostetler, and W. N. Felder. Mathematical models of emergence in complex systems-of-systems. In *Proc. 23rd INCOSE International Symposium*, 2013.

[24] S. Colin and L. Mariani. 18 run-time verification. In *Model-Based Testing of Reactive Systems*, pages 525–555. Springer, 2005.

[25] D. J. Cook, J. C. Augusto, and V. R. Jakkula. Ambient intelligence: Technologies, applications, and opportunities. *Pervasive and Mobile Computing*, 5(4):277–298, 2009.

[26] J. R. P. Cruz and S. E. P. Hernandez. Temporal alignment model for data streams in wireless sensor networks based on causal dependencies. *International Journal of Distributed Sensor Networks*, 10(3):938698, 2014.

[27] A. Czyżewski, J. Kotus, and G. Szwoch. Estimating traffic intensity employing passive acoustic radar and enhanced microwave doppler radar sensor. *Remote Sensing*, 12(1):110, 2020.

[28] M. J. de C Henshaw. Systems of systems, cyber-physical systems, the internet-of-things… whatever next? *Insight*, 19(3):51–54, 2016.

[29] T. De Wolf and T. Holvoet. Emergence versus self-organisation: Different concepts but promising when combined. In *International workshop on engineering self-organising applications*, pages 1–15. Springer, 2004.

[30] A. Dearle, D. Balasubramaniam, J. Lewis, and R. Morrison. A component-based model and language for wireless sensor network applications. In *Proceedings of the 32nd annual IEEE international conference on computer software and applications (COMPSAC'08)*, pages 1303–1308. Turku, Finland, 28 July-1 Aug. 2008.

[31] K. Devlin. Situation theory and situation semantics. In *Handbook of the History of Logic*, volume 7, pages 601–664. Elsevier, 2006.

[32] P. S. Dhake and S. S. Borde. Embedded surveillance system using pir sensor. *International Journal of Advanced Technology in Engineering and Science, www. ijates. com Volume*, (02), 2014.

[33] O. Diallo, J. J. Rodrigues, and M. Sene. Real-time data management on wireless sensor networks: A survey. *Journal of Network and Computer Applications*, 35(3):1013–1021, 2012.

[34] D. Djenouri and M. Bagaa. Synchronization protocols and implementation issues in wireless sensor networks: A review. *IEEE Systems Journal*, 10(2):617–627, 2016.

[35] B. M. Edmonds. *Syntactic measures of complexity*. University of Manchester Manchester, UK, 1999.

[36] J. Ehala, **J. Kaugerand**, R. Pahtma, S. Astapov, A. Riid, T. Tomson, J.-S. Preden, and L. Mõtus. Situation awareness via internet of things and in-network data processing. *International Journal of Distributed Sensor Networks*, 13(1):1–21, 2017.

[37] J. C. Eidson, E. A. Lee, S. Matic, S. A. Seshia, and J. Zou. Time-centric models for designing embedded cyber-physical systems. Technical report, CALIFORNIA UNIV BERKELEY DEPT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE, 2009.

[38] J. C. Eidson, E. A. Lee, S. Matic, S. A. Seshia, and J. Zou. Distributed real-time software for cyber–physical systems. *Proceedings of the IEEE*, 100(1):45–59, 2012.

[39] M. R. Endsley. Measurement of situation awareness in dynamic systems. *Human factors*, 37(1):65–84, 1995.

[40] M. R. Endsley. *Designing for situation awareness: An approach to user-centered design*. CRC press, 2016.

[41] M. R. Endsley. Toward a theory of situation awareness in dynamic systems. In *Situational awareness*, pages 9–42. Routledge, 2017.

[42] V. Estivill-Castro, R. Hexel, and J. Stover. Modeling, validation, and continuous integration of software behaviours for embedded systems. In *2015 IEEE European Modelling Symposium (EMS)*, pages 89–95. IEEE, 2015.

[43] W. Felder. Interactions among components in complex systems. *Hoboken, NJ: Stevens Institute of Technology, American Institute of Aeronautics and Astronautics*, pages 1–5, 2013.

[44] D. C. Foyle, A. D. Andre, and B. L. Hooey. Situation awareness in an augmented reality cockpit: Design, viewpoints and cognitive glue. In *Proceedings of the 11th International Conference on Human Computer Interaction*, volume 1, pages 3–9, 2005.

[45] P. B. FREng. On the nature of systems of systems. In *INCOSE International Symposium*, volume 26, pages 1477–1493. Wiley Online Library, 2016.

[46] D. Goldin and D. Keil. Interaction, evolution, and intelligence. In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, volume 2, pages 805–814. IEEE, 2001.

[47] C. Gomez, S. Chessa, A. Fleury, G. Roussos, and D. Preuveneers. Internet of things for enabling smart environments: A technology-centric perspective. *Journal of Ambient Intelligence and Smart Environments*, 11(1):23–43, 2019.

[48] A. E. Goodloe and L. Pike. Monitoring distributed real-time systems: A survey and future directions. *NASA STI Report Series*, 2010.

[49] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[50] N. Halbwachs. A synchronous language at work: the story of lustre. In *Proceedings. Second ACM and IEEE International Conference on Formal Methods and Models for Co-Design, 2005. MEMOCODE'05.*, pages 3–11. IEEE, 2005.

[51] M. Henshaw, C. Siemieniuch, M. Sinclair, S. Henson, V. Barot, M. Jamshidi, D. Delaurentis, C. Ncube, S. L. Lim, and H. Dogan. Systems of systems engineering: a research imperative. In *2013 International Conference on System Science and Engineering (ICSSE)*, pages 389–394. IEEE, 2013.

[52] A. Hiromori, A. Uchiyama, H. Yamaguchi, and T. Higashino. Deadline-aware data collection in csma/ca-based multi-sink wireless sensor networks. In *Proceedings of the 6th international conference on mobile computing and ubiquitous networking, ICMU*, volume 12, pages 284–294. Fukuoka, Japan, 2012.

[53] I. Horváth. What the design theory of social-cyber-physical systems must describe, explain and predict? In *An Anthology of Theories and Models of Design*, pages 99–120. Springer, 2014.

[54] D. Izadi, J. H. Abawajy, S. Ghanavati, and T. Herawan. A data fusion method in wireless sensor networks. *Sensors*, 15(2):2964–2979, 2015.

[55] J. K. Jacoub, R. Liscano, and J. S. Bradbury. A survey of modeling techniques for wireless sensor networks. In *Proceedings of the 5th international conference on sensor technologies and applications (SENSORCOMM)*, volume 2011, pages 103–109. Nice, France, 2011.

[56] G. Jakobson. A framework for cognitive situation control. In *2017 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, pages 1–7. IEEE, 2017.

[57] A. Jalal, S. Kamal, and D. Kim. A depth video sensor-based life-logging human activity recognition system for elderly care in smart indoor environments. *Sensors*, 14(7):11735–11759, 2014.

[58] A. Jhumka, M. Bradbury, and S. Saginbekov. Efficient fault-tolerant collision-free data aggregation scheduling for wireless sensor networks. *Journal of Parallel and Distributed Computing*, 74(1):1789–1801, 2014.

[59] H. Jo, S.-H. Sim, K. A. Mechitov, R. Kim, J. Li, P. Moinzadeh, B. Spencer Jr, J. W. Park, S. Cho, H.-J. Jung, et al. Hybrid wireless smart sensor network for full-scale structural health monitoring of a cable-stayed bridge. In *Sensors and Smart Structures Technologies for Civil, Mechanical, and Aerospace Systems 2011*, volume 7981, page 798105. International Society for Optics and Photonics, 2011.

[60] F. T. Johnsen, T. H. Bloebaum, and P. Puente. Towards friendly force tracking with mqtt over lora. 2019.

[61] F. T. Johnsen, Z. Zieliński, K. Wrona, N. Suri, C. Fuchs, M. Pradhan, J. Furtak, B. Vasilache, V. Pellegrini, M. Dyk, et al. Application of iot in military operations in a smart city. In *2018 International Conference on Military Communications and Information Systems (ICMCIS)*, pages 1–8. IEEE, 2018.

[62] C. W. Johnson. What are emergent properties and how do they affect the engineering of complex systems? *Reliability Engineering and System Safety*, 91(12):1475–1481, 2006.

[63] P. Joshi, G. Singh, and A. Raghuvanshi. Comparative study of different routing protocols for ieee 802.15. 4-enabled mobile sink wireless sensor network. In *Advances in VLSI, Communication, and Signal Processing*, pages 161–170. Springer, 2020.

[64] **J. Kaugerand**, J. Ehala, L. Mõtus, and J.-S. Preden. Time-selective data fusion for in-network processing in ad hoc wireless sensor networks. *International Journal of Distributed Sensor Networks*, 14(11):1–17, 2018.

[65] **J. Kaugerand**, J.-S. Preden, E. Suurjaak, S. Astopov, L. Motus, and R. Pahtma. A system of systems solution for perimeter control: Combining unmanned aerial system with unattended ground sensor network. In *Systems Conference (SysCon), 2015 9th Annual IEEE International*, pages 317–323. IEEE, 2015.

[66] D. Keil and D. Goldin. Modeling indirect interaction in open computational systems. In *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.*, pages 371–376. IEEE, 2003.

[67] B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi. Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28 – 44, 2013.

[68] A. Kobetski and J. Axelsson. Federated embedded systems–a review of the literature in related fields, 2012.

[69] R. K. Kodali and A. Sahu. An iot based weather information prototype using wemos. In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, pages 612–616. IEEE, 2016.

[70] P. Kolar, N. Gamez, and M. Jamshidi. Impact of time delays on networked control of autonomous systems. In *Beyond Traditional Probabilistic Data Processing Techniques: Interval, Fuzzy etc. Methods and Their Applications*, pages 151–178. Springer, 2020.

[71] H. Kopetz. *Real-Time Systems: Design Principles for Distributed Embedded Applications*. Springer Science & Business Media, 2011.

[72] J. Kramer. Distributed software engineering. In *Proceedings of 16th International Conference on Software Engineering*, pages 253–263. IEEE, 1994.

[73] A. Kumar and D. Singh. Importance of energy in wireless sensor networks: A survey. *An International Journal of Engineering Sciences*, 17:500–505, 2016.

[74] S. Lai, J. Cao, and X. Fan. Ted: Efficient type-based composite event detection for wireless sensor network. In *Proceedings of the 11th international conference on distributed computing in sensor systems (DCOSS)*, pages 1–8. Barcelona, Spain, June 2011.

[75] D. A. Lambert. A blueprint for higher-level fusion systems. *Information Fusion*, 10(1):6–24, 2009.

[76] S. M. Lasassmeh and J. M. Conrad. Time synchronization in wireless sensor networks: A survey. In *Proceedings of the IEEE SoutheastCon 2010 (SoutheastCon)*, pages 242–245. IEEE, 2010.

[77] E. A. Lee. Cps foundations. In *Design Automation Conference*, pages 737–742. IEEE, 2010.

[78] E. A. Lee. The past, present and future of cyber-physical systems: A focus on models. *Sensors*, 15(3):4837–4869, 2015.

[79] J. Lee, B. Bagheri, and H.-A. Kao. A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters*, 3:18–23, 2015.

[80] S.-Y. Lee and F. J. Lin. Situation awareness in a smart home environment. In *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, pages 678–683. IEEE, 2016.

[81] M. Leucker and C. Schallhart. A brief account of runtime verification. *The Journal of Logic and Algebraic Programming*, 78(5):293–303, 2009.

[82] H. Levesque, F. Pirri, and R. Reiter. Foundations for the situation calculus, 1998.

[83] Y. Li and R. Bartos. A survey of protocols for intermittently connected delay-tolerant wireless sensor networks. *Journal of Network and Computer Applications*, 41:411–423, 2014.

[84] Y. Liu, X. Wang, Y. Liu, and S. Cui. Data-aware retrodiction for asynchronous harmonic measurement in a cyber-physical energy system. *Sensors*, 16(8):1316, 2016.

[85] M. Lohstroh and E. A. Lee. An interface theory for the internet of things. In *SEFM 2015 Collocated Workshops*, pages 20–34. Springer, 2015.

[86] C. MacGillivray, C. Macgillivray, and D. Reinsel. Worldwide global datasphere iot device and data forecast, 2019-2023. Technical report, International Data Corporation (IDC), (IDC #US45066919), May 2019.

[87] T. W. Malone and M. S. Bernstein. *Handbook of collective intelligence*. MIT Press, 2015.

[88] M. Maroti and J. Sallai. Packet-level time synchronization. *TinyOS Core Working Group, Technical Report, May*, 2008.

[89] I. Mayk, W. Regli, D. Nguyen, M. Mai, A. Chan, T. Urness, B. Goren, S. Kizenko, S. Randles, Z. Jastrebski, et al. Net-centric information and knowledge management and dissemination for data-to-decision c2 applications using intelligent agents and service-oriented architectures. In *2011-MILCOM 2011 Military Communications Conference*, pages 1568–1573. IEEE, 2011.

[90] E. Mayr. *The growth of biological thought: Diversity, evolution, and inheritance*. Harvard University Press, 1982.

[91] I. C. Motorola and C. Perkins. Dynamic manet on-demand (dymo) routing. *draft-ietf-manet-dymo-12*, 2008.

[92] L. Motus, J. Ehala, J. Preden, and J. Kaugerand. Attaining self-awareness in cyber-physical systems. A poster presented at 2016 CPS Week 2016, Workshop on Self-Awareness in Cyber-Physical Systems (SelPhys), $selphys.ict.tuwien.ac.at/2016/src/P\_LeoMotus.pdf$, last visited at 2020.04.05, 2016.

[93] L. Motus, M. Meriste, and W. Dosch. Time-awareness and proactivity in models of interactive computation. *Electronic Notes in Theoretical Computer Science*, 141(5):69–95, 2005.

[94] L. Mõtus, M. Meriste, and J.-S. Preden. Towards middleware based situation awareness. In *Proceedings of the IEEE Military Communications Conference (MILCOM)*, pages 1–7. IEEE, Boston, MA, USA, Oct 2009.

[95] L. Motus, J.-S. Preden, M. Meriste, and R. Pahtma. Self-aware architecture to support partial control of emergent behavior. In *2012 7th International Conference on System of Systems Engineering (SoSE)*, pages 422–427. IEEE, 2012.

[96] L. Motus and M. G. Rodd. *Timing analysis of real-time software*. Elsevier, 1994.

[97] L. Motus, M. Teichmann, T. Kangilaski, J. Priisalu, and **J. Kaugerand**. Some issues in modelling comprehensive situation awareness. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 540–545. IEEE, 2019.

[98] C. Ncube. On the engineering of systems of systems: key challenges for the requirements engineering community. In *2011 Workshop on Requirements Engineering for Systems, Services and Systems-of-Systems*, pages 70–73. IEEE, 2011.

[99] R. S. Oliver and G. Fohler. Probabilistic estimation of end-to-end path latency in wireless sensor networks. In *Proceedings of the IEEE 6th international conference on mobile adhoc and sensor systems (MASS'09)*, pages 423–431. Macau, China, 2009.

[100] A. Padovitz, S. W. Loke, A. Zaslavsky, B. Burg, and C. Bartolini. An approach to data fusion for context awareness. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 353–367. Springer, 2005.

[101] H. V. D. Parunak, R. S. VanderBok, et al. Managing emergent behavior in distributed control systems. *Proceedings of ISA Tech*, 97:1–8, 1997.

[102] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos. Context aware computing for the internet of things: A survey. *IEEE communications surveys & tutorials*, 16(1):414–454, 2013.

[103] J. Perry. Semantics, possible worlds. *Routledge Encyclopedia of Philosophy. London: Routledge. URL http://www. rep. routledge. com/article U*, 39, 1998.

[104] R. Pew and A. Mavor. Modelling human and organizational behaviour: Application to military simulation, 1998.

[105] F. Polack and S. Stepney. Emergent properties do not refine. *Electronic Notes in Theoretical Computer Science*, 137(2):163–181, 2005.

[106] M. A. Pradhan, S. Patankar, A. Shinde, V. Shivarkar, and P. Phadatare. Iot for smart city: Improvising smart environment. In *2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS)*, pages 2003–2006. IEEE, 2017.

[107] J. Preden, J. Llinas, G. Rogova, R. Pahtma, and L. Motus. On-line data validation in distributed data fusion. In *Proceedings of the international conference on SPIE Defense, Security, and Sensing*, volume 8742, pages 87420S–87420S. International Society for Optics and Photonics, Baltimore, Maryland, United States, 2013.

[108] J. Preden, L. Motus, M. Meriste, and A. Riid. Situation awareness for networked systems. In *2011 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA)*, pages 123–130. IEEE, 2011.

[109] J.-S. Preden, **J. Kaugerand**, E. Suurjaak, S. Astapov, L. Motus, and R. Pahtma. Data to decision: pushing situational information needs to the edge of the network. In *Proceedings of the 2015 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision*, pages 158–164. New York: IEEE, Orlando, FL, USA, 9-12 March 2015.

[110] J.-S. Preden, J. Llinas, and L. Motus. Middleware for exchange and validation of context data and information. In *Context-Enhanced Information Fusion*, pages 205–230. Springer, 2016.

[111] J.-S. Preden, L. Motus, R. Pahtma, and M. Meriste. Data Exchange for Shared Situation Awareness. In *Proceedings of the 2012 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision support (CogSIMA)*, pages 198–201. 2012.

[112] J. S. Preden, K. Tammemäe, A. Jantsch, M. Leier, A. Riid, and E. Calis. The benefits of self-awareness and attention in fog and mist computing. *Computer*, 48(7):37–45, 2015.

[113] J.-S. Preden and M. Uusjärv. Device and method for a multi-hop mobile ad hoc network, Feb. 24 2015. US Patent 8,966,046.

[114] W. Quirk. The automatic analysis of formal real-time system specifications. Technical report, CM-P00068596, 1978.

[115] R. Rajagopalan and P. K. Varshney. Data aggregation techniques in sensor networks: A survey. 2006.

[116] M. V. Ramesh. Design, development, and deployment of a wireless sensor network for detection of landslides. *Ad Hoc Networks*, 13:2–18, 2014.

[117] I.-K. Rhee, J. Lee, J. Kim, E. Serpedin, and Y.-C. Wu. Clock synchronization in wireless sensor networks: An overview. *Sensors*, 9(1):56–85, 2009.

[118] C. Richards. Boyd's ooda loop. *JV/M, Inc*, pages 1–5, 2012.

[119] A. Riid, **J. Kaugerand**, J. Ehala, M. Jaanus, and J.-S. Preden. An application of a low-cost microwave radar to traffic monitoring. In *2018 16th Biennial Baltic Electronics Conference (BEC)*, pages 1–4. IEEE, 2018.

[120] P. M. Salmon, N. A. Stanton, and D. P. Jenkins. *Distributed situation awareness: Theory, measurement and application to teamwork*. CRC Press, 2017.

[121] R. G. Sands. William i thomas, the unadjusted girl, 2014.

[122] I. Sanduka, T. Lochow, and R. Obermaisser. Runtime fault prediction and prevention for emerging services in system of systems. In *CSDM (Posters)*, pages 139–152, 2013.

[123] R. Saracco. Digital twins: Bridging physical space and cyberspace. *Computer*, 52(12):58–64, 2019.

[124] M. A. Sarvghadi and T.-C. Wan. Message passing based time synchronization in wireless sensor networks: a survey. *International Journal of Distributed Sensor Networks*, 12(5):1280904, 2016.

[125] M. Satyanarayanan. The emergence of edge computing. *Computer*, 50(1):30–39, 2017.

[126] R. Savimaa. Integrating uml, the q-model and a multi-agent approach in process specifications and behavioural models of organisations. *Journal of Systemics, Cybernetics and Informatics*, pages 45–57, 2006.

[127] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *1994 First Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE, 1994.

[128] E. Serpedin and Q. M. Chaudhari. *Synchronization in wireless sensor networks: parameter estimation, performance benchmarks, and protocols*. Cambridge University Press, 2009.

[129] Z. Shelby and C. Bormann. *6LoWPAN: The wireless embedded Internet*, volume 43. John Wiley & Sons, 2011.

[130] Y. Shi, Y. Wan, and Q. Ge. A unified out-of-sequence measurements fusion algorithm for wsn. In *Proceedings of the 1st international workshop on database technology and applications*, pages 76–79. Wuhan, Hubei, China, 2009.

[131] N. A. Stanton. Distributed situation awareness, 2016.

[132] S. Sun, H. Lin, J. Ma, and X. Li. Multi-sensor distributed fusion estimation with applications in networked systems: A review paper. *Information Fusion*, 38:122–134, 2017.

[133] G. Tanzmeister and S. Steyer. Spatiotemporal alignment for low-level asynchronous data fusion with radar sensors in grid-based tracking and mapping. In *2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 231–237. IEEE, 2016.

[134] N. Tarafder. Applications of wearable electronics as smart clothing. *Man-Made Textiles in India*, 46(5), 2018.

[135] L. Wang, H. Wang, Q. Yu, H. Sun, and A. Bouguettaya. Online reliability time series prediction for service-oriented system of systems. In *International Conference on Service-Oriented Computing*, pages 421–428. Springer, 2013.

[136] Y. Wang, M. C. Vuran, and S. Goddard. Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. *IEEE/ACM Transactions on networking*, 20(1):305–318, 2012.

[137] M. Winkler, K.-D. Tuchs, K. Hughes, and G. Barclay. Theoretical and practical aspects of military wireless sensor networks. *Journal of Telecommunications and Information Technology*, pages 37–45, 2008.

[138] Y.-C. Wu, Q. Chaudhari, and E. Serpedin. Clock synchronization of wireless sensor networks. *IEEE Signal Processing Magazine*, 28(1):124–138, 2010.

[139] F. Xiaoliang, W. Chenglin, and X. Lizhong. Multi-sensor multi-oosm distributed sequential fusion filtering. In *Proceedings of the 10th IEEE international conference on control and automation (ICCA)*, pages 77–776. Hangzhou, China, 2013.

[140] S. S. Yadav, A. Chitra, and C. L. Deepika. Reviewing the process of data fusion in wireless sensor network: a brief survey. *International Journal of Wireless and Mobile Computing*, 8(2):130–140, 2015.

[141] G. Zhao. Wireless sensor networks for industrial process monitoring and control: a survey. *Network Protocols and Algorithms*, 3(1):46–63, 2011.

Acknowledgements

I would like to take this opportunity to thank all those who made this work possible. First, I would like to thank my supervisor Dr. Jürgo-Sören Preden for all the support and guidance throughout my research. His trust in me and his excellent advice throughout my doctoral studies made this thesis possible. I would also like to express my gratitude to my research colleagues at the Research Laboratory for Proactive Technologies. Especially professor Leo Mõtus for his inspiring scientific guidance, Mr Johannes Ehala for many interesting and challenging theoretical discussions and Mr Raido Pahtma for his help and assistance with application of methods developed in this thesis and experiments. Also I would like to thank, Dr. Andri Riid, Mr Sergei Astapov and Mr Erki Suurjaak for assistance in conducting the field experiments.

Last but not least, I thank my family for their unlimited patience without which, writing of this thesis would not have been possible.

This work was partly supported by the European Defence Agency project IN4STARS 2.0, by the project SmENeTe2 (Smart Environment Networking Technologies), which was funded by the European Regional Development Fund within the framework of EU Smart Specialisation programme 2014–2020, by the two grants from Estonian Research Council: Infotechnological Mobility Observatory and Manticus Apollo and by the Estonian IT Academy program.

Abstract

Mediated interactions for collection and exchange of situational information in Smart Environments

Embedded computation and communication technologies for IoT and Smart Environments have developed very rapidly in recent years and have become capable to perform complex computational tasks needed for collecting and processing situational information already within the network. The applications for such tasks, implemented already at the edge within Smart Environments, are characterised by persistent dynamical update and stream-like processing of data. This in turn requires methodical consideration of contextual (e.g. temporal and spatial) validity and consistency. In order to ensure the quality of the situational information, this thesis utilises a concept of mediated interactions. This concept allows on-line data validation and novel selective delivery and processing to improve consistency of exchanged data. The thesis points out that in Internet of Things solutions, which may rely on distributed wireless ad hoc and mesh sensor networks, especially those, where global clock synchronisation is not either feasible or even possible, the data that is used for in-network data processing algorithms may not always be temporally consistent. This problem is particularly clear in systems where, for simplicity, data is used in the order of arrival. The different communication paths may have different and time variable delays, for example due to changing topology, changing network load or different network nodes getting access to the common communication channel at different times. Applying concept of mediated interactions allows viewing each communication channel at an abstract level as an intelligent mediator agent with a task to ensure data validity and that only mutually consistent and compatible data is selected from the streams as inputs for situational information processing. This intelligent mediator agent is implemented as a middleware software layer at each network node. In this way, the rules for validity and consistency can be configured separately for each data stream according to the situational information requirements. The corresponding validation and consistency checking tasks are carried out online and information is conveyed to the consumer only through valid situation parameters. In software, the channel, or smart mediator agent, is implemented as a middleware called ProWare.

Situation Awareness applications based on ProWare technology can be used in ad hoc mesh networks where data flows between autonomous and distributed sources can be created within the network without prior fixed topology. In order to validate data in the network on-line, the data is augmented with respective contextual metadata. For example, spatial metadata is determined according to the position of the observed phenomena and temporal metadata is computed by each node based on age of data and/or information being processed in the distributed network. The same contextual metadata is also used for selective access to data in order to ensure mutual contextual compatibility for identification of situations.

The network nodes exchange situational data between each other based on dynamic contracts called subscriptions. Any piece of information processed according to the situational information subscription is called a situation parameter. These parameters are produced, stored and exchanged by distributed network nodes that can be also viewed as intelligent and autonomous agents. Combining situation parameters through mediated interactions enables to compute situations and their parameters at a higher abstraction level. In the context of Distributed Situation Awareness, it can be said that the higher level or rather system level Situational Awareness emerges trough interactions by the agents who possess local relevant situational information. Computing higher-level parameters

already within the network is done via in-network data processing. In alternative architectures all situation parameters are communicated to an application on the cloud server, which in turn performs the corresponding sensor fusion or aggregation task, but this approach has many limitations, including the throughput of communication channels, temporal alignment of data and dynamic adaptivity.

Situation Awareness applications differ from conventional data collection in that firstly only relevant information is collected from the environment and secondly it must be dynamically updated. This type of information can, in principle, be consumed for decision-making to foster rapid solution of many issues in the fields of transport, communication, security and many others. The dynamical renewability of collected situational information also means that both its validity and consistency should be checked on-line. The concept of Situation Awareness assumes that the system user (human or artificial agent) has access to the correct information under the right circumstances to understand and manage the situation he, she or it is interested in. In temporal and spatial context, the right circumstances means that information stemming from the right location, collected at the right time is delivered at the right time interval to the right location. This is achieved by defining and adding validity metadata (e.g., temporal and spatial tags) to any produced piece of information and checking these tags against requirements imposed on the collected situational information, most likely defined by the user depending on the characteristics of the situation. The collection, exchange and processing of situational information is goal-driven, driven by SA needs.

The theoretical contribution of the dissertation is researching and supplementing the concept of mediated interactions with the purpose of applying it in Smart Environments for collecting situational information. The main contributions are the introduction of methodology for achieving context-based consistency of input data from distributed sources and a technical solution for on-line checking of the validity and consistency of data required for processing situational information. The dissertation presents the results of a number of experiments described also in articles published by the author of the dissertation which demonstrate the practical application of the aforementioned theoretical work. The experiments have been performed using wireless sensor network technologies in military and urban environments. The results of the experiments show both the applicability of the developed methods and a significant improvement in the quality of produced situational information.

Kokkuvõte
Vahendatud interaktsioonid olukorrateadlikkuse informatsiooni kogumiseks ja vahetamiseks arukates keskkondades

Arukate keskkondade ja asjade interneti jaoks vajalikud sard- ja kommunikatsioonitehnoloogiad on viimastel aastatel väga kiiresti arenenud ning on muutunud võimeliseks täitma olukorrateabe kogumiseks ja võrgusiseseks töötlemiseks vajalikke keerulisi arvutusülesandeid. Vastavaid olukorrateadlikkuse rakendusi iseloomustab vajadus andmeid püsivalt ja dünaamiliselt värskendada ning käsitleda ja töödelda andmevoogudena. See omakorda tähendab, et hajutatud sensorsüsteemidega kogutud olukorrateabe kvaliteedi tagamiseks on vaja metoodiliselt arvestada kogutud andmete kontekstipõhist (nt ajalise ja ruumilise) valiidsust ja kooskõlalisust. Käesolev väitekiri kasutab selleks vahendatud interaktsioonide kontseptsiooni. Aruka keskkonna sensorvõrgu täiendamine vahendatud interaktsioonidega loob võimaluse valideerida andmeid hajutatud võrgusõlmedes reaalajas ning valikuliselt edastada ja töödelda ainult valideeritud andmeid. Käesolev töö toob välja, et hajutatud traadita spontaanvõrkudes, eriti sellistes, kus globaalne sünkroniseerimine on ressursside kasutuse mõistes väga kulukas või isegi võimatu, ei pruugi võrgusisestes andmetöötlusalgoritmides kasutatavad andmed alati olla kokkusobivad. Eriti selge on see probleem süsteemides, kus andmeid kasutatakse lihtsuse mõttes nende saabumise järjekorras. Hajutatud traadita spontaanvõrkudes võivad erinevatel kommunikatsiooniteedel olla erinevad ja ajas muutuvad hilistumised. Seda põhjustab näiteks muutuv topoloogia, andmeside kanalite koormuse muutumine või ajaliselt jagatud ühise sidekanali kasutamine. Vahendatud interaktsioonide kontseptsiooni kasutamine võimaldab vaadelda iga kommunikatsiooni kanalit kui intelligentset vahendaja-agenti, mille ülesanne on reaalajas tagada nii andmete valiidsus kui ka vastastikku kooskõlalised ehk ühilduvad andmed. Tarkvaraliselt on see vahendaja-agent rakendatud igas võrgusõlmes kui vahevara moodul. Vastavaid reegleid, mille alusel andmete valideerimine ja sobivate andmete valik toimub, on iga andmevoo jaoks võimalik erinevalt konfigureerida. Nii edastatakse olukorrateabe töötlemise protsesside sisenditeks ainult kehtivad ja omavahel kooskõlalised andmed. Intelligentse vahendajaagendi tarkvara nimetatakse ProWare vahevaraks.

Kasutades ProWare tehnoloogiat saab luua olukorrateadlikkuse rakendusi silmusvõrktopoloogiaga võrkudes, kus andmevoogusi edastatakse ja vahendatakse autonoomsete ja hajutatud allikate vahel võrgus sees ilma eelnevalt fikseeritud topoloogiata. Selleks et andmeid sellises võrgus reaalajas valideerida, laiendatakse olukorrateabe loomiseks vajalikud sensorandmed kontekstipõhiste metaandmetega. Näiteks ruumilised metaandmed määratakse jälgitava olukorra või sündmuse asukoha järgi ning ajalised metaandmed lisatakse kõigepealt andmete tootja poolt vastavalt huvipakkuva olukorra või sündmuse teatele ja seejärel iga võrgusõlme poolt arvestades andmete vanust nende transpordil ja töötlemisel läbi võrgu. Samamoodi kasutatakse kontekstipõhiseid metaandmeid ka andmete selektiivsel valikul andmevoogudest, selleks et tagada olukordade kirjeldamiseks kasutatavate andmete valiidsus ja omavaheline kontekstipõhine kokkusobivus ehk kooskõlalisus.

Võrgusõlmed vahetavad andmeid kasutades dünaamilisi lepinguid, mida nimetatakse tellimusteks. Metaandmetega täiendatud sensorandmeid, mida töödeldakse vastavalt olukorrateabe tellimusele, nimetatakse olukorra parameetriteks. Neid parameetreid toodavad, säilitavad ja vahetavad hajutatud võrgusõlmed. Olukorraparameetrite omavaheline kombineerimine läbi vahendatud interaktsioonide võimaldab juba võrgus sees arvutada kõrgema abstraktsiooni taseme parameetreid. Alternatiivsetes arhitektuurides edastatakse kõik olukorraparameetrid pilveserveris asuvasse rakendusse, mis omakorda täidab vastava teabe integratsiooni ja analüüsi ülesande, kuid arukates keskkondades, kus kasutatakse sensorplatvormidena odavaid sardsüsteeme ja traadita sensorvõrke, on mitmed piirangud, näiteks kommunikatsioonikanalite läbilaskevõime, andmete ajaline joondamine ja süsteemi dünaamiline adaptiivsus, mis ei võimalda sellise lahenduse kasutamist.

Olukorrateadlikkuse rakendused erinevad tavapärasest andmekogumisest selle poolest, et ümbritsevast keskkonnast kogutakse ainult olulist teavet ning seda uuendatakse dünaamiliselt. Kogutavat olukorrateavet saab kasutada reaalaja otsuste tegemiseks paljudes valdkondades nagu transpordi-, kommunikatsiooni-, turvalisuse valdkondades, kus vajatakse olukordade kiiret lahendamist. Olukorrateadlikkuse mõiste eeldab, et süsteemi kasutajale (inimene või tehislik agent), kelle eesmärk on mõista ja hallata teda huvitavaid olukordi, on ligipääs õigele teabele ja õigetel asjaoludel. Ajalises ja ruumilises kontekstis tähendab õiged asjaolud, et teave on õigest kohast kogutud, õigel ajal kogutud ja tarnitakse õigel ajal õigele kasutajale. See saavutatakse määratledes ja lisades igale toodetud olukorraparameetrile kontekstipõhine kehtivuse metaandmestik ning kontrollides seda olukorrateabe kogumisele seatud nõuete suhtes, mille kasutaja on määratlenud sõltuvalt olukorra iseloomust. Olukorrateabe kogumine, vahetamine ja töötlemine on eesmärgipõhine, juhitud olukorrateadlikkuse vajadustest.

Väitekirja teoreetilise panusena võib nimetada vahendatud interaktsioonide kontseptsiooni uurimist ja täiendamist eesmärgiga seda rakendada arukates keskkondades olukorrateadlikkuse informatsiooni kogumiseks. Peamiste panustena töötatakse väitekirjas välja metoodika hajutatud allikate sisendandmete kontekstipõhise kooskõla saavutamiseks ja tehniline lahendus reaalajas olukorrateabe töötluseks vajalike sisendandmeteandmete nii valiidsuse kui ka kooskõla kontrolliks. Lisaks kajastab väitekiri mitmes väitekirja autori avaldatud artiklis kirjeldatud eksperimente, mis demonstreerivad eelnimetatud teoreetilise tulemuse praktilist rakendamist. Eksperimendid on läbi viidud kasutades traadita sensorvõrgu tehnoloogiat, militaar- ja linnakeskkondades. Eksperimentide tulemused näitavad nii väljatöötatud meetodite rakendatavust kui ka olulist olukorrateabe kvaliteedi tõusu.

Appendix 1

I

J. Kaugerand, J. Ehala, L. Mõtus, and J.-S. Preden. Time-selective data fusion for in-network processing in ad hoc wireless sensor networks. *International Journal of Distributed Sensor Networks*, 14(11):1–17, 2018

International Journal of Distributed Sensor Networks

International Journal of Distributed Sensor Networks
2018, Vol. 14(11)
© The Author(s) 2018
DOI: 10.1177/1550147718811302
journals.sagepub.com/home/dsn
SAGE

Time-selective data fusion for in-network processing in ad hoc wireless sensor networks

Jaanus Kaugerand[1], Johannes Ehala[1], Leo Mõtus[1] and Jürgo-Sören Preden[2]

Abstract

This article introduces a time-selective strategy for enhancing temporal consistency of input data for multi-sensor data fusion for in-network data processing in ad hoc wireless sensor networks. Detecting and handling complex time-variable (real-time) situations require methodical consideration of temporal aspects, especially in ad hoc wireless sensor network with distributed asynchronous and autonomous nodes. For example, assigning processing intervals of network nodes, defining validity and simultaneity requirements for data items, determining the size of memory required for buffering the data streams produced by ad hoc nodes and other relevant aspects. The data streams produced periodically and sometimes intermittently by sensor nodes arrive to the fusion nodes with variable delays, which results in sporadic temporal order of inputs. Using data from individual nodes in the order of arrival (i.e. freshest data first) does not, in all cases, yield the optimal results in terms of data temporal consistency and fusion accuracy. We propose time-selective data fusion strategy, which combines temporal alignment, temporal constraints and a method for computing delay of sensor readings, to allow fusion node to select the temporally compatible data from received streams. A real-world experiment (moving vehicles in urban environment) for validation of the strategy demonstrates significant improvement of the accuracy of fusion results.

Keywords

Time-selective data fusion, in-network data processing, data validity, data alignment, data simultaneity, wireless ad hoc sensor networks, situation awareness, middleware

Date received: 31 January 2018; accepted: 10 October 2018

Handling Editor: Jose Molina

Introduction

Detecting complex situations in real time typically requires simultaneous observations originating from several autonomous and multi-modal wireless sensor network (WSN) nodes. The problem, however, when employing ad hoc WSNs, is that the data transport times of even simultaneous sensor readings, acquired by distributed nodes, may not match temporarily even if the same communication path is used. This article focuses on communication between sensor and fusion processes, explores how readings from multiple distributed sensor nodes are consumed by the fusion node in real time and how data validity and

[1]Research Laboratory for Proactive Technologies, Tallinn University of Technology, Tallinn, Estonia
[2]Thinnect, Inc., Sunnyvale, CA, USA

Corresponding author:
Jaanus Kaugerand, Research Laboratory for Proactive Technologies, Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia.
Email: jaanus.kaugerand@ttu.ee

simultaneity intervals affect the selection of temporally matching data for fusion process.

The purpose of the information produced by the WSN nodes at the edge of the network is to cater for the needs of the data users[1] deeper in the WSN. When used for situation awareness (SA) applications, the credibility of such information depends on timely processing of sensor data within the network and the temporal validity of data used.[2,3] The users subscribe to situational information of interest, not from a central server but directly from nodes performing in-network data processing,[4] which are able to provide the requested SA information. The in-network processing nodes in turn subscribe to data from sensor nodes. The subscription contains information about what data should be provided, its expected refresh rate and can also specify the requirements for validity and simultaneity intervals. The WSN middleware for handling the subscriptions for exchanging SA information has been introduced in our previous work.[5]

In this article, we consider the communication between asynchronous WSN nodes – meaning that clocks in different nodes are not synchronized and the start-up, data production and consumption processes in the nodes are activated independently from each other. The nodes in the network may employ different operating modes or duty cycling schemes and incorporate several heterogeneous sensors, with different modalities, characteristics and sampling frequencies. When a sensor node receives a subscription, it activates a periodic or event-based process for data production, the parameters of the process being dependent on the details of the subscription. Each WSN node may simultaneously service multiple active subscriptions and respectively run several processes for data production or consumption. The data produced by periodic execution of sensor processes (as subscribed by fusion processes) form data streams which can be intermittent[4] with elements not uniformly distributed in time with the possibility of some elements being sporadically delayed due to behavioural pattern of ad hoc WSN. As a result, some of the stream data elements may violate the required validity periods,[3] arrive out of order[6,7] and can often have only partial temporal coverage.[8] This behaviour could be caused by several factors, such as the combined effect from the application of low data rate communication standard (e.g. IEEE 802.15.4, Bluetooth Low Energy or proprietary standards), ad hoc nature of WSN and an unpredictable, volatile – that is, disconnected, intermittent and low-bandwidth (DIL) – communication environment,[9] where WSN nodes operate. Therefore, the sensor readings used in the order of arrival as inputs for in-network data fusion and aggregation processes may not characterize the same situation. Hence, using always the freshest data from available streams may not be desirable. We suggest that only temporally and spatially compatible data should be fused and/or combined to infer new synthesized readings, and we propose a strategy for selection of temporally suitable data for improving the temporal consistency of input data for in-network processing. The strategy, implemented in the WSN fusion nodes' middleware component as a default service, combines the use of temporal constraints[4] with a temporal alignment and selection algorithm and is customized for processing multiple streams of sensor data, which can be intermittent and arrive out of order. The mechanisms for data alignment, selection of suitable input data and verifying them against validity and simultaneity constraints are described using Q-modelling formalism.[10] Q-model allows to model the data streams in distributed systems (e.g. WSNs) and to analyse the delays of stream elements caused by periodic or sporadic activations of asynchronous processes in distributed systems (e.g. WSN nodes). As opposed to other methods that use time constraints and prefer freshest data first for time-sensitive WSN applications, the time-selective strategy allows fusion node to purposefully select temporally compatible data items from input streams.

An urban traffic monitoring experiment is used to demonstrate the enhancement of temporal consistency of input data for in-network data processing and a respective significant improvement in accuracy of data fusion results.

Section 'Related work' gives a short overview of the related work. Section 'Modelling data streams and time-selective data fusion in WSN' describes dataflows in ad hoc WSN using Q-model formalism, explains important theoretical notions to avoid ambiguity and introduces the alignment and selection algorithm. Section 'Accumulated delays of the sensor readings' describes the method for delay computation for the sensor readings and explains why it is necessary. Section 'Experiment setup' describes the experiment setup. Section 'Results' describes the results of the field tests, analysing the influence of time-selective strategy on in-network data processing, and section 'Conclusion' concludes this article and discusses some relevant aspects and future directions.

Related work

Timely handling of SA information collected by WSN requires distributed data fusion and aggregation by network nodes within the network. Instead of transporting all sensor data to a central server, we apply the paradigms of edge computation and in-network data processing.[4] The former is about performing as much computation close to the source of data as possible (either in sensor nodes or close to them) and the latter is about completing data fusion and aggregation within the network to mitigate bandwidth and energy scarcity, to increase solution resilience and the reliability of situation detection.

In most WSNs, the data acquired by sensors are processed in the same order as they arrive – even if the overall structure of the network and the definitions for detectable situations are known at design time. For example, Izadi et al.[11] present a data fusion approach which distinguishes low-quality input data from good-quality input data by assigning weights on sensor readings. The network delay is considered as one of the factors in the computation of weights, such that sensor readings with longer delays have lower influence on fusion result. This approach favours the freshest data and discards the opportunity to use delayed data that may be of high quality and better suitable for multi-sensor fusion. Other examples of prioritizing data freshness can be found in the papers that analyse quality-of-service (QoS) aspects in WSNs. A good survey of the state-of-the-art QoS techniques for delay handling and reliability mechanisms is provided in Al-Anbagi et al.[12] Similar overviews of WSN solutions for manufacturing and industrial control are given by Zhao[13] and Diallo et al.[3] The solutions described have reasonably good time-aware behaviour, that is, the ability to handle time-critical data and time-sensitive communication. However, the QoS aspects, such as real-time constraints and data freshness are considered most important in these surveys.

Another approach to guaranteeing timeliness in conventional WSN systems is to design them so that the delays caused by different communication paths meet the given deadlines.[14] Such strategy cannot cater for asynchronous nature of DIL and ad hoc WSN, where in-network data processing occurs with random and intermittent data bursts. Cheng et al.[15] present a method to modify the network structure in order to optimize the delays and to minimize the energy consumption. However, the structure of ad hoc networks is difficult to control by nature, hence the method suggested in Cheng et al.[15] may not applicable here. In order to provide better understanding of timeliness capabilities of WSNs,[16,17] consider probabilistic methods for traffic flow aspects, such as end-to-end delay, jitter and throughput. Both works point out that in most practical cases, the worst-case bounds for end-to-end delay in WSNs are not applicable. We emphasize that in ad hoc sensor networks and especially in networks for collecting SA information, the data consumer must be able to analyse the validity of the data online[18] and to determine how long the data are usable. The time-selective strategy for in-network processing can handle more variability in end-to-end delays, but requires a means to compute the delays accumulated during the data transport through the network.

The main sources for timing non-determinism in contemporary WSNs include transmission delays, packet losses, queuing for transmission, nodes contest for radio frequency medium and clock drifts and jitters in individual nodes of the network. Transmission-related delays originating from send time, access time, propagation time and receive time are a well-researched area in traditional Ethernet-based networks.[19] In ad hoc WSN solutions, where time synchronization is not used, the transmission related non-determinism can be mitigated for low number of hops by applying contemporary transceivers (e.g. using IEEE 802.15.4 protocol), which allow for modifying the contents of a packet after packet transmission is started by utilizing delay computation method described in Maroti and Sallai.[20]

Timing challenges in WSNs also include packet losses, which can happen due to dynamically changing network structure and unreliable wireless links.[12] The nodes may autonomously join or leave the network, interference from other sources may influence the wireless links (which may force the WSN to find different routing paths) and also mobile nodes must be considered. The delays can arise also from interactions where sending node is unable to transmit due to periodic activation, low duty cycle or other network scheduling policies, the resulting queuing delays for partner nodes are often ignored. Although the execution periods of the processes in network nodes may be highly deterministic, the messages are delayed and transmitted at non-deterministic times. This causes the end-to-end delays to be highly unpredictable, and the same applies to the order of data elements as packets may arrive out of order. Each time the system's structure changes due to changing goals by users or the environment, the network must adapt to the changing interaction patterns and delays.

Another aspect complicating timing analysis in an ad hoc WSN is unpredictability of the data production by autonomous nodes. First, the traffic rates of produced data by sensor nodes depend on the application, sensor modalities and sensor process signal processing capabilities. For example, more intelligent and autonomous sensor nodes can avoid reporting altogether if the monitored situation is unchanged or report only as often as required by the rate of change of situation (i.e. monitoring environmental aspects may not need as high rate of reports as measuring current or voltage spikes, or tracking a mobile object). Second, nodes in WSN often apply duty cycling or other transmission scheduling policies to mitigate bandwidth and energy usage.[21] Doing these decisions autonomously according the current situation, regarding environment or local energy level adds unpredictability.

The problem of handling out-of-order data for sensor fusion is not well researched in scientific literature[22] and even less so in papers considering ad hoc WSNs.

In area of multi-sensor data fusion, the related topic is called out-of-sequence measurements (OOSM).[22] OOSM can be caused by variable propagation times for different data sources or by heterogeneous sensors operating at multiple rates. The problem becomes especially relevant in large-scale networks consisting hundreds to thousands of measuring devices, as the complexity of network communication increases and communication delays of data packages get bigger.[23] In the area of multi-sensor data fusion, the solutions for this problem focus mostly on enhancing filtering algorithms (e.g. Kalman filter or particle filter) that cope with measurements arriving only a single or a few steps later.[22,24] We consider those approaches not well suited for in-network multi-sensor fusion in ad hoc WSNs. The delays in such networks can be much more unpredictable and longer and approaches considering filtering and state estimation are computationally more complicated and resource demanding.[23] Some early examples that consider out-of-order arrival of data for in-network processing in WSNs are Shi et al.[25] and Xiaoliang et al.[26] While both papers consider OOSM filtering approach with discrete step delays, the former handles mixed and bounded delays from a single sensor and the latter deals with delays from multiple sensors with delay length of a single sensor data refreshing period. These approaches are still in their early stages and not yet suitable for DIL and ad hoc WSNs where multi-sensor fusion is considered.

A good overview of the existing data fusion techniques for WSN is given by Yadav et al.,[27] but the listed works in given overview neither consider variable arrival delays in ad hoc networks or streams that may result in out-of-order arrival to the fusion node nor give sufficient attention to other timing characteristics other than freshness of data. Examples of distributed data fusion in WSNs are described in Bahrepour et al.[28] and Lai et al.,[29] where events detected and sensor readings collected by individual sensor nodes are assembled by a fusion node. These works do not discuss the validity or simultaneity of the input data for the fusion algorithms.

Classical models for distributed systems often use abstractions at various levels to compensate for timing non-determinism.[30] Examples are lock-step synchronous models,[31] fixed or no drift in individual clocks[32] and/or delays with fixed bounds[33] (which essentially models a subset of synchronous systems). We consider the Q-modelling technique[34] for the analysis of ad hoc WSNs, as it naturally facilitates modelling of timing aspects of asynchronous communication and queuing delays across the communication paths through the network while considering the precision of data timestamps. The original purpose of the Q-model is to analyse time correctness of interprocess communication of a collection of loosely coupled, repeatedly activated and terminating processes,[10] where the purpose of the time-selective communication is that the input data for the consumer process should be exact from the desired time interval (not produced before or after that time interval – the freshest data are not always desirable). However, the time-selective communication on such autonomous and distributed real-time systems results in a situation where some of the execution sequences and data produced by them are discarded and some may be used as inputs to another process several times.[34]

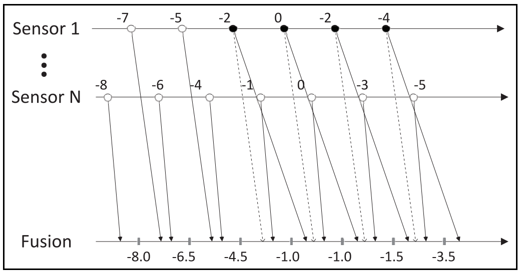


Figure 1. Example with delayed sensor readings causing out-of-order arrival.

Modelling data streams and time-selective data fusion in WSN

This section defines and explains some important concepts, such as temporal alignment of data, validity time of a stream element and simultaneity interval for stream elements across streams from different sources. The section also introduces a time-selective data fusion strategy for WSN and gives a detailed overview of the algorithm for the temporal alignment of data and selection of compatible elements for data fusion.

Sensor readings arriving out of order

In order to illustrate the necessity for selecting temporally correct data from sensor data streams for data fusion, we describe a simple freezer example. Imagine a large freezer which has several spatially distributed temperature sensors inside. As using wired sensors in such an environment can be costly and difficult to deploy, WSN technology is used for convenience. All wireless nodes are considered asynchronous, that is, each node has its own individual clock that is not synchronized to the global reference. In this example, only the latest readings from each of sensors are fused to get an average. The fused value is reported to the user periodically. Neither sensor readings nor fusion results are stored in the fusion node. If the temperature rises equal or above zero, there is a risk of spoiled goods. The notion of data fusion in this example is an exaggeration and is used for consistency reasons. The example is illustrated in

Figure 1. The white round markings on sensors time axis indicate sensor readings with normal delay. The black round markings indicate sensor readings with increased delay, and dashed line indicates the delay as it was expected by the designer of fusion algorithm. The computation and reporting of the averaged results take place at instances indicated by the ticks on fusion time axis.

The cause for the increased delay, as depicted in Figure 1, could be a route change in the multi-hop network as goods are being stacked up on the radio path (similar increased delay could easily be caused also by network overload, etc.). Consider a situation where both sensors register a 0° value, but due to changed route, one of the sensor readings delay increases (now another WSN node relays its readings to the fusion node). If the fusion node does not consider variable delays and averages only readings according to their arrival, then the reported temperature never rises above −1.0° and the fact that the freezer temperature was zero for a short period of time is left unnoticed. It should be noted that even if there are no increased delays in the described freezer example, there is still a chance that the zero temperature would not be reported. As processes in this example are considered asynchronous, the fusion node execution and following reporting can happen between the arrival of two zero readings.

To mitigate such problems, we present a time-selective strategy. The fusion node should, during each of its algorithm execution, have access not only to the latest sensor readings but also preferably to an array or a batch of past readings from each distributed sensor. The storage size for the available past readings should be large enough to hold also readings as old as the longest allowed delay that can happen for fusion inputs for that particular network. Furthermore, it should be possible to align the arrived readings from different sensors to the fusion node's time axis. This makes it possible for fusion process to select readings that are compatible in the temporal domain. The theoretical model for time-selective data fusion is discussed and analysed in the next section.

Modelling time-selective data fusion in WSN

We use Q-model[10] formalism to specify and model the WSN as a distributed communication system, consisting two main classes of components: processes and channels. Each interacting node in the WSN can execute several different processes p. The communication between the processes in different nodes across the WSN is modelled by a channel σ_{sf}, where s denotes a sensor process and f denotes a data fusion process, respectively, that are communicating. The channel is a logical tool that maps the output from one process to input of another process according to their timesets with the expression

$$\sigma_{sf}\colon T(p_s) \times T(p_f) \times valp_s \rightarrow proj_{valp_s} domp_f$$

which, in the context of this article, conveys sensor process values to fusion process domain of definition. Here, the sensor process p_s and fusion process p_f are considered to be running, respectively, in disparate network nodes for sensing and fusion (in some scenarios, it is also possible that a single network node is running both processes). The variables $T(p_s)$ and $T(p_f)$ represent the execution timesets of these processes. The mappings between processes are activated repeatedly, either periodically or sporadically. For example, the execution timesets for periodically activated processes can be modelled by expression

$$T(p) = \{t : t_n = t_0 + n \cdot t_a\}$$

where $t_0 = 0$, $n \in \mathbb{N}$ and t_a is the interval between two process executions. The processes are considered asynchronous and have each their own timeset and time counting mechanism. For modelling purposes, each activation/execution instant t_n for a process also determines the timestamp of the data produced by this process. When the produced data is transmitted by the node, its timestamp is updated to reflect the delay between the process activation instant and the actual transmission moment. The practical process of delay computation is described in section 'Accumulated delays of the sensor readings'. If computationally feasible, the timestamp is also updated to reflect the actual time-moment of the physical-world situation that is captured by sensor process. Computing the exact time instant of the situation might not always be trivial due to limited resources of low-cost WSN nodes.

The data usage between processes is time-selective. The data stream resulting from data produced by one of the processes is moderated by the channel function and transferred to another process. Formally, the channel function is expressed as

$$K(\sigma_{sf}, t) \subset T(p_s), t \subseteq T(p_f)$$

where the number of stream elements conveyed by channel or rather temporal span of the accessible elements received by fusion process is defined as time interval $K(\sigma_{sf}, t) = [\mu, \nu]$ on the time axis of the fusion process. An overview of the nodes, processes and related channels is given in Figure 2. Each node may run several processes, where each process may have its own execution timeset and execution period. Each

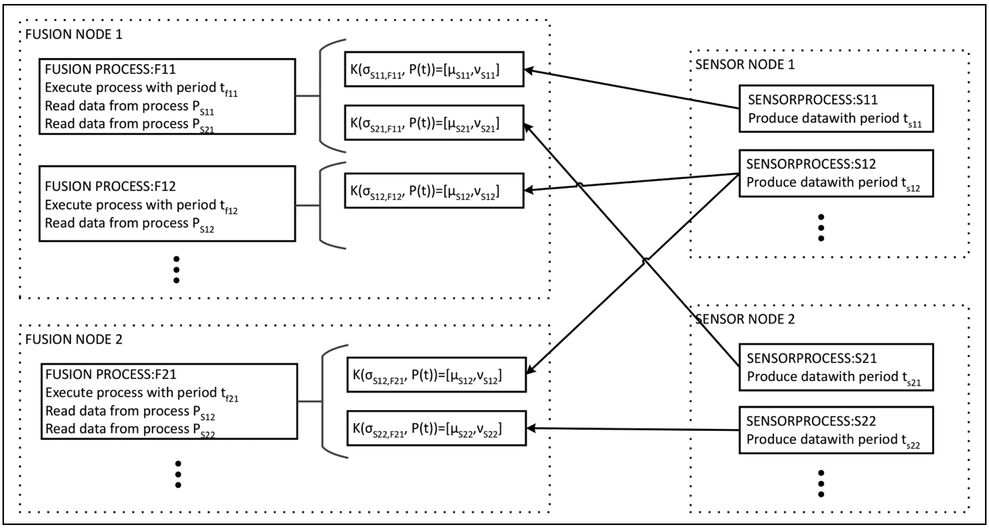


Figure 2. Overview of processes and channels.

sensor process execution can result in a sensor reading which is conveyed to the fusion process via the channel function. Each fusion process (data consumer) establishes a separate channel for each sensor (data producer) process. Each channel may have a different interval $[\mu, \nu]$ of the accessible elements, and the fusion process has access to the transmitted sensor readings according to the channel function.

For example, for the fusion process p_f during its single execution, the interval $[\mu, \nu]$ represents the requirement for the accessible stream elements from sensor process p_s. The variables μ and ν are defined by the fusion process and represent, respectively, the earliest (oldest) and latest (freshest) instants of the sensor process output data. Usually, $\nu = 0$ as the freshest possible data is required by the fusion process. During each execution, the fusion process can read data from several channels, that is, it can have access to several streams, each from a different process. The actual time instant for the freshest possible element for specific channel for the interval $[\mu, \nu]$ is specified by the expression

$$t = \max_{t_s}\{t_s < t_f + \eta(\sigma_{sf}, t_f) - \zeta(p_s, t_s)\}$$

where $t_s \subseteq T(p_s)$, $t_f \subseteq T(p_f)$, and $\eta(\sigma_{sf}, t_f)$ is the length of time interval during which the fusion process receives the data. The variable $\zeta(p_s, t_s)$ computes the execution time of the sensor process. For simplicity, the propagation time of a radio packet is considered zero. The actual delay due to periodic execution of processes at any fusion execution instant can be computed by the formula

$$\varphi_n(t) = t_f - t_s$$

The oldest feasible element (denoted with variable μ) for each channel during a single fusion process execution is determined according acceptable delays according to the use case or the estimated delays from other streams used for same fusion process. As a consistent stream of data elements must be stored, the oldest feasible element stored is practically limited by the maximum number of elements stored, which is limited by the memory available on the node.

Regarding the feasibility of alignment and selection of temporally suitable sensor readings for fusion process, the memory buffers for storing the sensor readings from different channels should be large enough to cope with the delays caused by the nature of ad hoc DIL WSN.

Temporal validity interval of input data

This section discusses the importance of temporal validity intervals of input data for fusion node and how the value of validity interval of sensor readings affects the selection of temporally compatible inputs for the data fusion in WSN. The necessity of checking and ensuring the sensor data validity has been discussed in our earlier papers,[4,35] where it has been explained how every sensor reading has temporal and spatial validity intervals associated with it. These intervals depend on several aspects, for example, the validity area depends on the location of the WSN and on the properties of the phenomenon being observed, while the temporal validity interval depends both on the properties of the

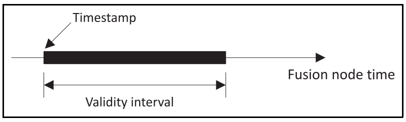


Figure 3. Timestamp and a validity interval.

environment where the node is located and on the phenomenon being observed. The sensor node augments its output data, with the validity intervals, and verifies that readings are still valid before transmitting them. The fusion node in turn verifies that the validity intervals of the sensor readings upon their arrival do match with the constraints set on incoming data. The output of the fusion process is in turn again accompanied with the metadata which also contains respective validity intervals checked by the users of fused data.

It is difficult to determine the precise arrival time of data to the fusion node in an ad hoc WSN in advance. The temporal validity interval is used to set an upper bound on the transport and usability time of the sensor readings. When temporal validity interval expires before the sensor readings arrive to the fusion node, the readings are discarded. The temporal constraints employed by the fusion node are not necessarily related to the validity intervals of arriving data. The constraints can be stricter or more relaxed depending on the application and context (as decided online by the fusion node or at design time by the system designer). If the validity of arrived data satisfies the temporal constraints, it is stored in the fusion node memory, where it remains available so that the fusion process can select the suitable inputs at the right time. Figure 3 describes a timestamp and a validity interval for a sensor reading on a fusion node timeline. The timestamp t_s indicates the time moment when the sensor reading was acquired and the validity interval $I_{validity}$ indicates the period of time during which the resulting sensor reading is valid.

The validity interval for a single sensor reading with timestamp t_s can be expressed as follows

$$I_{validity} = [t_s, t_s + t_{valid}]$$

where t_{valid} is a length of validity interval on fusion node's time axis.[35] In case the fusion node receives input data from different sources, all the data must be valid at their arrival.

Fusion requires overlapping validity of stream elements

Validity intervals of individual data elements can be used for grouping data and selecting data elements with overlapping validity intervals. Figure 4 depicts four sensor readings, their timestamps and validity intervals.

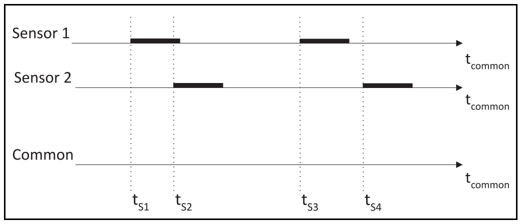


Figure 4. Overlap of validity intervals.

The t_{S1}, t_{S2}, t_{S3} and t_{S4} are the timestamps of the sensor readings aligned on the fusion node time axis, and black rectangles indicate the respective validity intervals $I_{valid}(t_{Sn})$. It can be observed that sensor reading with timestamp t_{S2} falls within the validity interval of another sensor reading with timestamp t_{S1}. There is a period of time during which both sensor readings are valid and both can be used as inputs for fusion process. This period of simultaneous validity or an overlapping validity interval can be expressed as

$$I_{valid}(t_{S1}, t_{S2}) = I_{valid}(t_{S1}) \cap I_{valid}(t_{S2})$$

The opposite situation can be observed in case of t_{S3} and t_{S4}, where the validity of sensor reading with timestamp t_{S4} does not overlap with the validity of sensor reading with timestamp t_{S3}, thus they should not be used together for detection or synthesis of a more abstract situation (i.e. data fusion). Supposing now that there are several distributed sensors that produce data streams, the fusion will provide correct results only when the fusion process takes as an input, sensor readings that are valid simultaneously, that is, for which there exists a common overlapping validity interval. However, one can also consider a situation where the fusion process, after its execution, has access to data which were valid during their arrival at fusion node, but which validity has expired by the time moment when the actual selection of suitable input data takes place (the channel length of accessible input data for fusion may be longer to hold also data, which validity has expired). In this case, it is important that the validity intervals of potential input data from different sources have an overlap. The resulting fusion output data have their own validity interval assigned before the fusion output is transmitted to its corresponding consumer. The validity interval of fusion output is subjected to same constraints as described previously.

The feasibility analysis of fusion of stream elements requires us to consider some necessary design decisions – for example, assigning periodicity of sensor reading, defining validity intervals for sensor readings, managing clock jitter in sensor nodes, maintaining average

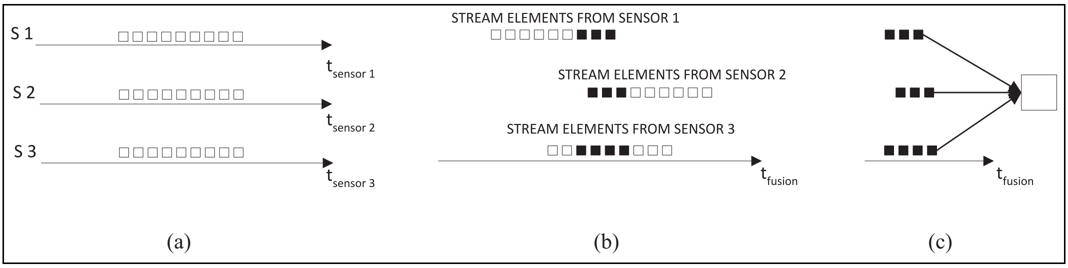


Figure 5. Example of alignment and selection of temporally compatible elements: (a) sensor data streams, (b) data stream alignment and (c) data fusion.

traffic speed between network nodes and defining the length of simultaneity interval that enables data fusion.

Simultaneity interval

The simultaneity interval serves a dual role – it enables to convey and evaluate the actually achieved synchronicity in a network and, if necessary, to compare it with the required synchronicity; and it provides a design parameter for assigning validity intervals for individual sensor readings in order to achieve feasible fusion of those readings. In general, the simultaneity interval specifies a set of events (e.g. sensor readings) that can be considered 'simultaneous' within some window of tolerance and can be used for fusion, and it is a period of time that elapses from the occurrence of the first of a group of events until the occurrence of the last event of the same group.[10] A simultaneity interval for two sensor readings with timestamps t_{S1} and t_{S2} is expressed as

$$I_{sim}(t_{S1}, t_{S2}) = |t_{S2} - t_{S1}|$$

For example, if fusion process receives four sensor readings as inputs with the delays of $d_1 = 980$ ms, $d_2 = 1010$ ms, $d_3 = 875$ ms and $d_4 = 1045$ ms, the simultaneity interval for these inputs is $I_{sim}(d_1, d_2, d_3, d_4) = 170$ ms.

As design goal or rather a requirement for simultaneity of sensor readings (more precisely the observed situations that the sensor readings represent), we define the simultaneity constraint C_{sim}. For example, if we look at the two sensor readings with timestamps t_{S1} and t_{S2} depicted in Figure 4, the correct fusion of these readings requires (in addition to overlapping validity intervals) that the simultaneity interval of the given group of sensor readings satisfies: $C_{sim} \geqslant I_{sim}$. This requirement is independent of the group size, all sensor readings grouped into single I_{sim} according to their timestamps must satisfy C_{sim} in order to be interpreted as simultaneous.

In practice, the simultaneity constraint is first chosen on the basis of application and second on the basis of the precision of computed delays of sensor readings. The computation of delay of sensor readings is discussed in section 'Accumulated delays of the sensor readings'. The sample application used in this article is the detection of moving vehicles. The choice of simultaneity constraint will influence the precision of the position estimate of the detected vehicle. For example, if $C_{sim} = 400$ ms is chosen, the position of the vehicle is interpreted to be within the area it can cover in 400 ms (given that the speed of the vehicle is known).

Alignment and selection of compatible elements from streams

The basic idea of the alignment and selection algorithm is to group the available readings from different sensors by temporal characteristics (such as validity intervals $I_{validity}$ and/or simultaneity interval I_{sim}) and to use only these groups as inputs for fusion process. The need for such an approach is driven by the problem which arises when distributed and autonomous ad hoc WSN nodes are used for simultaneous observation to detect complex situations in real time. Due to the delays, the sensor readings used as inputs for in-network distributed fusion and aggregation nodes may not characterize the same real-world situation if used in the order of arrival. One of the real-world cases can be observed in Figure 10, where the received stream elements have been projected onto the fusion node's time axis. One can observe that the stream elements on the bottom axis do not overlap with the elements from two of the streams above. However, even in this case, it can be the case that the validity intervals of the stream elements overlap and one can also define a sufficiently relaxed simultaneity constraint, so that a set of four stream elements can be selected and presented to the fusion algorithm as inputs.

Figure 5 shows three steps of the alignment and selection process of compatible data from sensor streams. Figure 5(a) represents the received stream elements by the fusion node. The arrival order of the stream elements from different sensor nodes is not

Algorithm 1. Alignment and selection of temporally compatible elements

Input:

a) $S = \{S_1, \ldots, S_m\}$, where m is a number of streams
b) C_{sim}– a simultaneity constraint

Definitions and functions:

a) A_{sim}– An ordered array for sets of simultaneous sensor readings
b) $T(element)$– returns the timestamp of a stream element
c) $I_{sim}(t_1, t_2)$– returns a simultaneity interval of a set of timestamps

function align_and_select $(S, C_{sim})\}$
Sort streams according to length
Choose shortest stream $S_1 = Min(S)$
$Foreach(elements_{S_1} \subset S_1)$ do:
declare an empty set D
insert $elements_{S_1}$ to D
$Foreach(S_i \subset S)$, where $1 < i \leqslant m$ do:
compute $t_w = weighted\_average(D)$
find $elements_{S_i} \subset S_i$, such that $|t_w - T(elements_{S_i})|$ is minimal
insert found $elements_{S_i}$ to D
if expression $C_{sim} \geqslant I_{sim}(D)$ evaluates true, then:
insert identified set of simultaneous elements D to A_{sim}
return A_{sim}

known in advance as sensor nodes run asynchronously. The incoming stream elements are received by middleware component at the fusion node. The middleware performs a validity check[18] and projects the stream elements to the node's local time domain. The black filled squares in Figure 5(b) and (c) depict temporally compatible stream elements. When the fusion process executes and requests for inputs, the data alignment and selection algorithm aligns the stream elements from different sensors in the fusion node time domain as depicted in Figure 5(b). The selection of temporally compatible elements from streams is depicted in Figure 5(c).

The process of selection of temporally compatible elements is described by Algorithm 1. The algorithm takes m number of streams as inputs. Each stream S_m contains n number of elements $element_n \subset S_m$. As the fusion node specifies a separate channel function $K(\sigma_{sf}, t) = [\mu, \nu]$ for each communication partner, the requirements for each stream may be different (the exact parameters for each channel function are specified in the data subscriptions made by the fusion node middleware). The incoming streams are sorted by their relevance. The criteria for relevance can either be confidence or fidelity level of the stream elements or also currently available number of elements in the stream. The most relevant stream S_1 (e.g. with minimal number of elements) is taken as a starting point. The algorithm processes the elements of S_1 one by one. For each element $elements_{S_1} \subset S_1$, the algorithm finds the closest element $elements_{S_2} \subset S_2$ from the next stream S_2. Closeness is defined temporally as the time interval between the timestamps of two stream elements. After finding the closest element to $elements_{S_1}$, a new time instant t_w is computed. t_w is a weighted average of the timestamps of the identified closest stream elements. The usage of the weights for timestamps is motivated by the desire to take into account the confidence level of the computed delay. For example, stream elements which transport include more hops, resulting in lower precision for computed delays may have lower weights. The obtained t_w is then used to find the temporally closest element from the next stream. The process repeats until all streams have been processed. Each time a new closest element from the next stream is found, a new t_w of timestamps is computed from all previously identified elements. This way, the algorithm finds for each $elements_{S_1} \subset S_1$ a set of temporally closest elements across all streams. In Algorithm 1, this set is denoted as D. The obtained sets of temporally closest elements are then inserted into an ordered array A_{sim}, which is ordered by the simultaneity intervals I_{sim} of the sets in D. The sets D, whose simultaneity interval I_{sim} values exceed the simultaneity constraint C_{sim}, are discarded. The algorithm returns A_{sim}.

In practice and in the test described in section 'Experiment setup', only set D with the smallest simultaneity interval $I_{sim}(D)$ is used in data fusion and the other elements in A_{sim} are discarded. This step is needed to simplify the first iteration of the WSN experiment described in this article. The feasibility of passing all sets of D that satisfy the simultaneity and validity constraints to the fusion process depends both on available computational resources and time available for fusion process execution in a practical use case. Executing fusion process more than once, to consume all available inputs, would also produce a more consistent stream of fusion outputs.

Accumulated delays of the sensor readings

In order to process the sensor readings in a time-sensitive manner and to align them on a common reference time, the processing node must be able to compute the delays of its inputs with certain required precision. There are two aspects to consider here, first, how the timestamp of the observed situation is computed by the sensor data acquisition process and, second, how the delays are computed and projected to the fusion node local time axis.

The former problem may not be trivial in the case of low-cost sensor nodes. In ad hoc WSN, it is not feasible that a sensor reading is transmitted from each single sample. In most cases, multiple sensor samples, called a frame, are either aggregated (averaged, summed, etc.) or processed into a single sensor reading for the entire frame period. Due to limited computational resources in low-cost sensor nodes, it may not be always feasible to compute the exact time instant of the actual situation from the sampled frame, so a start of the frame is considered as the process activation instant t_s and is used as a creation time instant (timestamp on a sensor node time axis) for sensor readings. Although this does make the modelling and analysis easier, this approach may result in considerable, but bounded error $e \leq t_a$ (t_a is a single sensor process execution period) in sensor reading delay computation. This error must be taken into account when computing the accumulated delay of sensor readings delay as this affects the comparison of the validity intervals of several readings from different sensors, when projected on to the fusion node time axis and interpreting the fusion results. When the sensor process supports the computing of the exact time instant of the observed situation (for which the sensor reading has been computed), the resulting timestamp for the sensor reading should be updated accordingly.

For the latter problem, the classical methods align sensor data to a common reference with the help of time synchronization algorithms.[36,37] However, applying classical methods, where all data are collected via gateway (sink) to a central server outside of WSN, may lead to significant communication overhead and is not optimal in ad hoc networks. Other methods to align data without synchronizing the WSN nodes include, for example, alignment based on causal dependencies,[38] where authors use vector clocks. We consider the system of vector clocks inefficient because of two specific reasons. First, the size of a timestamp is proportional to the number of nodes in the network, and second, using vector clocks requires additional communication between the sensors in order to establish the causal relations between the sensor readings.

Instead of traditional synchronization methods in WSNs, which can lead to significant communication overhead,[37] we take advantage of existing TinyOS packet-level delay computation service,[20] which allows to mitigate considerably the timing indeterminism for transmission-related delays (send time, access time and receive time) for a single hop. Its main advantage over other synchronization methods is its lightweight nature. Each node computes the accumulated delay for the data and passes this temporal information along with the transmitted data. The packet-level delay computation method supported by TinyOS operating system allows the communication stack to automatically convert the sending node local time to the receiving node local time by appropriately modifying the time value within the packet after its transmission is started. The sending node converts the time value within the packet to a delay d_{comp} spent up to that moment since the creation of data and the receiving node in turn can use d_{comp} to compute the data creation time moment on its own local time domain by subtracting its value from the time moment of data arrival. This method does not

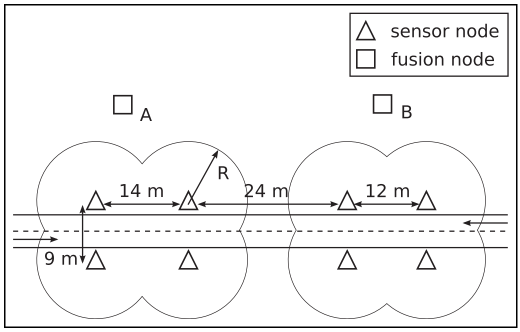


Figure 6. Sensor node placement for vehicle detection.

provide synchronized network time, but provides a sub-millisecond accuracy for a single hop. Combining this method with time-selective strategy makes it possible to obtain correct results when data are fused from sensors, which readings are produced asynchronously. In other words, neither the clocks nor the actual sampling of the data by distributed sensor nodes are synchronized in any way. In case of multi-hop situation, each forwarding node in the network estimates the time interval d_{comp} between receiving and transmitting data and adds it incrementally to the previous delay (age) of data.

Experiment setup

This section describes the field experiment carried out to demonstrate the application of time-selective data fusion in WSN. Eight microphone array sensor nodes were used to record 30 min of acoustic signals by the side of an urban road with moderate traffic. The same sensor nodes were then set up in laboratory conditions where they, instead of recording signals, now read the previously saved acoustic data and treated it as if it were directly received from their analogue-to-digital converter (ADC) modules. This way, it was possible to repeatedly play through the same 30 min of situations with different experiment configurations to compare and analyse the results.

As stated above, the sensors used for the experiment are microphone array sensors. Each array consists of six microphones which enable sensors to compute an angle of arrival (AoA) of sound sources using a time-difference-of-arrival method. The sensor nodes are based on BeagleBoneBlack development boards for running sensor processes and an IEEE 802.15.4-compliant 2.4 GHz transceiver (based on Atmel ATmega256RFR2) for wireless ad hoc networking. The fusion nodes are implemented using only Atmel ATmega128RFA1 microcontroller-based platforms.

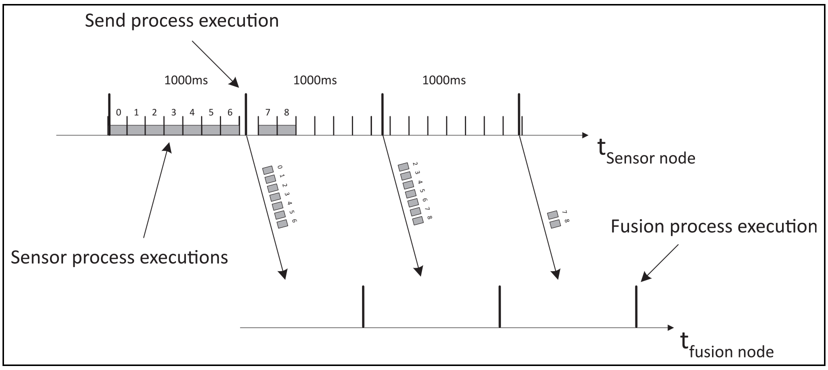


Figure 7. The figure illustrates how the number of transmitted values depends on the validity intervals. Note that all processes are asynchronous.

The more detailed overview of the hardware is given in our previous work.[4]

Sensor node placement for the experiment is depicted in Figure 6. Two fusion nodes A and B were used, with node A receiving messages from the four sensors on the left and node B receiving messages from four sensors on the right. Both fusion nodes transmit their results to a single gateway, not depicted in the figure. For brevity, the results from the two distinct clusters A and B are presented together as results from a single network.

Sensors were placed next to the road in order to detect passing vehicles. A total of 92 vehicles, of which 2 were buses, 2 were motorcycles, and the rest were passenger cars, passed by the sensors during the 30 min. The speed limit at this stretch of road is 50 km/h. Sensor sampling speed for each microphone was 20 kHz and measurement frame length, used in AoA processing, was 136.5 ms. As a result, approximately seven AoA calculations were done per second by a single node. Before transmitting the results, the sensor node was able to check the temporal validity of readings (described in Ehala et al.[4]) and to transmit only the valid results to fusion nodes at an interval determined by the data subscription agreement between sensor and fusion nodes.

For the experiment described in this article, the sensor node sending period was 1000 ms. In between the sending periods, the seven sensor readings that the sensor node was able to sample covered 955.5 ms, and the sampling of frames (sensor process) is asynchronous with the sending period. At the end of each sending period, the sensor node assembled the available valid readings into a batch of single payload and transmitted it to the fusion node. In order to process all received readings, the fusion process execution period was also chosen to be 1000 ms. The different execution times of sensor, sending and fusion processes, for the experiment setup are illustrated in Figure 7. The delay arising from periodic activation at any fusion process execution instant can be computed by formula $\varphi_n(t) = t_f - t_s$. The maximum delay due to periodic asynchronous processes with given settings can be up to 2000 ms. The actual transport time depends on uncertainties induced by ad hoc network and environment. Considering maximum delay, the validity interval for sensor readings in this experiment was chosen to be 2000 ms.

However, when no vehicles are near the sensors, the AoA calculations end with a negative result, meaning that no vehicle is detected – in Figure 7, these cases are illustrated as empty slots at the sensor process executions. Negative results are never sent to the fusion node. If previous AoA estimation results which are still valid at the sending time and no new AoA estimations have been computed, then the old results (within the validity interval of 2000 ms) are retransmitted to the fusion node. This means that some sensor readings could be used more than once by the fusion process. When validity time of buffered readings expires and there are no new positive results, nothing is sent to the fusion node.

In order to monitor what happens in the network during different runs of experiment, all sensor and fusion nodes logged their activity by writing different log messages to serial port. This way, the execution timesets for all processes, delays and other temporal parameters which cannot be otherwise extracted from the wireless processing environment could be recorded for analysis. Several single-board computers

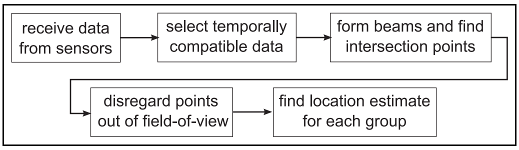


Figure 8. Estimating the location of sound source.

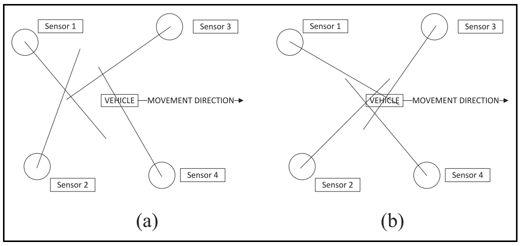


Figure 9. Fusion result without data alignment (a) and expected improvement with data alignment and selection (b).

(Raspberry Pi 2) collected these messages and timestamped them upon arrival. The single-board computers kept their own clocks synchronized via the network time protocol (NTP), so that all log records were comparable (WSN nodes themselves were not synchronized).

Location estimation by fusion nodes

Individual microphone array sensors alone can estimate the direction to a sound source from their position, but cannot effectively determine the distance to it, and therefore also the location of the source. A location estimate can be established, however, by several sensors in the same area by combining their direction estimates. Special fusion nodes are dedicated to this task, although in principle any network node can take up this task, if it has the necessary resources. The fusion process is depicted in Figure 8.

First, data are collected from all sensor nodes, which have detected a sound event. The data include the location of the sensor node (geographical coordinates), the measured direction estimate – the AoA of the sound (a geographic bearing) and metadata such as the sensor sensing range and a timestamp indicating the delay (or age) of the direction estimate. Based on the age of each direction estimate, compatible sound event instances are found and analysed together. Next, AoA beams are formed along all the direction estimates and intersection points of these beams are found. Due to the discrete nature of AoA calculation procedure and other inaccuracies of input data, all the beams will very seldom intersect in a single point. Rather, a cluster of intersection points emerges and the scattering or dispersion of this cluster determines whether the result should be considered a valid location estimate or not. From this cluster, a single geographical coordinate can be computed, which is a weighted average of the intersection points in the cluster. It is also checked that intersection points fall within the field of view of the involved sensors. Intersection points out of range of the sensors are not considered.

The resulting cluster of valid intersection points provides a basis for analysing the effectiveness of the fusion process. When the inputs to the fusion node are not acquired simultaneously, the resulting cluster of intersection points is more scattered as depicted in Figure 9(a). Figure 9(b) illustrates how applying time-selective data fusion strategy leads to improved fusion precision. In this case, provided the fusion node has access to streams of sensor readings that cover the vehicle passing, the alignment and selection algorithm should be able to select more compatible inputs for fusion.

In order to compare the experiment results, two separate parameters are used for analysis. These are simultaneity interval I_{sim} of the fusion algorithm inputs and the area of location estimation S_{loc}. The simultaneity interval I_{sim} of the fusion algorithm inputs describes the temporal dispersion of the computed delays of the sensor readings used as inputs. The second parameter, the area of the location estimation S_{loc}, is a rectangular area covering the cluster of intersection points formed by AoA vectors provided by the sensors. The S_{loc} is a way to assess the scattering (or dispersion) of the intersection points. If the cluster of intersection points is more scattered, the rectangular area is larger and vice versa. The actual position estimation of the noise source is computed by taking a weighted average of all the intersection points. Our hypothesis is that there is a correlation between I_{sim} and S_{loc}. The lower I_{sim} should result in smaller S_{loc}.

Experiment configurations

The experiments are carried out the same way as the WSN would have been deployed in real world by replaying the recorded data streams at every sensor node. The WSN nodes use their radio transceivers to exchange the data as they would if they were deployed in the field. The two different configurations of experiments are listed in Table 1.

The first experiment configuration is about using the freshest data first. This configuration does not use the temporal alignment and selection algorithm. The purpose of the experiment is to demonstrate the naive version of data collection from WSN, where each sensor

Table 1. Experiments, their configurations and parameter varied.

No.	Experiment configuration name	Varied configuration parameter	Value (ms)
1	Freshest data first	Validity constraint	2000
2			1500
3			1000
4			800
5			500
6	Temporal alignment and selection	Simultaneity constraint	1000
7			800
8			600
9			400
10			200
11			100

node periodically transmits a result to the fusion node, which consumes the data in their order of freshness.

The second experiment configuration applies the temporal alignment and selection algorithm, so that the temporarily compatible input data for fusion algorithm are selected from available inputs according to the similarity of the computed delays. This experiment configuration requires that the fusion process at every execution has access to a stream of sensor readings from each sensor process. In the current experiment, due to the limited memory in the fusion node, a solution was implemented where instead of storing the stream elements on fusion node, the sensor node transmits a batch of readings in each of its packets. As the maximum length of IEEE 802.15.4 physical layer frame is 127 bytes, it was possible to transmit a maximum of seven sensor readings (accompanied by appropriate metadata) in a single batch.

Both experiment configurations compute the delays of sensor readings using the same method as described in section 'Accumulated delays of the sensor readings'. The only difference is how the delay information is exploited. Without the alignment and selection algorithm, no simultaneity constraint is applied and the delays of sensor readings are only checked against validity constraints (the same validity constraint is applied both on sensor node before transmission and on fusion node side upon receival of data). The sensor readings with longer delays, which did not satisfy the validity constraints, were not used for fusion. With the alignment and selection algorithm the inputs are projected and aligned to fusion node time domain and only temporally most compatible inputs are selected and passed to the fusion algorithm, provided they satisfy the simultaneity constraints. During all experiment runs, all execution periods for both sensor and fusion processes were set to 1000 ms. The data validity intervals for fusion inputs are subject to different validity constraints during first the experiment, and during the second experiment, the validity constraint for fusion inputs is fixed to 2000 ms.

The difference between the two experiment configurations is illustrated by Figure 10, which depicts a sample set of sensor streams as inputs for fusion process. In the figure, the streams from different sensor nodes have been projected onto fusion node time domain and aligned according to their respective delays. If the fusion process starts to consume the sensor readings by the freshest data first from each stream, then the length of simultaneity interval I_{sim} of the resulting set of inputs will be more than 700 ms. However, if the fusion process is allowed to select temporally suitable elements, the value of I_{sim} is significantly reduced.

Results

This section presents the results of a total of 11 experiments. The results for the first five experiments are presented in Table 2. During these experiments, the temporal alignment and selection algorithm and simultaneity constraint were not applied. The results of the application of temporal alignment and selection algorithm on stream elements and the use of different simultaneity constraints are presented in Table 3. In both tables, column no. 4 contains measured average simultaneity intervals for fusion inputs (a measure of

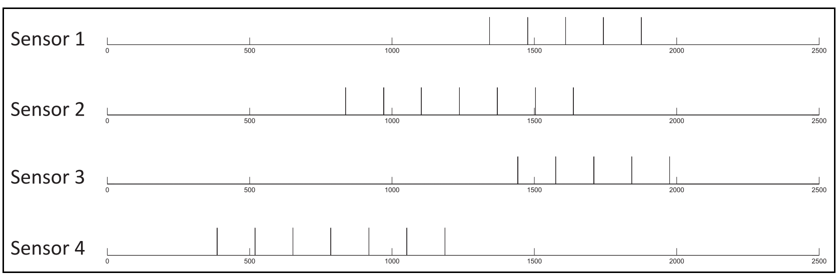


Figure 10. An example of stream elements aligned on fusion nodes time axis before single fusion execution.

Table 2. Results without the alignment and selection algorithm, with application of validity constraint.

No.	Validity constraint (ms)	Simultaneity constraint	*Average*(I_{sim}) (ms)	*Average*(S_{loc}) (m^2)	Computed positions	False negatives	False positives
1	2000	Not applied	947.5	34.4	500	1	31
2	1500	Not applied	700.8	31.7	348	17	14
3	1000	Not applied	506.9	22.2	188	19	3
4	800	Not applied	344.0	11.4	90	47	0
5	500	Not applied	140.0	11.3	10	87	0

temporal consistency of inputs) and column no. 5 contains average rectangular area of intersection points, which represents the precision of fusion result (a position of a passing vehicle). The results for I_{sim} and S_{loc} are averaged for each experiment, which is 30 min. The next column presents the number of completed fusions (successful fusion means that a position that satisfied spatial constraints was computed), and the last two columns show how many of the fusion results were false negatives and false positives. A false negative is a vehicle that was undetected and a false positive is a computed position where there were actually no vehicles present. We are able to find false positives and negatives because the time intervals when a vehicle was in range of the sensors were recorded during the original field experiment.

During the first five experiments presented in Table 2, the fusion node consumed the arrived inputs as freshest data first, in the same order as they arrived. A considerably long average simultaneity interval achieved can be explained by periodic and asynchronous execution of ad hoc WSN nodes. Furthermore, the allowable age for the freshest available sensor reading for transmission depends on the validity interval. With validity being longer than sensor process execution period, the sensor node was allowed to transmit or retransmit older values. The experiments 1–5 show that when the value of validity constraint is reduced, the values of I_{sim} and S_{loc} improve. However, the number of false negatives quickly rises. The lower values of validity interval filter out the sensor readings with longer delays. This does not improve the fusion reliability as with lower values of validity intervals more cars are left undetected. The effect can be explained by Figure 11, which presents a histogram from experiment 1 with measured sensor delays by fusion node A. During this experiment, the validity interval of sensor readings was 2000 ms, meaning the sensor is allowed to retransmit the valid readings if there are no newer readings. The figure is illustrative as it depicts the delays without the application of temporal constraints.

The average for all delays of sensor readings received by the fusion node is 1357.7 ms. Altogether, the fusion node A received 9157 sensor readings. It can be observed that (due to periodic execution) the majority of the readings fall into an interval between 500 and 2000 ms. The reason why there is ca. 200 ms delay before the first readings arrive to the fusion node must be, in addition to the sensor sampling time, fusion node's asynchronous and periodic execution. The readings that have been delayed more than 2000 ms are most likely the ones that were retransmitted due to no new valid readings. The theoretical maximum of a delay due to periodic execution and retransmission can be up to 3000 ms (validity time added to delay caused by periodic execution of processes). Longer delays must have been caused by network and environment induced uncertainties (or other real-world unpredictable causes).

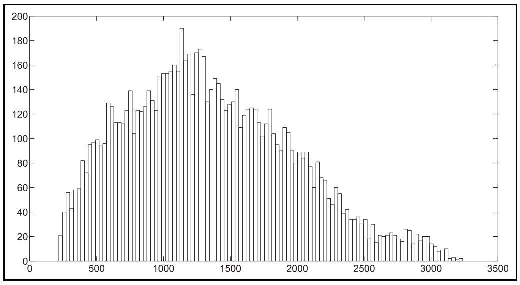


Figure 11. Sensor delays measured on fusion node time axis.

The rest of the experiments (6–11) in Table 3 show how averaged values for simultaneity interval and area of location estimation are influenced by alignment and selection algorithm together with different values for simultaneity constraints.

The experiment indicates a correlation between simultaneity constraint and the area of average location estimation. The lower the simultaneity constraint, the smaller the area, that is, the precision of position estimation improves. However, the same side effect as during the first five experiments without the temporal alignment and selection algorithm is present. Stricter simultaneity constraint filters out the actual vehicle detections with lower precision (larger values of S_{loc}).

Table 3. Results with the alignment and selection algorithm, with application of simultaneity constraint.

No.	Validity constraint (ms)	Simultaneity constraint (ms)	$Average(I_{sim})$ (ms)	$Average(S_{loc})$ (m^2)	Computed positions	False negatives	False positives
6	2000	1000	208.3	22.7	446	0	14
7	2000	800	180.8	21.2	428	0	12
8	2000	600	147.5	19.8	404	1	11
9	2000	400	86.1	17.6	342	2	8
10	2000	200	58.9	17.1	284	7	3
11	2000	100	6.0	16.7	200	20	2

For example, the usage of simultaneity constraint 100 ms leaves 20 vehicles undetected (false negatives).

Experimental results of the two different experiment configurations (freshest data first vs time-selective strategy) clearly show that time-selective approach achieves considerably better results than the configuration which uses only validity constraints and prefers the freshest data first.

Choosing a good criterion for WSN performance is not trivial. One possibility is to use accuracy as a criterion. In statistical tests, the accuracy can be measured by formula $Acc = (TP + TN)/(TP + FP + FN + TN)$, where Acc = accuracy, TP = True Positives, TN = True Negatives, FP = False positives and FN = False Negatives. As we can see, the accuracy is increased if either false positives or false negatives or both are decreased. For these experiments, we consider a low number of false positives as the most important outcome. This number should be low as we do not want false alarms and if possible prefer to avoid the positions computed based on false alarms. Considering the results of all experiments carried out, the minimum acceptable number of false positives is chosen as three. The other outcome parameters to be considered are average area of computed positions S_{loc}, the number of computed positions (successful fusions) and the number of false negatives.

With the first experiment configuration (the freshest data first approach), the best results are with validity constraint being 1000 ms, which is the first threshold, where the number of false positives is three. However, the number of false negatives is too high, 19 false negatives out of 92 vehicles leaves 20.7% of vehicles undetected. In total, this leaves only 73 vehicles detected with 185 correct positions. The average precision of positions was 22.2 m^2.

The second configuration shows much better results. The outcome of simultaneity constraint of 200 ms gives three false positives and less than 7.6% of false negatives. In total, 85 vehicles of 92 were detected with 281 correct positions. The average precision of positions was 17.1 m^2. It can be noticed that the average S_{loc} is getting more stable after the simultaneity constraint of 400 ms. This indicates that more precise average position is difficult to achieve (the reason for this could be that the frame start is chosen as timestamp for sensor readings, not actual sound event).

In conclusion, with the same number of false positives in experiments 3 and 10, the second experiment configuration with time-selective algorithm showed significantly less false negatives (decreased by more than six times). Experiment 10 also improves the $Average(S_{loc})$ by 23.0%. When using more strict constraints in either configurations (with experiments 4, 5 or 11), the constraints start to filter out too many detections. We conclude also that the constraints were too relaxed in experiments 1, 2, 3, 6, 7, 8 and 9. The task of this article was not to find best configuration, but to give indication for the necessity of time-selective handling of input data for in-network processing.

Conclusion

Various situations exhibit physical phenomena which can be observed and measured with individual sensors practically simultaneously. The parallelism in the observation process is important, as it is otherwise difficult to combine these measurements during a fusion process later. We do not consider global clock synchronization feasible in an ad hoc WSNs, neither is the data transport time deterministic in such networks. The in-network data processing nodes receive packets out of order and with time varying delays, in addition sensor nodes themselves are unreliable. The purpose of this article was to show that using a time-selective strategy for in-network processing improves the temporal consistency of input data for SA information acquired in ad hoc WSN. To demonstrate the improvement, we used distributed autonomous sensors for detection of moving vehicles in an urban street. By applying the time-selective data fusion strategy, the fusion algorithm is able to select temporally compatible data from the arriving streams of sensor data. The data which satisfy simultaneity constraints have a higher probability for describing the observed situation accurately. After alignment and selection of temporally compatible data, the data still need to be checked against spatial constraints. As the data fusion considered in this article computes the position from distributed observations, the spatial check is done by the fusion process. Spatial constraints were briefly discussed in our earlier paper,[4] and an idea of combining of temporal and spatial constraints has been discussed in Mõtus et al.[39] This area is a topic for a separate research paper.

We also consider the time-selective strategy generic enough to be applied in ad hoc WSNs regardless of media access control and link layer protocols. Furthermore, we consider that it is worth to research whether the time-selective data fusion strategy improves the effectiveness of filtering-based multi-sensor multi-lag OOSM approach for WSNs.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was partially funded by the project SMENETE (Smart Environment Networking Technologies), which is funded by the European Regional Development Fund within the framework of EU Smart Specialisation programme 2014–2020 and the Estonian IT Academy program.

ORCID iD

Jaanus Kaugerand https://orcid.org/0000-0001-8678-0295

References

1. Preden JS, Kaugerand J, Suurjaak E, et al. Data to decision: pushing situational information needs to the edge of the network. In: *Proceedings of the 2015 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision*, Orlando, FL, 9–12 March 2015, pp.158–164. New York: IEEE.
2. Endsley MR. *Designing for situation awareness: an approach to user-centered design*. Boca Raton, FL: CRC Press, 2016.
3. Diallo O, Rodrigues JJ and Sene M. Real-time data management on wireless sensor networks: a survey. *J Netw Comput Appl* 2012; 35(3): 1013–1021.
4. Ehala J, Kaugerand J, Pahtma R, et al. Situation awareness via internet of things and in-network data processing. *Int J Distrib Sens N* 2017; 13(1): 686578.
5. Preden JS, Motus L, Pahtma R, et al. Data exchange for shared situation awareness. In: *Proceedings of the 2012 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision support (CogSIMA)*, New Orleans, LA, 6–8 March 2012, pp.198–201. New York: IEEE.
6. Yamaguchi A, Watanabe Y, Sato K, et al. In-vehicle distributed time-critical data stream management system for advanced driver assistance. *J Inform Process* 2017; 25: 107–120.
7. Cugola G and Margara A. The complex event processing paradigm. In: Colace F, De Santo M, Moscato V, et al. (eds.) *Data management in pervasive systems*. Cham: Springer, 2015, pp.113–133.
8. Abdelgawad A and Bayoumi M. Data fusion in WSN. In: Ahmed A and Magdy B (eds) *Resource-aware data fusion algorithms for wireless sensor networks*. New York: Springer, 2012, pp.17–35.
9. Li Y and Bartos R. A survey of protocols for intermittently connected delay-tolerant wireless sensor networks. *J Netw Comput Appl* 2014; 41: 411–423.
10. Mõtus L and Rodd MG. *Timing analysis of real-time software*. New York: Elsevier, 1994.
11. Izadi D, Abawajy JH, Ghanavati S, et al. A data fusion method in wireless sensor networks. *Sensors* 2015; 15(2): 2964–2979.
12. Al-Anbagi I, Erol-Kantarci M and Mouftah HT. A survey on cross-layer quality-of-service approaches in WSNS for delay and reliability-aware applications. *IEEE Commun Surv Tut* 2016; 18(1): 525–552.
13. Zhao G. Wireless sensor networks for industrial process monitoring and control: a survey. *Netw Protocol Algorithm* 2011; 3(1): 46–63.
14. Hiromori A, Uchiyama A, Yamaguchi H, et al. Deadline aware data collection in CSMA/Ca-based multi-sink wireless sensor networks. In: *Proceedings of the 6th international conference on mobile computing and ubiquitous networking (ICMU)*, vol. 12, Fukuoka, Japan, 23–24 May 2012, pp.284–294. Tokyo: IPSJ.
15. Cheng CT, Leung H and Maupin P. A delay-aware network structure for wireless sensor networks with in-network data fusion. *IEEE Sens J* 2013; 13(5): 1622–1631.
16. Wang Y, Vuran MC and Goddard S. Cross-layer analysis of the end-to-end delay distribution in wireless sensor networks. *IEEE/ACM T Network* 2012; 20(1): 305–318.
17. Oliver RS and Fohler G. Probabilistic estimation of end-to-end path latency in wireless sensor networks. In: *Proceedings of the IEEE 6th international conference on mobile Ad hoc and sensor systems (MASS'09)*, Macau, China, 12–15 October 2009, pp.423–431. New York: IEEE.
18. Preden J, Llinas J, Rogova G, et al. On-line data validation in distributed data fusion. In: *Proceedings of the international conference on SPIE defense, security, and sensing*, vol. 8742. Baltimore, MD, 22 May 2013. Bellingham, WA: SPIE.
19. Kopetz H. *Real-time systems: design principles for distributed embedded applications*. New York: Springer, 2011.
20. Maroti M and Sallai J. Packet-level time synchronization. Technical report, TinyOS Core Working Group, Berkeley, CA, 2008.
21. Carrano RC, Passos D, Magalhaes LC, et al. Survey and taxonomy of duty cycling mechanisms in wireless sensor networks. *IEEE Commun Surv Tut* 2014; 16(1): 181–194.
22. Khaleghi B, Khamis A, Karray FO, et al. Multisensor data fusion: a review of the state-of-the-art. *Inform Fusion* 2013; 14(1): 28–44.

23. Liu Y, Wang X, Liu Y, et al. Data-aware retrodiction for asynchronous harmonic measurement in a cyber-physical energy system. *Sensors* 2016; 16(8): 1316.
24. Sun S, Lin H, Ma J, et al. Multi-sensor distributed fusion estimation with applications in networked systems: a review paper. *Inform Fusion* 2017; 38: 122–134.
25. Shi Y, Wan Y and Ge Q. A unified out-of-sequence measurements fusion algorithm for WSN. In *Proceedings of the 1st international workshop on database technology and applications*, Wuhan, China, 25–26 April 2009, pp.76–79. New York: IEEE.
26. Xiaoliang F, Chenglin W and Lizhong X. Multi-sensor multi-OOSM distributed sequential fusion filtering. In: *Proceedings of the 10th IEEE international conference on control and automation (ICCA)*, Hangzhou, China, 12–14 June 2013, pp.77–776. New York: IEEE.
27. Yadav SS, Chitra A and Deepika CL. Reviewing the process of data fusion in wireless sensor network: a brief survey. *Int J Wireless Mob Comput* 2015; 8(2): 130–140.
28. Bahrepour M, Meratnia N and Havinga PJM. Sensor fusion-based event detection in wireless sensor networks. In: *Proceedings of the 6th annual international mobile and ubiquitous systems: networking services (MobiQuitous)*, Toronto, ON, Canada, 13–16 July 2009, pp.1–8. New York: IEEE.
29. Lai S, Cao J and Fan X. Ted: efficient type-based composite event detection for wireless sensor network. In: *Proceedings of the 11th international conference on distributed computing in sensor systems (DCOSS)*, Barcelona, 27–29 June 2011, pp.1–8. New York: IEEE.
30. Jacoub JK, Liscano R and Bradbury JS. A survey of modeling techniques for wireless sensor networks. In: *Proceedings of the 5th international conference on sensor technologies and applications (SENSORCOMM)*, vol. 2011, Nice, 21–27 August 2011, pp.103–109. IARA XPS Press.
31. Jhumka A, Bradbury M and Saginbekov S. Efficient fault-tolerant collision-free data aggregation scheduling for wireless sensor networks. *J Parallel Distr Com* 2014; 74(1): 1789–1801.
32. Dearle A, Balasubramaniam D, Lewis J, et al. A component-based model and language for wireless sensor network applications. In: *Proceedings of the 32nd annual IEEE international conference on computer software and applications (COMPSAC'08)*, Turku, 28–1 August 2008. pp.1303–1308. New York: IEEE.
33. Eidson JC, Lee EA, Matic S, et al. Distributed real-time software for cyber–physical systems. *Proc IEEE* 2012; 100(1): 45–59.
34. Mõtus L, Meriste M and Preden JS. Towards middleware based situation awareness. In: *Proceedings of the IEEE military communications conference (MILCOM)*, Boston, MA, 18–21 October 2009, pp.1–7. New York: IEEE.
35. Preden JS, Llinas J and Motus L. Middleware for exchange and validation of context data and information. In: Snidaro L, García J, Llinas J, et al. (eds) *Context-enhanced information fusion*. New York: Springer, 2016, pp.205–230.
36. Sarvghadi MA and Wan TC. Message passing based time synchronization in wireless sensor networks: a survey. *Int J Distrib Sens N* 2016; 12(5): 1280904.
37. Djenouri D and Bagaa M. Synchronization protocols and implementation issues in wireless sensor networks: a review. *IEEE Syst J* 2016; 10(2): 617–627.
38. Cruz JRP and Hernandez SEP. Temporal alignment model for data streams in wireless sensor networks based on causal dependencies. *Int J Distrib Sens N* 2014; 10(3): 938698.
39. Mõtus L, Preden JS, Meriste M, et al. Self-aware architecture to support partial control of emergent behavior. In: *Proceedings of the 7th international conference on system of systems engineering (SoSE)*, Genova, 16–19 July 2012, pp.422–427. New York: IEEE.

Appendix 2

II

J. Ehala, **J. Kaugerand**, R. Pahtma, S. Astapov, A. Riid, T. Tomson, J.-S. Preden, and L. Mõtus. Situation awareness via internet of things and in-network data processing. *International Journal of Distributed Sensor Networks*, 13(1):1–21, 2017

Research Article

International Journal of Distributed Sensor Networks

International Journal of Distributed Sensor Networks
2017, Vol. 13(1)

DOI: 10.1177/1550147716686578
journals.sagepub.com/home/ijdsn
SAGE

Situation awareness via Internet of things and in-network data processing

Johannes Ehala, Jaanus Kaugerand, Raido Pahtma, Sergei Astapov, Andri Riid, Timo Tomson, Jürgo-Sören Preden and Leo Mõtus

Abstract

Computing on the edge of the Internet of things comprises among other tasks in-sensor signal processing and performing distributed data fusion and aggregation at network nodes. This poses a challenge to distributed sensor networks of low computing power devices that have to do complex fusion, aggregation and signal processing in situ. One of the difficulties lies in ensuring validity of data collected from heterogeneous sources. Ensuring data validity, for example, the temporal and spatial correctness of data, is crucial for correct in-network data fusion and aggregation. The article considers wireless sensor technology in military domain with the aim of improving situation awareness for military operations. Requirements for contemporary intelligence, surveillance and reconnaissance applications are explored and an experimental wireless sensor network, designed to enhance situation awareness to both in-the-field units and remote intelligence operatives, is described. The sensor nodes have the capability to perform in-sensor signal processing and distributed in-network data aggregation and fusion complying with edge computing paradigm. In-network data processing is supported by service-oriented middleware which facilitates run-time sensor discovery and tasking and ad hoc (re)configuration of the network links. The article describes two experiments demonstrating the ability of the wireless sensor network to meet intelligence, surveillance and reconnaissance requirements. The efficiency of distributed data fusion is evaluated and the importance and effect of establishing data validity is shown.

Keywords

Distributed data fusion, distributed data aggregation, data validity, in-sensor computation, Internet of things, wireless sensor networks, edge computing, situation awareness, middleware

Date received: 15 July 2016; accepted: 26 November 2016

Academic Editor: Ping Yi

Introduction

Tens of billions of devices connected by Internet of things (IoT), which analysts predict will be deployed by 2020, will operate in our environment, enhancing our capability for acquiring real-time data for decision making and automating mundane tasks. The lowest layer of IoT will operate on low-power, low-bandwidth embedded networks, which conventionally have been called wireless sensor networks (WSNs). Due to the rapid development and spread of embedded computer technology over the last decade, sensor nodes are widely used and produce potentially abundant information for IoT applications. However, the disconnected, intermittent and limited (DIL) communication environment that these nodes often operate in make the usage of Internet-level communication solutions not applicable on the WSN level.

Research Laboratory for Proactive Technologies, Tallinn University of Technology, Tallinn, Estonia

Corresponding author:
Jaanus Kaugerand, Research Laboratory for Proactive Technologies, Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia.
Email: jaanus.kaugerand@ttu.ee

In order to manage the large data flows and to minimise the bandwidth requirements, novel paradigms such as data to decision (D2D) and mist computing (an extension of fog computing) need to be exploited.[1] Traditional data aggregation[2] is a predecessor of these paradigms, but in its classical form is not enough for IoT applications, because the traditional flow of data from network edge (sensor nodes) to centre (databases) remains. According to the D2D concept,[3] relevant data are identified in the network and delivered to decision makers and fashioned to their current data needs, which are derived from the specific decisions that they need to make. Mist computing characterises the architecture where computation occurs not only in the cloud but also in end nodes, such as sensor nodes.[4] Combining the D2D approach with the Mist Computing paradigm allows to utilise a large number of sensing nodes while overcoming technical bandwidth challenges and providing the consumers the required situation information with reasonable use of network resources. This article presents a military purpose WSN that follows these paradigms, brings computation to the edge of the network and makes results available to users directly, without being dependent on the cloud.

Successful deployments of WSNs have been demonstrated in a range of domains, for environmental monitoring applications in rural[5] and urban[6] areas, for infrastructure (bridges, buildings) structural health monitoring,[7,8] for early detection of natural disasters (such as forest fires,[9] landslides[10] and volcano eruptions[11]), for industrial monitoring[12] and for various specific tasks such as object tracking,[13] perimeter or object security monitoring[14] and patient health monitoring.[15] In most WSNs, the data acquired by sensors are communicated and collected to a central server, where the data are processed and made available to potential users. Although actual communication paths for data are established at run-time, the overall structure of the network, types of data and structure of the central database are known at design time. Typically, in such scenarios, data processing, fusion and analysis are done at the server database level, not in the WSN itself. Inside the network data are usually processed only for aggregation, mostly for the purposes of optimising sensor power usage and network throughput.

A growing number of contemporary IoT applications require more from WSNs than simple data acquisition, (conditional) communication and collection to databases. For example, military applications, such as intelligence, surveillance and reconnaissance (ISR) systems, aim to improve the situation awareness of decision makers and expect pre-processed data that are already converted to human understandable form. Data collection to central databases, as used in typical WSNs for monitoring, creates overhead, potential bottlenecks and offers limited resilience. An additional aspect is that some of the potential WSN users, such as in-the-field military units, need situational information in a timely manner and would prefer to receive information tailored to their current information needs directly from the network to minimise delays and dependence on central infrastructure.

We outline the general concept and design choices of our solution for a military WSN that provides situation awareness to users on different hierarchical levels (from tactical to strategic operation). The architecture and operating concept follows a service-oriented approach and publish–subscribe principles. As such, users (e.g. in-the-field military units, autonomous network nodes such as unmanned aerial vehicle (UAV) and command centre analysts) task the sensor network directly and subscribe to data of their interest. There is no requirement to collect all raw data to a central database and access the data through this database, although a database can still be created and used (e.g. for post-operation analysis). This greatly contrasts the common approach to environment monitoring WSNs, where users access data as clients of a central database. The service-oriented architecture and publish–subscribe principles fit well for WSNs designed to operate in tactical military settings, considering the highly unreliable communication links and persistent shortage of bandwidth that these networks encounter. Allowing users to directly interact with sensors constrains the network less than general network-wide data collection and distribution via a central database.

Figure 1 presents a general overview of the different functionality of the designed WSN. Signal acquisition and initial signal processing are conducted on sensor nodes. The produced data are then transmitted to interim fusion or aggregation nodes which combine the distributed data into new data types and/or structures. It is possible to have several fusion levels before the results are presented to the end user. Three different in-sensor signal processing cases are demonstrated. First, sensor nodes capable of audio signal acquisition compute the angle of arrival (AoA) of measured sound waves based on the time difference of arrival (TDOA) method. Second, some of these sensors are also capable of performing in situ fuzzy classification of the measured sound source (we are classifying vehicles based on the sounds they emit). And third, a camera-equipped sensor node performs video analysis with the goal of locating and counting mobile foreground objects (personnel) captured on the video.

In-network data fusion and aggregation are demonstrated by special network nodes (fusion or aggregation nodes). Fusion nodes collect AoA estimates from sensor nodes and calculate the location of the sound source from the intersection of beams formed from the AoA estimates. We consider this to be in-network data fusion, since a new data type (location coordinate

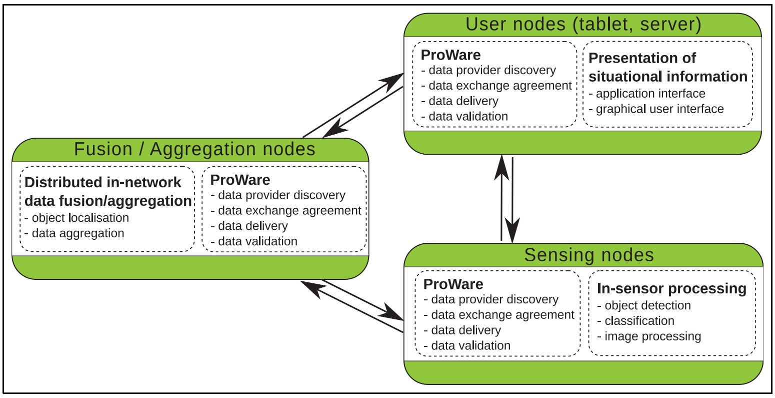


Figure 1. Overview of the functionality of network nodes. Arrows indicate possible communication flows. A service-oriented middleware (ProWare) handles communication between nodes.

estimate) was created from another, different type of input data (AoA estimates). Location coordinate estimates can then be combined with fuzzy classification results to form new meaningful data structures (e.g. a data bundle comprising the classification result and location of a detected object). We consider this to be data aggregation, since data of different types are meaningfully grouped together and presented in a human understandable form. Additionally, data validity checking, necessary for the fusion and aggregation processes, is discussed.

Evaluating the efficiency of a military WSN is a challenging task, since performance depends highly on the configuration and size of the network, operational circumstances and usage (changing number of users and their requirements). However, we present experimental data of bandwidth usage in an urban setting and evaluate the efficiency of our signal processing and fusion algorithms in another experiment in military settings.

The WSN experiment presented in this article is still at a TRL 3–5 level. Therefore, some important topics, for example, energy-related issues such as energy conservation, usage optimisation and harvesting; scalability issues such as message routing, multi-hop communication and network congestion; security issues required by ISR systems such as message encryption and node hijacking resistance; and sensor synchronisation are not discussed in this article. However, we refer the reader to Egner et al.[16] and Turkmen et al.[17] for an overview of the possible security solutions.

Section 'ISR requirements and situation awareness for military WSN' describes ISR and military WSN requirements. Section 'Related work' gives a short overview of related work. Section 'In-sensor signal processing' presents three different instances of in-sensor signal processing. Section 'Distributed data fusion and aggregation' discusses in-network distributed data fusion, aggregation and data validity checking. Section 'Communication architecture and principles of ProWare' describes the proposed communication solution that meets (selected) military ISR and D2D requirements. Section 'Demonstrations and experiments' describes two field tests, one analysing the overall operation of the proposed WSN and the other focuses on bandwidth usage and quality of service. Section 'Conclusion' concludes the work.

ISR requirements and situation awareness for military WSN

The specific operational and functional requirements of military WSNs and overlying ISR systems call for a different architectural design and adaptive topology of WSNs compared to civilian examples. This section reviews some of these requirements. At the core of most of these requirements lies the need to have a situational understanding of events taking place in a monitored area. The situations (or rather the events) can be very versatile and numerous and it is not always clear at design and deployment time, what events can occur and need to be detected. Therefore, the section starts with a short explanation of how situations are handled and described.

We refer to our previous work on situations and give the definition of a situation as:[18]

> a situation is the aggregate of biological, psychological, socio-cultural, and environmental factors acting on an individual or a group of agents to condition their behavioral patterns. Here agent denotes natural (e.g. humans) or artificial (e.g. computing systems, or software-intensive multi-agents) agents, and environment means mix of natural or artificial environments.

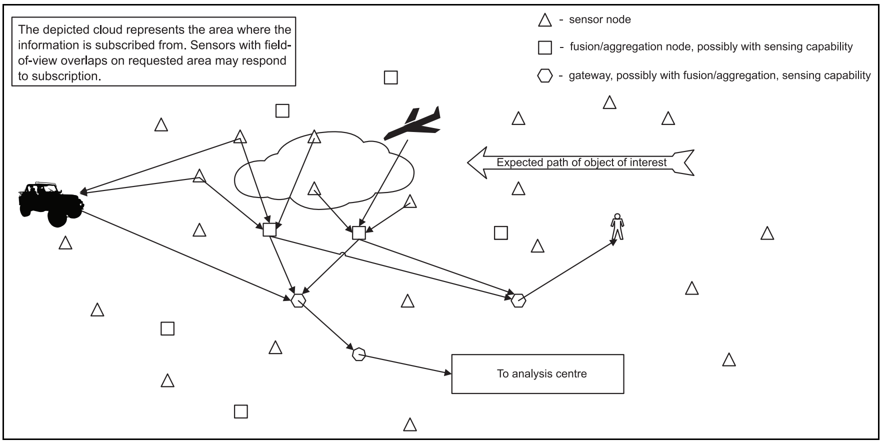


Figure 2. WSN deployment and dynamic formation of network links at run time according to the required situational information. Military units, such as ground patrols and unmanned aerial vehicles, acquire data directly from network nodes when in vicinity. When requested, data are also forwarded to analysis centre for online and/or offline analysis.

Situations are treated hierarchically and defined by 3-tuples $S = \{S_p, S_t, S_a\}$, where S_p denotes a set of situation parameters and S_t and S_a comprise, respectively, temporal and spatial information about the situation. The parameters can be numeric variables or other situations, representing the hierarchical nature of situations. For example, the temperature of an area is a basic situation $S_{temp} = \{S_p, S_t, S_a\}$, where S_p is a unit set holding the actual temperature value. A similar construct can be made for humidity S_{hum}. A higher level situation is composed of other situations either by fusion, for example, $S_{operation} = \{S_p, S_t, S_a\}$, where $S_p = f(S_{temp}, S_{hum})$ is a function of temperature and humidity situations, or by aggregation $S_{weather} = \{S_p, S_t, S_a\}$, where S_p is a two element set $S_p = \{S_{temp}, S_{hum}\}$. Here, $S_{operation}$ is for instance the suitable environmental conditions for the operation of some device and $S_{weather}$ is a set of weather parameters. Temporal and spatial information, S_t and S_a, are defined based on application needs and may include different information, for example, constraints, requirements and/or other data necessary for the interpretation of situation parameters S_p. Further examples of defining situations are given in section 'Demonstrations and experiments'.

The hierarchical principle for defining situations allow for a very dynamic and open system of situations to be created. This suits well with the changing nature of military situations and supports the design of military WSNs. The notation allows for general discussions over WSN requirements and feasibility at the preliminary design phase and technical discussions at the detailed design phase, while also transferring seamlessly between the different phases, when changes need to be made. High-level WSN architecture can be decided without the need to specifically define S_p, S_t and S_a, these are only determined at the detailed design phase, where all relevant factors (environmental, technological, etc.) are taken into account.

The requirements for ISR situational information are as diverse as are the units and the levels of hierarchy in the military organisation. This article concentrates on in-the-field units who need situation awareness information relevant to them in a timely manner, preferably directly from network nodes as depicted in Figure 2 and in human understandable form. Information for situational awareness therefore needs to be provided already at WSN level utilising the capabilities of network nodes. Emphasis is put on requirements that are derived from highly dynamic ISR military situations and the corresponding complex data flows within WSNs. We exploit the design paradigms of mist computing and D2D paradigm which are about pushing computation to the edge of IoT network and bringing correct data in timely manner to the right decision makers.

One of the main requirements, which differentiates military WSNs from typical civilian special purpose (scientific and commercial) WSNs, is the need for highly dynamic network structure – the ability to add or remove nodes and reconfigure communication paths on the go. In ISR applications, information collection is context based (i.e. constraints for information are contextual), therefore precise data requirements for tactical operations of military units are not known before WSN deployment and often change dynamically during operation. A WSN with fixed functionality and structure cannot answer to the changing needs. Sensors and communication paths for specific situational information need to be chosen and formed on ad hoc basis. Sensors may also leave these networks, as their power resources become depleted or they get destroyed, and new sensors, possibly with different functionality, may join the network according to the changing demands of the overlying ISR system. It is therefore not possible to design a complete sensor network for ISR applications, with fixed configuration, structure and predefined data flows. Rather, the system must accommodate dynamic run-time sensor discovery, tasking of data providers, that is, identifying and subscribing to available data sources (sensor and/or other information providers) and ad hoc formation of communication paths to cope with the changing goals and environment. As is described in section 'Communication architecture and principles of ProWare', sensors do not subscribe to data from specific data providers with their unique identification numbers, but rather subscribe to data according to its type, location, time and other parameters that describe (and give context to) the needed situational information.

Another requirement, from the point of in-the-field military units, is to be able to acquire data directly from nearby network nodes rather than connecting to a remote database. The reasons for this are that connecting to a remote database takes time, communication can be intermittent and the database may not have the latest data. Also all WSN nodes would have to constantly update the database, which consumes network bandwidth and takes time. As an alternative, distributed data aggregation and fusion must be performed at network node level to provide the necessary situational awareness to military units. Sensor and other nodes must be capable of carrying out the required signal processing, data fusion and aggregation calculations, while at the same time assuring that the data used are mutually conforming.

Performing in-network data fusion and aggregation in timely manner requires consistency of data collected from different sensor nodes in both temporal and spatial domains. In order to ensure the validity and usability of fusion and aggregation results, constraints must be applied to collected data. Spatial constraints, such as bounds to the area of interest, and temporal constraints, such as acceptable age of data, are defined within the subscription made to the WSN by the user. In either case, data providers, the different sensor nodes, must append all measured data with appropriate temporal and spatial metadata tags, which are later used in the data validation process. In addition, network communication layer must support in-time packet delivery (within the pre-specified delivery interval) or inform data provider and consumer of failure to (temporarily) meet these requirements during operation. The WSN presented in this article utilises a messaging syntax and communication protocol for WSN that facilitates satisfying the above described data validity needs.[19,20]

The expected environmental conditions for tactical military WSNs are for the most part similar to those in typical civilian environment monitoring WSNs – sensor nodes are situated in harsh environments and need protection against the elements. Operational conditions, however, are different due to constantly changing military situation and the existence of malicious adversaries trying to disrupt network operation. Among other properties, it is desirable that sensor nodes be physically and electronically inconspicuous and if possible resistant to tampering, denial of service[21] and deception type of attacks. The latter properties that concern security and electronic warfare are outside the context of this article.

In conclusion, we identify five major requirements for tactical operation purpose WSNs:

1. Dynamic network structure and functionality is preferred along with ad hoc formation of communication paths.
2. Situation awareness should be created on WSN level.
3. Network nodes should be capable of in-sensor signal processing, distributed data aggregation and fusion.
4. It must be possible to assure that data are mutually conforming.
5. It should be possible to identify and task data providers at run-time using context-based data constraints.

The list is not conclusive but serves as a starting point for developing distributed in-network fusion systems. Sections 'In-sensor signal processing', 'Distributed data fusion and aggregation', 'Communication architecture and principles of ProWare' and 'Demonstrations and experiments' will present our solutions to these requirements and analyse the overall operation and feasibility of such a WSN.

Related work

We review related work in three parts, first focusing on general military WSN examples and requirements, then discussing signal processing on WSN nodes and finally reviewing distributed data fusion and aggregation in WSNs.

Common existing examples of military tactical WSNs operating on the edge of ISR networks are either highly specialised or based on commercial off the shelf systems (COTS) that are slightly ruggedised in terms of hardware and software, as compared to those used in civilian applications. (See, for example, the SPAN system from Lockheed Martin or the MicroObserver system from Textron Systems.)

Many civilian WSN, for example, those applied in manufacturing and industrial control, have reasonably good time-aware behaviour, that is, the ability to handle time-critical data and time-sensitive communication. Overviews of such systems can be found in Kopetz[22] and Zhao.[23] Providing information in a timely manner is a necessary property for military WSNs and having temporal knowledge of information communicated in the network is the basis of creating correct situation awareness. However, the drawback of majority of civilian applications is that WSNs are assumed to have a fixed structure, fairly reliable end-to-end communications, capability of time synchronisation in nodes and that they operate on fixed rules and goals.

Military tactical WSNs, as a rule, must cope with disruptive communication and random communication delays,[24] changing topology and composition of network,[3] dynamically changing rules and goals[25] and asynchronously operating heterogeneous nodes. Tactical WSNs must be disruption-tolerant networks (DTNs) that can cope with the fragmentary connectivity of nodes, no guarantee of successful end-to-end message transfer and malicious cyber-physical attacks.[26] The WSN experiment presented in this article does not specifically tackle the problem of unreliable communications, but network delays, disruptions and changes in topology are covered by the discussed service-oriented architecture and online data validity checking.

Due to listed discrepancies, COTS systems (and civilian WSNs) are most efficient in situations where the network operates in a stationary environment, for example, perimeter monitoring and control.[21] As one of the ways to tackle specific military requirements, several middleware solutions for WSNs have been developed (an example can be found in Pham et al.[25]) that implement service-oriented concepts known from Internet domain. Examples like publish–subscribe concept, service-oriented architectures and service provider discovery are gradually being implemented for WSNs. A survey of service-oriented middleware solutions for WSN can be found in Mohamed and Al-Jaroodi.[27]

In-the-field military units need timely situation awareness. Creation of situation awareness therefore starts on WSN level utilising the capabilities of the sensor network. Distributed data aggregation and fusion are the methods of choice in military context, since the alternative of remote data analysis requires central data collection, which consumes network bandwidth and time.

Signal processing in WSN sensor nodes

Features, such as in-sensor signal processing and in-network data aggregation and fusion, are not new to WSNs nor are they military WSN specific. Two audio signal processing cases are considered in this article – sound wave direction of arrival estimation and sound signal–based classification of signal source. Detecting the location of objects based on the different physical waves they emit into environment is typically based on the TDOA of these waves to detectors. Popular algorithms for this purpose are MUSIC and SRP-PHAT, both of which have been utilised also for WSNs.[28,29] The latter example also demonstrates that while a lot of existing systems use general-purpose computers for signal processing, examples with low computing power sensor devices also exist. Applications that benefit from those methods include target tracking and shooter localisation for instance.[30]

Classification of objects based on assessing characteristics of emitted sound signals is another well-studied signal processing area and its application in WSNs for low computing power devices is an emerging trend.[31–33] Until a short time ago, wireless sensor nodes were not powerful enough to perform feature extraction from measured signals and to run conventional classification algorithms. Although today's technology facilitates advanced signal processing, the issue of training the classifier (regardless of whether it uses neural networks, fuzzy classifiers or some other methods) remains. Classifier training is still hard to perform at run-time on currently available sensor devices or other WSN nodes. In our experiments, the training was done beforehand and sensor devices were deployed with the required class feature vectors installed.

Distributed data fusion and aggregation in WSN

WSN data aggregation techniques have developed hand in hand with the advancement and spread of WSN technology. The primary motivation behind data aggregation has been energy efficient data acquiring in order to extend network lifetime and enhance quality of service of WSN.[34] Different data collection and processing schemes arrange that not all data are individually transferred to the network sink, but instead related data are accumulated temporarily somewhere in the WSN, where it is aggregated and only the results are forwarded to the sink. This reduces the amount and length of messages passed and subsequently saves energy and bandwidth. Use cases of data aggregation in WSNs include Jo et al.[8] and Ramesh.[10] The data aggregation we present does not only serve the purpose of saving energy, it also aims at enhancing the situation awareness of data consumers by combining different types of sensor data characterising a particular event detected in the WSN. Majority of WSN in-network aggregation, however, focuses on aggregating only data of the same type.

Data fusion also serves the purpose of enhancing situation awareness of the data consumer. The difference between data fusion and data aggregation is that different data types of sensor readings are not combined into a bundle, but instead are fused into a new type of data.

Data fusion in WSNs can be carried out locally, where a sensor node comprising different types of sensors fuses the data acquired from sensor readings and communicates the fusion result to consumers. In the case of distributed data fusion, the data from respective sensor nodes are collected by a prefixed sensor/fusion node that performs the fusion process and distributes the result to data consumers. Examples of distributed data fusion in WSNs are described in Mayk et al.,[3] Bahrepour et al.[35] and Lai et al.,[36] where events detected and sensor readings collected by individual sensor nodes are assembled by a fusion node. These references do not discuss the consistency and validity of the inputs for the fusion algorithms. In case of large ad hoc sensor networks and especially networks for ISR solutions, it is important that the consistency and validity of the fused information is analysed online.[20]

In-sensor signal processing

Computing on the edge of IoT requires that the computation is moved from the cloud to the network and as much as possible to the sensors themselves. This section will give an overview of in-sensor signal processing and describe three different cases of in-sensor signal processing that we have currently utilised in our WSN experiments. This section gives an overview of the three different cases of in-sensor signal processing that were utilised in WSN experiments presented in this article. The three applications were identifying and counting objects in video-streams, classification of objects based on sound signals and estimating the direction of arrival of sound signals (to sensor node). The choice of applications was motivated by ISR requirements, for example, the need to detect, count, classify and position objects (events) found in the environment. An example of a use-case scenario for these applications is described in section 'Demonstration of system operation in military setting'. For each application presented in this section, signal processing was done locally on appropriate sensor nodes using local data acquired by the node itself. No additional information or data from outside were needed once operation had started.

The difficulty of performing in-sensor signal processing lies mainly in efficiently coping with the limited resources and constrained computational power of WSN computing devices. The specifications and hardware of used sensor nodes is presented in section 'Demonstration of system operation in military setting'.

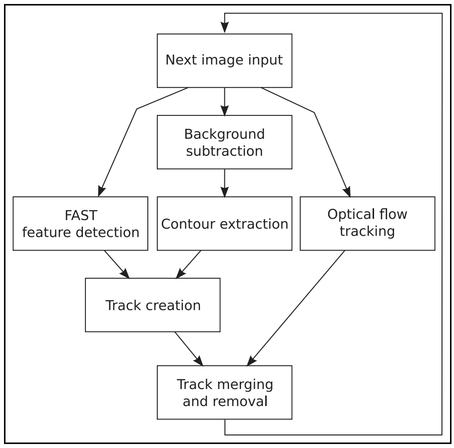


Figure 3. Principal steps of the object counting algorithm.

Automatic object counting

Information extraction by image processing is a resource-intensive task and is usually infeasible in the resource-limited WSN nodes at the edge of IoT. The described sensor node is equipped with a camera and applies signal processing methods to count moving objects by separating mobile foreground objects (e.g. people) from relatively static background in the field of view (FOV) and counts the average number of those foreground objects in a prefixed time period. This method does not classify detected objects and thus requires less computing resources. It can be implemented even in low-power sensor nodes. The output of the sensor is the count of detected objects and the temporal interval between the object detections. This result is communicated to the subscriber of these data (e.g. a fusion node).

The working principle of the object counting algorithm is to follow objects that are considered to be foreground of an image. The movement history of these objects is stored as a vector within a 'Track' data structure, one element for every input frame while the object is visible. Each element includes a convex contour around a foreground object and feature points within that contour that can be used to follow the object.

The image processing software combines well-known algorithms implemented in OpenCV library. Steps of the algorithm are shown in Figure 3. Every input frame is fed to the adaptive Gaussian mixture model background subtraction algorithm.[37] Algorithm proposed in Suzuki and Abe[38] is used on the resulting binary image of estimated foreground areas to find contours of foreground objects. An example is depicted in Figure 4(a), where foreground objects are circled with red contours. Contours that are too small, large or unusually shaped are rejected. The initially found contours will be replaced by convex hulls (blue

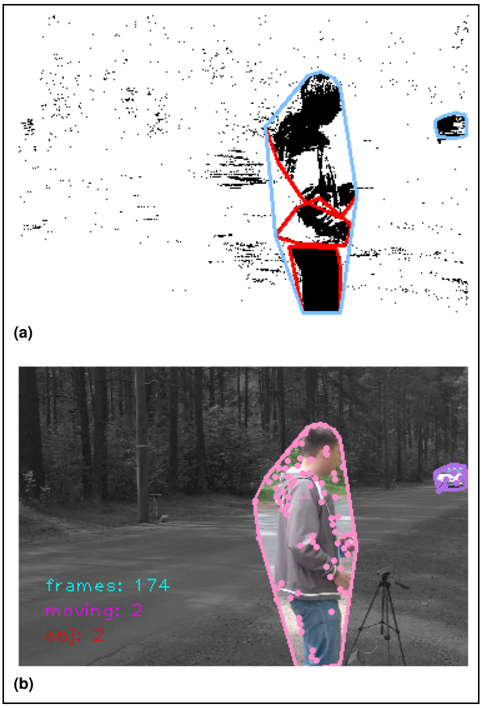


Figure 4. A video frame after different image processing steps: (a) frame after background subtraction and contour creation around foreground objects (black areas – foreground objects; red contours – initial contours; blue contours – convex hull of initial contours) and (b) black-and-white frame with two Track objects (pink and purple). A Track object consists of a convex contour and feature points for tracking associated with that contour.

contours in Figure 4(a)) around those contours.[39] This simplifies some of the following calculations. Meanwhile, a FAST corner detection algorithm[40] finds good feature points to track. Feature points that fall within convex contours are combined with those contours to create new Tracks (see Figure (b)). If the contour of a new Track has overlap with existing Tracks, then it will be merged into the older Track with the greatest intersection (notice the different shape of a Track contour, pink in Figure 4(b), and a convex contour, blue in Figure 4(a)). Merging combines contours and feature points of both Tracks. Tracks will be updated by following their feature points using optical flow algorithm (implementation proposed by Bouguet[41] based on Lucas and Kanade[42] and Lucas[43]).

Acoustic signal-based fuzzy classification

In-sensor classification of objects (different military vehicles in this case) is performed by analysing the sound signals emitted by the objects of interest. Classification is one of the basic tasks in pattern recognition and data analysis, and it is performed by analysing the training data set to develop an accurate class description or model for each object present in the training data set. These models are then used to establish class labels to new data for which the class labels are unknown. The training data set that is collected from previous experiments with known vehicle types consists of multiple instances each tagged with a class label and having multiple attributes. For classification purposes, we use a fuzzy rule-based classification algorithm,[44] because of its ability to deal with imprecise data, flexibility of the decision boundaries, because it has low resource requirements and the generated rules can be interpreted by a human if needed.

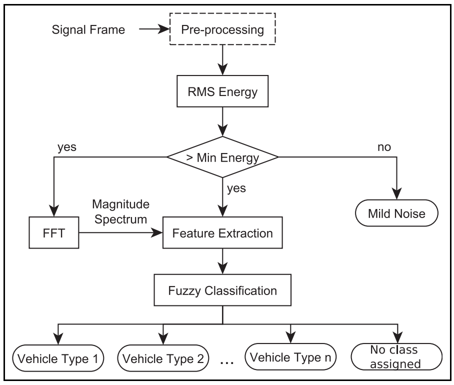


Figure 5. Classification procedure.

The classification procedure runs on microphone sensor nodes in parallel with AoA calculations described in section ‘AoA of acoustic sound waves’. The classification sequence is depicted in Figure 5, it takes as input a single acoustic signal frame from one of the microphones of the sensor array. The process starts with signal acquisition – the continuous analogue signal from the microphone is sampled and quantized by an analogue-to-digital converter (ADC). The one-dimensional digital acoustic sound signal does not provide rich enough information context by itself so it undergoes a procedure of signal analysis both in time domain and frequency domain to provide relevant attributes calculated for each short signal frame of 2730 samples, sampled at 20 kHz. Time domain signal analysis focuses on the shape and amplitude of a signal (in our case the root mean square energy is computed) and is well applicable to weakly oscillating and harmonic signals. If the signal is non-harmonic or highly polluted by noise, which foremost influences the signal amplitude and shape, time domain features (such as zero crossing rate, auto-correlation and root mean square energy) are less useful.

The acoustic noise patterns that can be collected with sensors in the vicinity of moving vehicles consist of multiple components, including noise produced by the engine, exhaust system and tires, ambient noise from wind and rain may also be present. The harmonic nature of the engine noise is therefore seldom detectable and parameters of the overall spectral shape and energy distribution describing the vehicle noise patterns are used. These may include choosing band energies of 10–15 sub-bands in the interval of 10–3000 Hz, spectral centroid, spectral roll-off, spectral slope parameters and so on. The feature extraction both in time and frequency domains is explained in our previous works.[45,46]

The sound patterns of passing vehicles are not consistent and depend on the distance, trajectory and type of a vehicle. As the vehicles under test produce a significant amount of noise, it is acquired even from a distance where the patterns are distorted and can cause classification errors. In order to minimise these errors, a minimum signal root mean square energy threshold is selected, which must be exceeded for the classification process to start.

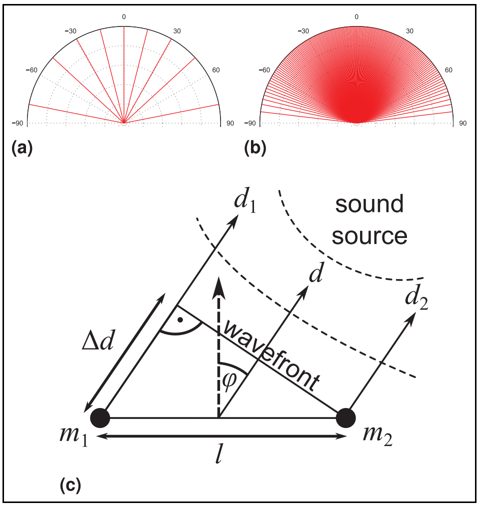


Figure 6. Sensor field-of-view sensitivity based on sampling speed and microphone distance (top) and angle of arrival of sound waves (bottom): (a) 4 kHz sectors, (b) 20 kHz sectors and (c) sound-wave arrival and angle orientation φ.[30]

AoA of acoustic sound waves

In addition to being able to detect, count and classify objects of the environment, it is desirable from ISR perspective to be able to estimate the location of detected objects. Location estimation is also performed based on the noise (sound signals) that objects emit and is computationally divided into two parts: estimating the AoA of sound waves at individual sensor nodes and combining the individual AoA estimates of several nodes into a single (or multiple) location estimates. The latter part is discussed in section 'Location estimation'.

The direction of arrival of sound waves to a sound sensor is found using at least two microphone elements, placed at different locations, and calculating the TDOA of sound waves to either microphone. By knowing the location of either microphone, the TDOA and the approximate speed of sound waves, it is possible to find the direction towards the source of the sound waves. TDOA is found by cross-correlation of the measured sound signals, it is the delay between the signals. Once the delay is known, the direction is calculated as

$$\varphi = arcsin\frac{\Delta k/f_s \cdot c(T)}{l}$$

where Δk represents the delay in samples, f_s is sampling frequency, l is distance between microphones and $c(T)$ is the speed of sound waves in air, which is a function of air temperature T. The orientation of angle φ, with regard to microphones, can be seen in Figure 6(c). The quality of the result mostly depends on the sampling rate and distance between microphones.

The method is implemented on two different platforms. The first, Atmel ATmega128RFA1 microcontroller based platform, has two microphones and samples each microphone at 4 kHz. The second, BeagleBoneBlack (BBB)-based platform, has six microphones and samples each microphone at 20 kHz. The six microphones are placed linearly with equal distance from each other and the extra number of acquired signals helps increase the accuracy of cross-correlation. The FOV of the sensor is 180° for both platforms. It is divided into discrete, non-equal segments as depicted in Figure 6(a) and (b). Sampling speed and distance between microphone pairs determines the number of segments. A sampling speed of 20 kHz enables a much denser segmentation of the FOV and thus the accuracy of angle estimation is much better for BBB platform.

Distributed data fusion and aggregation

In-sensor signal processing constitutes one half of computing in the edge of IoT, the other half being data fusion and aggregation. In our interpretation, the latter two differ from signal processing by the fact that for fusion and aggregation data are gathered from multiple functionally disparate and spatially distributed sources and need to be checked for compatibility before processing. Compatibility is currently checked against temporal and spatial parameters of the collected data, that is, the data from different sensors must originate from the same physical area and time period if it is to describe the same environmental event (or object). Of course other parameters can be considered, depending on the needs and requirements of the system. For example, adding confidence estimations to complicated sensor measurements (such as classification) or accuracy values (for AoA estimations) so that the fusion and aggregation mechanisms can pick the most reliable data for processing.

An example of distributed data fusion, presented in this section, is calculating the location of objects detected in the environment by angle estimates received from different sensor nodes. Distributed data fusion distinguishes from data aggregation in our work by the fact that a new data entity (object location, a geographical coordinate) is defined using other data types (angles and sensor node coordinates) as input. Distributed data aggregation is considered to be combining different data from different sources, which describe the same environmental event,into a meaningful bundle, but no new data are created. This approach expands traditional data aggregation in WSN data collection applications, where for bandwidth optimisation purposes usually only the same type of data is aggregated (i.e. at certain nodes in the collection chain the mean, minimum, maximum or other values are found for the collected data and only those results are forwarded).

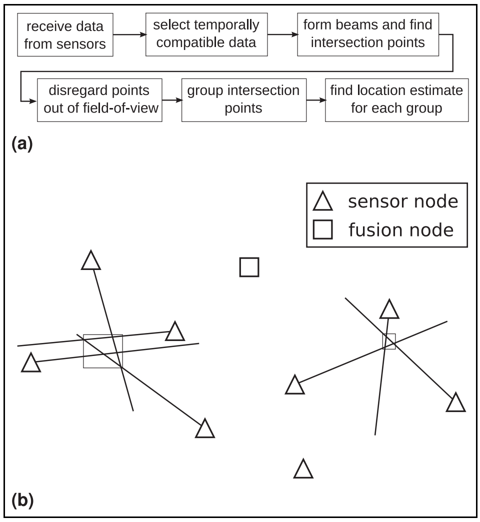


Figure 7. (a) Location estimation process and (b) example of location estimation in WSN for two sound events.

Location estimation

Individual microphone array sensors alone can estimate the direction to a sound source from their position, but cannot effectively determine the distance to, and therefore also the location of, the source. A location estimate can be established, however, by several sensors in the same area by combining their direction estimates. Special fusion nodes are dedicated to this task, although in principle any sensor or other type of node can take up this task, if it has the necessary resources.

The fusion process is depicted in Figure 7(a). First data are collected from all sensor nodes, which have detected a sound event. The data include the location of the sensor node (geographical coordinates), the measured direction estimate (a geographic bearing) and metadata such as sensor sensing range and a timestamp indicating the age of the measurement. Based on the age of each measurement, compatible sound event instances are found and only these are analysed together. Next, beams are formed along all the direction estimates and intersection points of these beams are found. Due to the discrete nature of AoA calculation procedure (see section 'AoA of acoustic sound waves') and other inaccuracies of input data, all the beams will never intersect in a single point. Rather, a cluster of intersection points emerges and the density or sparsity of this cluster determines whether the result should be considered a valid location estimate or not. It is also checked that intersection points fall within the FOV of the involved sensors, intersection points out of range of the sensors are not considered. The forming of clusters of intersection points may also be guided by classification results if they are provided together with AoA estimates. This enables fusion algorithm to separate the intersection points according to provided classes.

An example of a location estimation situation in a WSN is depicted in Figure 7(b). Two sound events are detected at the same time, one on the left side of the WSN by four sensor nodes and the other on the right side by three nodes. All seven sensors forward their direction estimates to the fusion node in the middle, which ideally should separate the estimates into two groups, left and right. There are many ways to do this and in our approach we consider the sensing range of sensors coupled with their direction estimates, to find nodes with converging results. Currently, the location estimation result is represented as a rectangular area encircling the cluster of intersection points, as depicted in Figure 7(b), from this cluster, a single geographical coordinate can be computed, which is a weighted average of the intersection points in the cluster.

However, the location estimations of moving sound sources and separating between several different sound sources may still be opportunistic and requires further analysis due to the nature of sound wave propagation and rapidly changing environment. The exact limitations have not been tested during the experiments described in this article.

Aggregation

Only one aggregation combination is used in the WSN experiment, that is, combining three different types of information from different sources into a bundle. The information that is aggregated is the number of detected objects, classification of these objects and their location estimates. The same way as for distributed data fusion, compatibility of information is verified based on data age and spatial distribution. The composed information bundle receives a time-stamp of its own, to represent the time of its creation.

The benefit of data aggregation is twofold: to optimise network resources, by having only one (slightly larger) message to deliver rather than several, and to provide better situational awareness to end users. Having data aggregation capability at the very edge of the network enables users to acquire information from network nodes directly while in-the-field without having to make database queries. This is an important operational advantage for military ISR applications.

Data validity for fusion and aggregation

In order to ensure the correctness of in-network data fusion at different network levels, the data validity must be re-evaluated at each level. Online validation service provided by the ProWare middleware realised as MURP modules (described in section 'Communication architecture and principles of ProWare') requires that the data produced are augmented with additional metadata for ensuring temporal and spatial correctness. According to ProWare concept, the data validity is checked on both sides, first the data producer decides if it is able to provide the data according to the consumer requirements and second the consumer evaluates the data validity when it arrives.[20] A WSN simulation described in Preden et al.[47] shows that ProWare data validation concept considerably helps to reduce the total number of packets exchanged in WSN. In the following some of the more important temporal and spatial validity aspects for current experimental WSN setup are described.

Checking the correctness of data in the temporal domain requires that a sensor reading is augmented with two pieces of time-related metadata: validity interval and the age of the sensor data. The age of data is represented in relative timescale and is incremented by each network node by the time it has spent on processing the data. The validity interval describes an interval when the data are usable and is decided based on the knowledge about the physical phenomena being measured. In case the validity interval expires before the message reaches the consumer, the message is dropped. Using the age of the measurements, computed by sensor platform and incremented by each node on communication path, it is possible for the data consumer (e.g. fusion node) to compute what was the original data acquisition time at the specific sensor and use it for example when evaluating the simultaneity of multiple arrived measurements.

A generic example of distributed sensor communication is presented in Figure 8. Black rectangles on the sensor timelines (t_{S_1}, t_{S_2}, t_{S_3}) represent the duration of signal processing on each sensor node, and arrows indicate the transport times. The packets reach the fusion node at times T_{S_1}, T_{S_2}, T_{S_3} (on timeline t_{FN}). On the fusion node ProWare estimates the total transport and processing times of each packet incoming from S_1 to S_3, denoted by rectangles T_{proc} and $T_{transport}$. The event detection times are then aligned to the estimated time moments $\widehat{T}_{S_1}$, $\widehat{T}_{S_2}$, $\widehat{T}_{S_3}$ and compared against the simultaneity interval $I_{simultaneity}$ (on timeline $\widehat{t}_{FN}$). Figure 8 illustrates that the processing time and packet transport

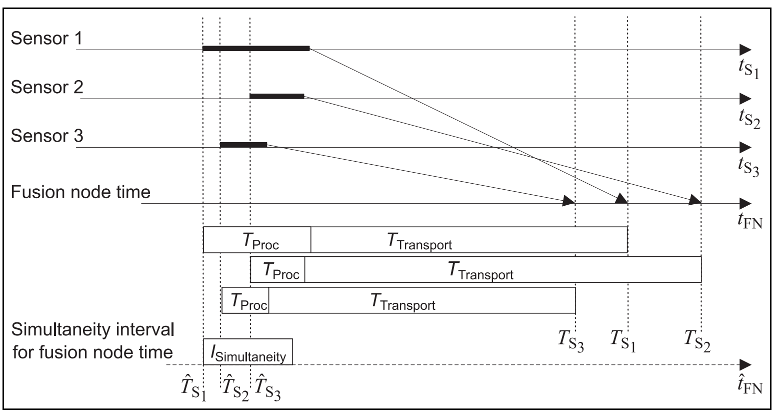


Figure 8. Temporal alignment and the simultaneity interval.

time for each sensor node may be different, which results in the packets arriving out of order and too far apart to be included in the same simultaneity interval. Therefore, without proper temporal validation event, concurrency is not guaranteed to be established. Processing delays, originating from sensor platform non real-time operating systems, clock jitters and drifts, ad hoc WSN transmission scheduling, limited bandwidth and packet collisions within the network – all can unexpectedly disrupt WSN communication. The questions of how precisely the original time moments of measured events from different sensors are estimated, and what is the maximum simultaneity period, required for fusion, are the topics of future research.

In spatial domain, the sensor data are augmented by validity metadata which are the sensor position and computed confidence value for the relative bearing to the noise source. This confidence is used by the fusion process in order to derive the confidence of the outcome of the fusion process. The subscription for the data broadcast by data consumer contains the area information from where the data is needed – this is spatial constraint. The data provider evaluates this spatial constraint against the sensor position and the area in the sensor FOV and in case the spatial constraint is satisfied (together with temporal constraint) the sensor-node will provide the data.

Communication architecture and principles of ProWare

The communication layer that ties together all the heterogeneous sensors provides data exchange and enables distributed fusion and aggregation in WSNs is a multi-functional middleware. There are numerous existing examples of middleware, with different capabilities and properties, a survey paper to some of them was referenced in section 'Related work'. We discuss the principles, hardware and software of the middleware layer developed by our team.[48]

Extracting valid situation information from a WSN relies on correct acquisition and interpretation of sensor data as well as correctly combining and evaluating data collected from different sources. In dynamic, quickly changing environments typical to military operations, only relevant data must be exchanged in a timely fashion and guided by real needs. Central data collection (to a remote database) and distribution comprises a lot of redundancy and is often not flexible enough to provide the necessary timely situation awareness. An alternative approach is one, where service agreements between data users and providers are established at run time, based on actual needs. In this case, communication links are formed locally in ad hoc manner, increasing system robustness and efficiency. The above described procedures ensure the validity of communicated data (its correctness and relevance upon arrival to consumer) over unreliable links with unknown delays.

Considering the described requirements to WSNs used in ISR applications, we describe a solution[48] that facilitates run-time data provider discovery and linkage to consumers, setting constraints to subscribed data, end-to-end transfer timing, tagging of exchanged data with metadata tags and checking data validity. The solution is in the form of a stand-alone communications module (custom design transceiver), which comprises software components, referred to as ProWare,[20] and a hardware platform, referred to as MURP. (MURP has been designed, developed and produced by Thinnect Inc in cooperation with Research Laboratory for Proactive Technologies, Tallinn University of Technology.)

The general principle of how network nodes are connected through MURP transceivers and how communication is organised by ProWare is depicted in Figure 9. Nodes are equipped with the MURP module and all data transmitted and received passes through the module. MURP connects to nodes over a serial interface and enables creating a unified network from devices that may otherwise be different in nature (i.e. with incompatible software and/or hardware). ProWare provides the necessary communication services. These include handling data requests (in the form of subscriptions), establishing service agreements with suitable data providers and facilitating the delivery of produced data to consumers. It absolves the sensing and fusion applications running on network nodes from locating and contracting data providers themselves. Nodes need only to specify the type of data they produce and data they consume (when the need arises) and ProWare is responsible for arranging the communication. A node may be both a consumer and a provider, depending on the situation and on its functionality. In Figure 9, node 1 is a data consumer for node 2 and a data provider for node 3.

ProWare also supports setting constraints (temporal and spatial) to data subscriptions, meaning that data consumers can impose restrictions (e.g. location – from where data are acquired, age – how fresh data must be) to specify what kind of data is acceptable to them. On the producer side data are tagged with necessary metadata tags (time and location of production) and upon arrival to consumers their validity and the satisfaction of subscription constraints are checked. Constraints are not limited to temporal and spatial measures, other norms (e.g. confidence and reliability) may be included.

The networking layer, established by MURP module, automatically forms clusters of well-connected nodes and partnerships between the clusters. All nodes are essentially equal, their roles in the network determined dynamically at run-time and automatically adapted to changing conditions. Advanced mesh

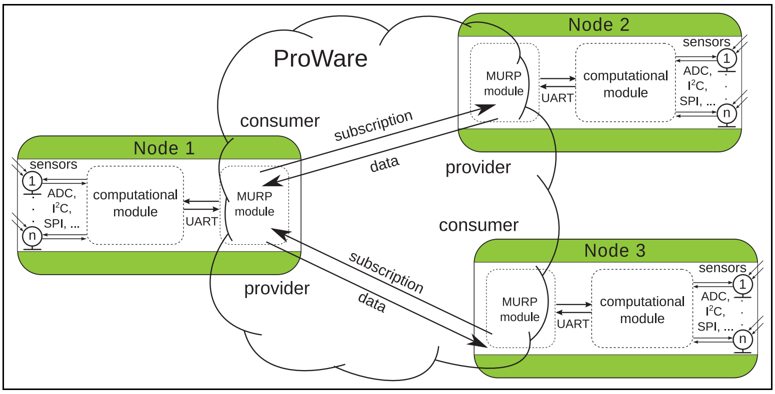


Figure 9. Network nodes equipped with MURP transceivers forming communication links through ProWare.

routing is provided on top of the clusters, making it possible for any individual node to communicate with any other node in the network. Although the clustering scheme achieves excellent reliability and low power consumption, it can be easily overloaded with classical WSN data collection tasks and currently does not implement any network level aggregation capabilities for reducing the load of network-wide data collection (aggregation described in section 'Aggregation' happens on the application level, not the network level). While it is possible to deploy several gateway nodes for data collection, the design is more oriented towards establishing complex data flows inside the network. Data are not methodically collected to one point, instead they are directly sent to users based on their existing needs. The capability of dynamically organising local interactions is especially suitable for fusion tasks, since data fusion in practice tends to take place between nodes that are in close physical proximity and therefore in the same or neighbouring clusters. Only fusion results need to be communicated further in the network. A means of tracking the time that a packet spends in transit from source to destination is provided, allowing for events to be correlated with an accuracy of a couple of milliseconds.

Demonstrations and experiments

We describe two field tests. One that evaluates the feasibility of the entire proposed WSN in a military operation scenario and one that evaluates network communication loads and in-network data fusion efficiency. The former experiment presents no numeric data, rather it evaluates the operation of individual sensors as an ensemble and the ability of the network to answer ISR user needs. The focus of this experiment was to demonstrate, in contrast to typical central data collection, how dynamic sharing of data between network nodes can benefit military operations. The latter experiment evaluates a smaller part of the whole network, namely the acoustic localisation part, in an urban setting and presents numerical data of communication loads involved in the acoustic localisation process (i.e. packets sent between sensor and fusion node and fusion node and end user). It also demonstrates the efficiency of in-network distributed data fusion.

Demonstration of system operation in military setting

A field demonstration for European Defence Agency (EDA) project IN4STARS 2.0 (Information Interoperability and Intelligence Interoperability by Statistics, Agents, Reasoning and Semantics) was performed during the fall of 2015. The broad goal of the IN4STARS 2.0 project is to enhance information exchange and analysis for ISR applications between multiple (and multinational) stakeholders. The field demonstration included a distributed, unattended sensor network with various sensor modalities (acoustic, motion detection, electro-magnetic and optical) enhanced with data validation and fusion capabilities. The purpose of this ground sensor network was to detect the presence of adversary personnel and vehicles, classify the type of the vehicles and track their progress, while at the same time a nearby friendly UAV, equipped with a camera, was deployed to provide visual confirmation of the detected phenomena.

A total of 16 sensor nodes were deployed: 4 microphone arrays implemented on 8-bit Atmel AVR-based platforms, 4 microphone arrays implemented on BBB development boards, 3 proprietary military grade passive infrared (PIR) sensors for personnel detection, 1 proprietary magnetometer sensor, 3 camera sensors, 2 aggregation and fusion nodes and 1 autonomous UAV

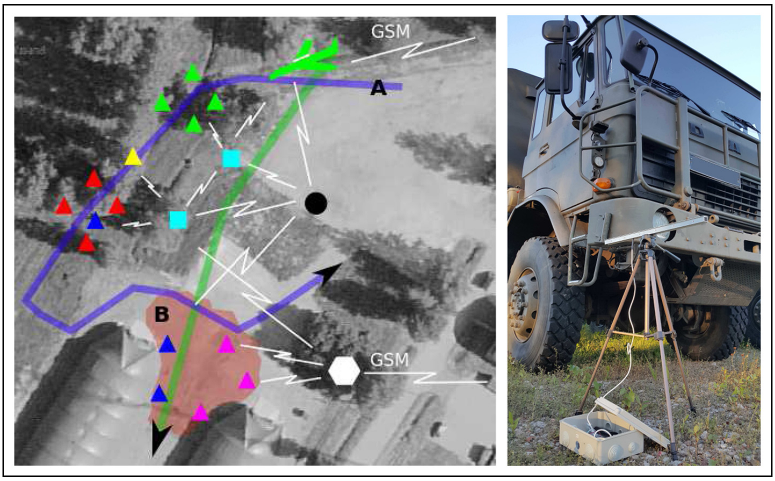


Figure 10. Sensor node placement (left) (green triangles – BBB acoustic sensors; red triangles – 8-bit microcontroller acoustic sensors; dark blue triangles – PIR sensors; pink triangles – camera sensors; yellow triangle – magnetometer; light blue squares – fusion and/or aggregation nodes; black circle – tablet user; white hexagon – gateway node; purple line – route A of military vehicles; pink area – monitored area B; green line – route of UAV) and acoustic sensor node with vehicle used in experiment (right).

with a daylight camera. The field experiment was conducted on the grounds of a military base, with the sensors covering an area of approximately 1.5 Ha. Sensor nodes placement can be seen in Figure 10. Sensor devices need to know their precise locations in order to perform data aggregation and fusion. In the experiment, the nodes were placed manually and global positioning system (GPS) coordinates of the positions were acquired with a GPS receiver and loaded into the nodes at the beginning of operation via the ProWare interface.

Sensor node communication was established using MURP modules, which have an IEEE802.15.4 compliant 2.4 GHz radio and provide mesh networking. The effective communication range was 60–100 m.

The demonstration scenario included different military vehicles passing along route A at different times, while area B was monitored to detect human activity (see Figure 10). Upon detection of activity, the UAV would be deployed to take pictures of route A or area B as required. Four different slow moving military vehicles were used: a light patrol vehicle, a light utility truck, a heavy truck for personnel and an armoured personnel carrier. The speeds of the vehicles, when driving through the sensor network, ranged from 10 to 35 km/h. The distances between the sensor nodes and tracked vehicles varied from 3 to 20 m. All microphone array sensors, one PIR sensor and the magnetometer were placed along route A to detect vehicles, while other PIR and camera sensors monitored area B.

All sensor nodes perform initial signal processing and data analysis. PIR sensors detect motion and nearby camera sensors take pictures according to motion events received from PIR sensors. Acoustic sensors determine the direction to sources of noise and try to classify the source (in this case the four different vehicles). Aggregation and fusion nodes combine the individual direction estimates received from acoustic sensors to distinguish real phenomena (and establish their precise location) from random noise.

The set of situations that the WSN can detect can formally be described using the notation referenced in section 'ISR requirements and situation awareness for military WSN'. Basic situations are implicitly defined for all sensor read-outs, while higher level situations must be defined based on the available basic situations and application needs. The situation $S_{location}$ is an example of a higher level situation, one that is created by fusing basic situations received from sensors, $S_{angle} = \{S_p, S_t, S_a\}$ in this case. The fusion function would be the function that calculates the intersection points of all beams (beams formed based on angle value S_p and sensor location S_a), while considering temporal compatibility S_t of the sensor measurements. Other higher level situations are created similarly, or by the aggregation technique, for example, $S_{vehicle} = \{S_p, S_t, S_a\}$, where $S_p = \{S_{location}, S_{mag}, S_{class}\}$ and S_{mag} and S_{class} are respectively magnetometer sensor read-outs and classification results, or $S_{hostiles} = \{S_p, S_t, S_a\}$, where $S_p = \{S_{pir1}, S_{pir2}, S_{camImage}\}$.

The data produced by the sensor network was accessible in two ways. First, autonomous friendly military units in the vicinity could subscribe to sensor information via rugged military tablets with the specific user interface installed. Second, a remote database server was set up for far-away stakeholders (e.g. analysts form friendly nations). Tablet users access the WSN directly, through an attached MURP device, and/or through a (GSM) gateway, while the database is connected to the WSN through a GSM gateway.

The database is not a necessary component of the system (since local users can access the network directly), in the experiment it was used to collect sensor-data for post-experiment analysis and to supply other subsystems of ISR with input data.

According to the scenario, PIR and camera sensors would detect hostile activity in area B and notify (with pictures of detected events) a remote command centre through the network gateway. A friendly military unit is then sent to investigate the situation. Once it reaches the vicinity of the sensor network, it starts receiving the latest data about detected events directly to its tablet device. While investigating the situation, additional information is received from acoustic and magnetometer sensors that warn the military unit of approaching vehicles along route A. The acoustic sensors determine the location of the vehicles and try to classify them. The early warning enables the friendly military unit to retreat to a safe distance and order an UAV to come and survey the new activity. The UAV can, in principle, communicate with sensor nodes, once it is in communication range, and adjust its mission (e.g. adjust the area to be surveyed) to the latest information. In the experiment, this was not tested. The requirements and challenges of UAV and ground sensor network cooperation have previously been described in Kaugerand et al.[49]

The experiment demonstrated the concept that signal processing, data analysis, distributed aggregation and fusion are done inside the network by sensors or other special nodes and that users access this information directly, when in vicinity, over physical links and through service agreements established automatically at run-time based on existing needs.

Evaluation of in-network data fusion and network bandwidth usage

The experiment evaluates how in-network data fusion can benefit network communication by reducing communication loads and how it can increase the dependability of overall WSN results. Only one type of sensor (microphone sensors nodes) and fusion nodes are used, in order to simplify the experiment and demonstrate the benefits clearly. The task of the WSN is to detect sound emitting objects and estimate their location based on acoustic information collected by microphone sensor nodes. The procedure is described in sections 'AoA of acoustic sound waves' and 'Location estimation'. Generality is not lost with this setup, as all the essential WSN characteristics that have been described so far, including in-sensor signal processing, in-network data fusion, subscription based sensor discovery and tasking and data validity checking are present in this experiment.

The three main claims that the experiments should validate or refute are as follows:

1. Utilising a consistent system of spatial and temporal constraints within the network is necessary for correct distributed data fusion and aggregation;
2. Correct fusion enables to eliminate some of the false-positive results of individual sensors, therefore improving the quality of the end result;
3. In-network fusion and aggregation reduces the number of packets sent to end users (e.g. a database or any other user).

We expect the experiment to show that without any spatial and temporal constraints (or with very loose constraints), the fusion process (in this case object location estimation) will produce a lot of erroneous results (results that do not describe an actual object). This will happen because data from sensors will be accepted and fused even if they are incompatible. We also expect the experiment to show that sensors on their own may detect objects that are actually not there (false positives) due to environmental disruptions such as winds or heavy rain. In these cases, proper fusion processes can eliminate some of the false positives by fusing only spatially and temporally compatible sensor data. Finally, we expect to see lots of sensor data messages (packets) being sent to fusion node, but much fewer fusion result messages (ideally only correct positive results) being sent to the end user. Sending fewer messages from the end of the network to the centre saves node energy and network bandwidth.

For the experiment, eight microphone array sensor nodes (based on BBB) were used to record 30 min of acoustic signals by the side of an urban road with moderate traffic. The same sensors were then set up in laboratory conditions where they, instead of recording signals, now read the previously saved acoustic data and treated it as if it where directly received from their ADC modules. By this way, it was possible to repeatedly play through the same 30 min of situations with different network and data constraint configurations and compare the results.

Network node configuration is depicted in Figure 11. Two fusion nodes A and B where used with node A receiving messages from the four sensors on the left and B receiving messages from four sensors on the

Table 1. Message sending intervals and temporal and spatial constraints for four different experiments.

Expeariment		Configurable parameters			
No	Name	Sensor message sending interval	Fusion message sending interval	Age of sensor data	FOV circle radius
1	Loose constraints	2	3	7	40
2	Extreme constraints	2	3	1	9
3	Optimal	2	3	3	9
4	High sending interval	1	2	3	9

right. The four sensors on the left are referred to as cluster A and the sensors on the right as cluster B. Sensors where placed next to the road in order to detect passing vehicles. A total of 92 vehicles, of which two where buses, two were motorcycles and the rest were cars, passed by the sensors during the 30 min. The speed limit at this stretch of road was 50 km/h. Sensor sampling speed for each microphone was 20 kHz and measurement frame length, used in AoA processing, was 136.5 ms. As a result, approximately seven AoA calculations were done per second. The results were sent to fusion nodes at an interval determined by the data subscription agreement between sensor and fusion nodes. The parameters for different experiment runs can be seen in Table 1. For example, a sensor message sending interval of 2 s means that a sensor node will buffer AoA results for the last 2 s and at send-time only the latest valid result will be sent. An alternative, not used in the experiments, is to send all buffered results at send-time and have the fusion node select which results it wants to use. However, most of the AoA calculations end with a negative result, meaning that no particular object could be detected. Negative results are never forwarded, and if during the message sending interval no positive results are buffered, then nothing is sent to fusion node.

In order to monitor what happens in the network during different experiment runs, all sensor and fusion nodes log their activity, by writing different log messages to their serial port. Several single-board computers (Raspberry Pi 2) collect these messages and time-stamp them upon arrival. The computers keep their own clocks synchronised via the network time protocol (NTP), so that all log records are comparable (note that WSN nodes themselves are not synchronised). Activity and results of network nodes that are currently most interesting are as follows:

1. Sensor node message sending times;
2. Sensor AoA result values in these messages;
3. Sensor message receive times at fusion node;
4. Fusion calculation times and results.

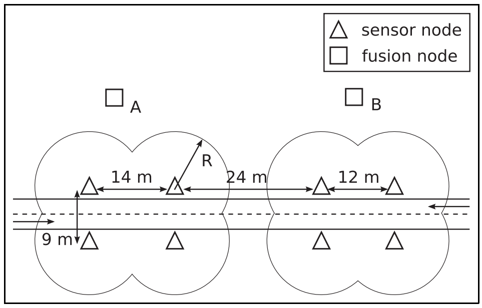


Figure 11. Sensor node placement for vehicle detection.

In order to compare fusion results to actual objects, it is necessary to know when vehicles passed through the WSN. During the acoustic signal recording process, a video-camera also recorded the passing of each vehicle and later this video was analysed to count all vehicles and record their times of occurrence. This was done using the object counting software described in section ‘Automatic object counting’ and the occurrence times were manually re-checked.

A total of four experiments are performed (see Table 1) and their results presented. The experiments differ by altering four select parameters, two that change sensor and fusion nodes message sending intervals and one for temporal and one for spatial constraints. Sensor sending interval, fusion sending interval and age of sensor data are all measured in seconds. For the experiments, the spatial constraint set for data is defined as an area (the FOV) encircling each sensor and the radius (R in Figure 11) of this circle is set in meters.

Formally, these four parameters, are related to each other by mathematical descriptions given in Motus et al.[19] and more thoroughly explained in Rodd and Motus.[50] The formalism defines a mapping between sensor and fusion node timesets $\sigma_{sf} : T(p_s) \times T(p_f) \times val\, p_s \rightarrow proj_{val\, p_s} dom\, p_f$, which conveys sensor values to fusion node domain of definition. Here, sensor and fusion nodes are considered processes p_s and

p_f, respectively, and $T(p_s)$ and $T(p_f)$ represent the execution timesets of these processes. In the experiments, sensor and fusion nodes execute periodically and the timesets are defined by $T(p) = \{t : t_n = t_0 + n \cdot t_a\}$, where $t_0 = 0$, $n \in \mathbb{N}$ and t_a is the message sending interval of the node. The actual time model of the Q-model[50] is more complicated, including among other features channel delays, execution time and start instance indeterminacy, but these are not considered here.

A channel function $K(\sigma_{sf}, t) \subset T(p_s), t \subseteq T(p_f)$ is defined between each sensor node and fusion node, which aligns the sensor node timeset to the fusion node timescale. The channel function is calculated for every fusion execution time and it determines which sensor data can be used at this fusion time instance, that is, the channel function must satisfy the relation $K(\sigma_{sf}, t) = [\mu, \nu]$, where $[\mu, \nu]$ is a time interval and μ defines the oldest and ν the latest allowable element in channel. In the experiments, $\nu = 0$ and μ corresponds to the age of data (see Table 1). Data elements exchanged through channels are the situation 3-tuples referenced in section 'ISR requirements and situation awareness for military WSN'. Sensor read-outs represent subscription parameters S_p, measurement time is a temporal parameter from S_t, and sensor location is a parameter in S_a. A modified channel function in Motus et al.[19] explains how spatial parameters are handled, for simplicity we have only used the temporal channel function here.

Experiments number 1 and 2 are performed with respectively very loose and very strict temporal and spatial constraints on sensor data, while message sending intervals are left unchanged. Altering data constraints should have considerable effect on fusion results and the successful positioning of vehicles. Experiments number 3 and 4 are performed with moderate and high message sending intervals for both sensor and fusion nodes. Data constraints in these two cases are left to what we have previously found are optimal values for good vehicle positioning. High message sending intervals should cause more packet collisions and packet loss, which will disrupt overall operation and the quality of results.

The results of the fusion process and overall efficiency of the sensor network to detect vehicles are presented in Table 3. Statistics of sent messages can be viewed in Table 2. Either cluster of sensor nodes has been reviewed separately because all sensor nodes are dedicated to their appropriate fusion nodes and no messages are sent between clusters (however, the single communication channel still has to be shared by both clusters). Generally, cluster A performed better than cluster B in all experiments by detecting vehicles more precisely and by giving less false positives (i.e. successful fusion results, when there actually is no vehicle near the sensor network). We are unsure of the exact reasons and plan on investigating the matter in the future. False positives and false negatives were reported by both clusters and are unfortunately inevitable unless additional sensors (e.g. a magnetometer) are added to the given

Table 2. Messages sent and received.

Exp. number	Sensor node sent messages			Fusion node	
	Total	Vehicle	No vehicle	Received	Sent
Cluster A					
1	1097	681	416	1084	224
2	1097	683	414	1091	10
3	1116	691	425	1089	122
4	2180	1341	839	2127	196
Cluster B					
1	1219	577	642	1208	264
2	1226	590	636	1217	10
3	1220	579	641	1219	123
4	2433	1163	1270	2406	210

Table 3. Fusion results for both fusion nodes.

Exp. number	Total	Unsuccessful fusion		Successful fusion		
		Temporal mismatch	Spatial mismatch	False positive	Vehicle	Total
Fusion node A						
1	342	91	27	55	169	224
2	180	170	30	2	8	10
3	285	113	50	36	86	122
4	393	141	56	60	136	196
Fusion node B						
1	418	98	56	100	164	264
2	303	283	10	7	3	10
3	368	154	91	54	69	123
4	501	162	129	92	118	210

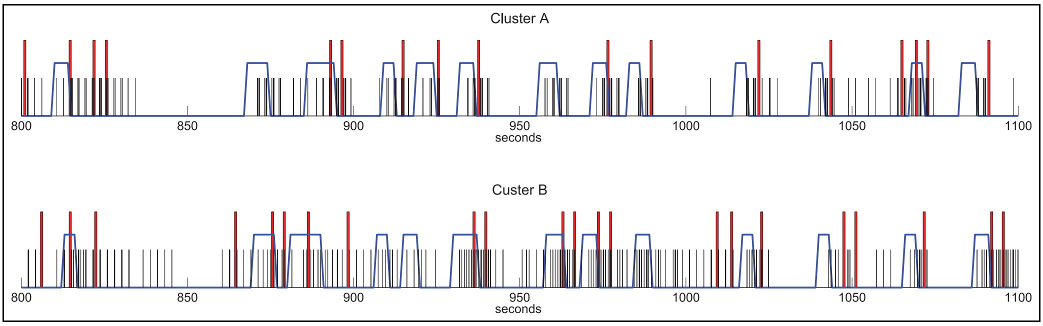


Figure 12. Sensor and fusion computation results versus vehicles passing (tall red bars – fusion node results; blue trapezoids – vehicle passing; low narrow black stripes – sensor node results).

network. This is understandable, as microphone array sensors are intended to locate sound-emitting objects, but not distinguish between vehicles and other environmental noise. During earlier testing, a case was documented, where the noise of a passing aeroplane fooled the sensor network to give consecutive false positives, the same can be caused by winds, heavy rain and so on.

Figure 12 depicts events happening during an experiment.

Data from experiment number 3 is used to depict the instances of successful fusion calculations (tall red bars), the instances when sensor nodes send messages (narrow black stripes) and time intervals when a vehicle was near sensors (blue trapezoids). Vehicle occurrence times are acquired from recorded video and plotted with 1 s granularity. Trapezoids with shorter width typically represent single vehicles and trapezoids with longer width either slow moving vehicles or several vehicles passing in close succession. The vehicle events depicted in either cluster are not precisely aligned, the shifts are caused by different directions and speeds of vehicles. The excerpt reveals that occasionally false negatives happen, that is, both clusters fail to detect a vehicle (e.g. after 850 s for cluster A and after 900 s for cluster B), but between the two no vehicle is left undetected. It also shows false positive fusion results (e.g. at around 825 s in cluster A and just before 900 s in cluster B). The fact that fusion instances occur a few seconds after vehicles is because sensor and fusion node execute periodically at discrete intervals (see Table 1).

One of the three main goals of experiments was to assess the effect of using different spatial and temporal data constraints on the fused sensor data. The results show that choosing appropriate constraints to suite the task at hand has major impact on the efficiency of the system. Comparing experiments number 1 and 2, which differed only by the constraints set for fusion input data (see Table 1), there is a more than 20 time difference between the number of successful fusions. Table 3 shows the number of unsuccessful fusions (fusions that were disregarded) because of either temporal or spatial mismatch of sensor data. Temporal mismatch of data disregards most of these unsuccessful fusions because it is the first constraint that is checked, data that pass the check are only then submitted to spatial validity checking and additional cropping. Both experiments 1 and 2 represent extreme cases of data constraint usage, showing that with very loose constraints, there will be more successful fusions including more false positives and that with very strict constraints there will be fewer fusions and less real objects detected. An optimal set of data constraints (determined empirically during the course of experiments) was used in experiment 3. In this experiment, the number of successful fusions is reasonable considering the total number of vehicles (92) and the number of false positives is in-between those of experiments 1 and 2. What is important is that when combining the results from both clusters only two vehicles were able to drive by undetected in experiment 3. Unless sensor technology itself is improved, this is one of the ways to deal with false negatives, that is, by adding sensors (with different modality) and considering the results of more sensors.

The second goal of experiments was to see how many AoA results individual sensors produce and how many of these lead to successful fusions. Table 2 presents the total number of messages sent to fusion node by all four sensors of a cluster. Each message contains a new AoA value, so the number of messages reflects the total number of valid AoA results produced by sensor nodes. Comparing message send times with the all the times when a vehicle was in the FOV of sensors, reveals that about 40% of messages for cluster A and roughly half of messages for cluster B occur when no vehicle is present (see Table 2). It shows that the environment is noisy and a lot of sound sources are detected, that are not of interest. Fusion helps improve the end result and decrease false positives by applying constraint checking and eliminating for example isolated AoA instances of individual sensors and concurrently produced random AoA instances of multiple sensors. For experiment number 3, the ratio of false positives and correctly detected vehicles improves to 30%/70% for cluster A and 44%/56% for cluster B (consider Table 3 successful fusions). The fact that sensor nodes send AoA messages when there are no vehicles in their FOV and that fusion nodes mostly don't fuse these results is also evident from the experiment timeline depicted in Figure 12.

The third and final goal was to determine the difference of bandwidth usage between forwarding all sensor messages and forwarding only fusion result messages to the higher ISR level (e.g. an operating military unit and a remote database). In all four experiments, the amount of fusion messages sent was approximately 10 times less than the number of sensor messages sent (see Table 2). However, what is more important than the total amount of messages sent is when they are sent. It is possible to see in Figure 12 that near vehicle occurrence times the number of sensor messages increases (sections of narrow black stripes get denser). This is because all sensors detect the vehicle and want to use (the shared) communication channel at the same time. In our small WSN of eight sensors, this did not cause a problem, not even for experiment number 4, where message sending intervals where changed, such that sensor nodes sent AoA results (when they had any) at an interval of 1 s. While we were able to show that by utilising in-sensor signal processing and in-network fusion it is possible to limit the total number of messages generated by a WSN, we were not able to demonstrate significant packet collisions and congestion of our network at all. An experiment with either a larger number of nodes or shorter message sending intervals is probably needed to demonstrate this.

In conclusion, the four experiments demonstrated the benefit of using in-network distributed data fusion to increase the dependability of WSN results and to decrease the amount of data forwarded to end users. It was also shown that data validity checking (at least against temporal and spatial compatibility) is essential for correct data fusion. We expected to see congestion at network usage peaks, but did not succeed in creating situations where network becomes congested by an overflow of sensor messages. Since the minimum message sending interval of 1 s (see experiment 4) and cluster size of approximately 10 nodes is sufficient and reasonable for our monitoring applications, we do not attempt to find the communication breaking point of the WSN at this moment. In general, the WSN was able to fulfil its task of detecting passing vehicles, although a considerable amount of false positive results were also produced. This shortcoming can further be improved by adding additional sensors to the WSN.

Conclusion

In this article, we have described contemporary situation awareness needs for ISR applications and presented the design and implementation of one possible approach, in the form of a real-life deployed WSN that satisfies those needs. ISR users operate at different levels of military hierarchy and have very different data and information needs depending on the tactical or strategical context. Some need near-real-time sensor information from their vicinity for tactical operation, others need already filtered and fused and/or aggregated information for specific purposes and yet others need matured high level information for strategic decision making and long-term analysis. Such a variety of network users require the creation of information understandable by humans at all hierarchical levels of data production and fusion and in each with their own time and location requirements. In-sensor signal processing (Mist Computing) avoids overwhelming the network with raw data and is the first step to providing data ready to be presented (D2D readiness) for human users.

Our WSN design utilises a middleware-based solution, which supports the service-oriented approach, dynamic data provider discovery and run-time ad hoc (re)formation of network links. It also supports validity checking of communicated data based on temporal and spatial constraints established in service agreements between data providers and consumers. The consistent utilisation of spatial and temporal constraints for ensuring data validity and mutual conformity is essential for correct data fusion and aggregation. Enabling correct in-network fusion in turn improves the quality of information provided by the overall network, helps to eliminate false positives and also reduces considerably bandwidth requirements. Two experiments were described both demonstrating the benefits of pushing computation to the edge of IoT. The first experiment was conducted to evaluate the feasibility of the entire proposed WSN in a military operation scenario and the purpose of the second experiment was to evaluate network communication loads and in-network data fusion efficiency. The WSN experiments demonstrated how data collected by the WSN are dynamically used at the different levels of military operation as dictated by D2D paradigm.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The work presented in this paper was partly supported by the European Defence Agency project IN4STARS 2.0 and the Estonian IT Academy program.

References

1. Preden J, Kaugerand J, Suurjaak E, et al. Data to decision: pushing situational information needs to the edge of the network. In: *Proceedings of the IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision*, Orlando, FL, 9–12 March 2015, pp.158–164. New York: IEEE.
2. Srisooksai T, Keamarungsi K, Lamsrichan P, et al. Practical data compression in wireless sensor networks: a survey. *J Netw Comput Appl* 2012; 35(1): 37–59.
3. Mayk I, Regli W, Nguyen D, et al. Net-centric information and knowledge management and dissemination for data-to-decision C2 applications using intelligent agents and service-oriented architectures. In: *Proceedings of the military communications conference (MILCOM 2011)*, Baltimore, MD, 7–10 November 2011, pp.1568–1573. New York: IEEE.
4. Preden JS, Tammemae K, Jantsch A, et al. The benefits of self-awareness and attention in fog and mist computing. *Computer* 2015; 48(7): 37–45.
5. Corke P, Wark T, Jurdak R, et al. Environmental wireless sensor networks. *P IEEE* 2010; 98(11): 1903–1917.
6. Dehwah AH, Mousa M and Claudel CG. Lessons learned on solar powered wireless sensor network deployments in urban, desert environments. *Ad Hoc Netw* 2015; 28: 52–67.
7. Shen Y, Yang P, Zhang P, et al. Development of a multi-type wireless sensor network for the large-scale structure of the national stadium in China. *Int J Distrib Sens N* 2013; 2013: 16pp.
8. Jo H, Sim SH, Mechitov KA, et al. Hybrid wireless smart sensor network for full-scale structural health monitoring of a cable-stayed bridge. In: *Proceedings of the SPIE7981 sensors and smart structures technologies for civil, mechanical, and aerospace systems*, San Diego, California, USA, 06 March 2011, vol. 7981, p. 798105.
9. Yoon I, Noh DK, Lee D, et al. Reliable wildfire monitoring with sparsely deployed wireless sensor networks. In: *Proceedings of the 26th international conference on advanced information networking and applications (AINA)*, Fukuoka, Japan, 26–29 March 2012, pp.460–466. New York: IEEE.
10. Ramesh MV. Design, development, and deployment of a wireless sensor network for detection of landslides. *Ad Hoc Netw* 2014; 13: 2–18.
11. Liu G, Tan R, Zhou R, et al. Volcanic earthquake timing using wireless sensor networks. In: *Proceedings of the 12th international conference on information processing in sensor networks*, Philadelphia, PA, 8–11 April 2013, pp.91–102. New York: ACM.
12. Salvadori F, De Campos M, Sausen PS, et al. Monitoring in industrial systems using wireless sensor network with dynamic power management. *IEEE T Instrum Meas* 2009; 58(9): 3104–3111.
13. Lim A, Yang Q, Casey K, et al. Real-time target tracking with CPA algorithm in wireless sensor networks. In: *Proceedings of the 5th annual IEEE communications society conference on sensor, mesh and ad hoc communications and networks*, San Francisco, 16–20 June 2008, pp.305–313. New York: IEEE.
14. Wittenburg G, Dziengel N, Wartenburger C, et al. A system for distributed event detection in wireless sensor networks. In: *Proceedings of the 9th ACM/IEEE international conference on information processing in sensor networks*, Stockholm, 12–16 April 2010, pp.94–104. New York: ACM.
15. Gao T, Massey T, Selavo L, et al. The advanced health and disaster aid network: a light-weight wireless medical system for triage. *IEEE T Biomed Circuits Syst* 2007; 1(3): 203–216.
16. Egner AI, Luu D, den Hartog J, et al. An authorization service for collaborative situation awareness. In: *Proceedings of the sixth ACM conference on data and application security and privacy*, New Orleans, LA, 9–11 March 2016, pp.136–138. New York: ACM.
17. Turkmen F, den Hartog J, Ranise S, et al. *Analysis of XACML policies with SMT*. Berlin; Heidelberg: Springer, 2015, pp.115–134.
18. Motus L, Meriste M and Preden J. Towards middleware based situation awareness. In: *Proceedings of the 2009 IEEE military communications conference*, Boston, MA, 18–21 October 2009, pp.1–7. New York: IEEE.
19. Motus L, Meriste M, Preden J, et al. Self-aware architecture to support partial control of emergent behavior. In: *Proceedings of the 7th international conference on system of systems engineering (SoSE)*, Genova, 16–19 July 2012, pp.422–427. New York: IEEE.
20. Preden J, Llinas J, Rogova G, et al. On-line data validation in distributed data fusion. In: *Proceedings of the SPIE8742 ground/air multisensor interoperability, integration, and networking for persistent ISR IV*, Baltimore, MD, 22 May 2013. Bellingham, WA: SPIE.
21. Winkler M, Tuchs KD, Hughes K, et al. Theoretical and practical aspects of military wireless sensor networks. *J Telecommun Inf Technol* 2008; 2: 37–45.
22. Kopetz H. *Real-time systems: design principles for distributed embedded applications*. Berlin: Springer, 2011.
23. Zhao G. Wireless sensor networks for industrial process monitoring and control: a survey. *Netw Protocol Algorithm* 2011; 3(1): 46–63.
24. Peng AS, Moen DM, Spinks JA, et al. Reliable data aggregation and dissemination framework in tactical network architecture. In *Proceedings of the 2010 military communications conference*, San Jose, CA, 31 October–3 November 2010, pp.569–574. New York: IEEE.
25. Pham T, Cirincione GH, Verma D, et al. Intelligence, surveillance, and reconnaissance fusion for coalition operations. In: *Proceedings of the 11th international conference on information fusion*, Cologne, 30 June–3 July 2008, pp.1–8. New York: IEEE.
26. McMahon A and Farrell S. Delay- and disruption-tolerant networking. *IEEE Internet Comput* 2009; 13(6): 82–87.

27. Mohamed N and Al-Jaroodi J. A survey on service-oriented middleware for wireless sensor networks. *Serv Oriented Comput Appl* 2011; 5(2): 71–85.
28. Damarla T and Kirkpatrick Alberts WC. Helicopter tracking using acoustic arrays. In: *Proceedings of the 17th international conference on information fusion (FUSION)*, Salamanca, 7–10 July 2014, pp.1–7. New York: IEEE.
29. Maddumabandara A, Leung H and Liu M. Experimental evaluation of indoor localization using wireless sensor networks. *IEEE Sens J* 2015; 15(9): 5228–5237.
30. Astapov S, Ehala J and Preden JS. Collective acoustic localization in a network of dual channel low power devices. In: *Proceedings of the 21st international conference on mixed design of integrated circuits systems (MIXDES)*, Lublin, 19–21 June 2014, pp.430–435. New York: IEEE.
31. Czúni L and Varga PZ. Lightweight acoustic detection of logging in wireless sensor networks. In: *Proceedings of the international conference on digital information, networking, and wireless communications (DINWC)*, 24–26 June 2014, pp.120–125. Hong Kong: Society of Digital Information and Wireless Communication.
32. Boulmaiz A, Messadeg D, Doghmane N, et al. Robust acoustic bird recognition for habitat monitoring with wireless sensor networks. *Int J Speech Tech* 2016; 19(3): 631–645.
33. Salomons EL and Havinga PJ. A survey on the feasibility of sound classification on wireless sensor nodes. *Sensors* 2015; 15(4): 7462–7498.
34. Rajagopalan R and Varshney PK. Data-aggregation techniques in sensor networks: a survey. *IEEE Commun Surveys Tutorials* 2006; 8(4): 48–63.
35. Bahrepour M, Meratnia N and Havinga P. Sensor fusion-based event detection in wireless sensor networks. In: *Proceedings of the 6th annual international conference on mobile and ubiquitous systems: computing, networking and services, MobiQuitous*, Toronto, ON, Canada, 13–16 July 2009, pp.1–8. New York: IEEE.
36. Lai S, Cao J and Fan X. TED: efficient type-based composite event detection for wireless sensor network. In: *Proceedings of the 2011 international conference on distributed computing in sensor systems and workshops (DCOSS)*, Barcelona, 27–29 June 2011, pp.1–8. New York: IEEE.
37. Zivkovic Z. Improved adaptive Gaussian mixture model for background subtraction. In: *Proceedings of the 17th international conference on pattern recognition*, Cambridge, 26 August 2004, pp.28–31. Washington, DC: IEEE Computer Society.
38. Suzuki S and Abe K. Topological structural analysis of digitized binary images by border following. *Comput Vis Graph Image Process* 1985; 30(1): 32–46.
39. Sklansky J. Finding the convex hull of a simple polygon. *Pattern Recogn Lett* 1982; 1(2): 79–83.
40. Rosten E and Drummond T. Fusing points and lines for high performance tracking. In: *Proceedings of the 10th international conference on computer vision*, Beijing, China, 17–20 October 2005, pp.1508–1515. New York: IEEE.
41. Bouguet JY. *Pyramidal implementation of the lucas kanade feature tracker description of the algorithm*. New York, USA: Microprocessor Research Labs, Intel Corporation, 2000.
42. Lucas BD and Kanade T. An iterative image registration technique with an application to stereo vision. In: *Proceedings of the 7th international joint conference on artificial intelligence*. Vancouver, BC, Canada, 24–28 August 1981, pp.674–679. San Francisco, CA: Morgan Kaufmann Publishers.
43. Lucas BD. *Generalized image matching by the method of differences*. PhD Thesis, Carnegie Mellon University, Pittsburgh, PA, 1985.
44. Riid A and Preden JS. A combined method for error and complexity reduction in fuzzy rule-based classification. In: *Proceedings of the 2015 IEEE international conference on fuzzy systems*, Istanbul, 2–5 August 2015, pp.1–6. New York: IEEE.
45. Astapov S and Riid A. A multistage procedure of mobile vehicle acoustic identification for single-sensor embedded device. *Int J Electron Telecommun* 2013; 59(2): 151–160.
46. Astapov S, Riid A, Preden JS, et al. Industrial process monitoring by multi-channel acoustic signal analysis. In: *Proceedings of the 2014 14th biennial baltic electronic conference (BEC)*, Tallinn, 6–8 October 2014, pp.209–212. New York: IEEE.
47. Preden JS, Motus L, Llinas J, et al. Improvised explosive devices in asymmetric conflicts: multisource data fusion for providing situational information. In: Gorod A, White BE, Ireland V, et al. (eds) *Case studies in system of systems, enterprise systems, and complex systems engineering*. London: Taylor & Francis, 2014, pp.407–443.
48. Preden JS, Motus L, Pahtma R, et al. Data exchange for shared situation awareness. In: *Proceedings of the 2012 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision support (CogSIMA)*, New Orleans, LA, 6–8 March 2012, pp.198–201. New York: IEEE.
49. Kaugerand J, Preden J, Suurjaak E, et al. A system of systems solution for perimeter control: combining unmanned aerial system with unattended ground sensor network. In: *Proceedings of the 2015 9th annual IEEE international systems conference (SysCon)*, Vancouver, BC, Canada, 13–16 April 2015, pp.317–323. New York: IEEE.
50. Rodd MG and Motus L. *Timing analysis of real-time software*. Amsterdam: Elsevier, 1994.

Appendix 3

III

J. Kaugerand, J.-S. Preden, E. Suurjaak, S. Astopov, L. Motus, and R. Pahtma. A system of systems solution for perimeter control: Combining unmanned aerial system with unattended ground sensor network. In *Systems Conference (SysCon), 2015 9th Annual IEEE International*, pages 317–323. IEEE, 2015

A System of Systems Solution for Perimeter Control: Combining Unmanned Aerial System with Unattended Ground Sensor Network

Jaanus Kaugerand, Jürgo Preden, Erki Suurjaak, Sergei Astopov, Leo Motus, Raido Pahtmaa,
Research Laboratory for Proactive Technologies
Tallinn University of Technology
Tallinn, Estonia
jaanus.kaugerand@ttu.ee

Abstract— **The paper describes a novel and practical System of Systems (SoS) approach for perimeter control in ISR (Intelligence, Surveillance and Reconnaissance) applications. The SoS combines an Unmanned autonomous Aerial System (UAS) with an Unattended Ground Sensor (UGS) SoS in order to provide enhanced situation awareness to the users. A simple mini Unmanned Aerial Vehicle (UAV) with an autopilot capable of waypoint navigation is equipped with additional hardware to become a UAS and is integrated with an existing UGS network, forming a System of Systems of heterogeneous systems. The systems in the resulting SoS are autonomous and they offer the generated information via a subscription based service architecture. The SoS described in the paper offers its detection and identification capabilities as services to external entities. The UGS nodes autonomously detect objects and phenomena of interest (based on information requests received from external entities) and by building up collective situation awareness they are able to classify the detected objects, involving the UAS in image acquisition if an object of interest has been classified. The data collected by the SoS is combined and delivered to the external entity that made the information request. The paper describes the creation of the UAS, its integration with an existing SoS and the evaluation of the performance of the resulting SoS.**

Keywords— System of Systems, UAS, UGS, serivice oriented architecture, subscription based interactions.

I. INTRODUCTION

In tactical operations control over large perimeters is a necessity to maintain good situational awareness of the area. This proves difficult and resource intensive if visual confirmation of detected objects is required. This problem could be solved with a considerable amount of personnel or by deploying advanced sensors equipped with cameras. Both of these alternatives are resource intensive and also have many drawbacks (such as response delays, limited field of view). We propose a System of Systems solution where the detection and identification of events is conducted using cost effective ad-hoc UGS network on the ground and complementing the collected data with visual data is provided by a single or multiple UAS platforms. The UGS nodes detect and track the detected objects and once an object of interest has been classified, a request is made for the imaging service from a UAS together with the estimated coordinates of the classified object. If the UAS is able to obtain visual data from the indicated area, the data is delivered to the ground system. The information produced from the data collected by the UGS network and the visual data provided by the UAS is delivered to the human user(s) who originally requested the information to enhance their situation awareness. Together the network of UGS and UAS form a System of Systems (SoS), the concept depicted on Figure 1, where the low fidelity data collected by the UGS is complemented with high fidelity imaging data from the UAS, resulting in more complete situational information.

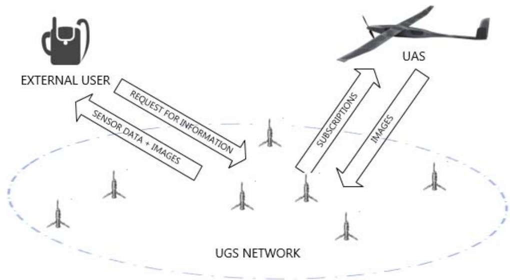


Fig. 1. SoS overview

The different collaboration scenarios between unmanned aerial vehicles and wireless sensor networks (WSN), which UGS networks essentially are, described in literature involve mainly data muling (or data ferrying) from WSN nodes as described in [1], using UAV as a data relay as described in [2] or in [3]. There are fewer examples for guiding or controlling the UAV by WSN. Authors in [1] carried out simulations where the UAV uses positions transmitted from nodes to dynamically re-plan its flight trajectory to aid data muling, however not tests with physical systems were performed in the context of that study. Another dynamical path re-planning scenario is also described in [3] where the UAV creates dynamically a new trajectory analyzing radio performance to the ground nodes in order to collect the data and then proceed to the next node. An example of a WSN guiding a UAV is also outlined in [4], describing two algorithms that can be applied for the task. In first the UAV equipped with GPS module assists the localization of the ground sensors. In the second algorithm the WSN controls the navigation of an UAV using directional radio broadcasts to localize both path waypoints stored in WSN and UAV. Another example of collaboration between unmanned vehicle and WSN is described in [5] where the WSN is used to extend autonomous vehicle sensors range and then the unmanned vehicle can choose an optimal trajectory from a roadmap.

The works referenced above do not describe the unmanned vehicle as an autonomous component of a System of Systems nor is the collaboration dynamic, established in the form of subscription based service calls between the provider and the consumer of data. The current paper considers both UAS and UGS nodes as a part of a SoS, where autonomy of individual systems is crucial for obtaining the desired SoS behaviour and therefore autonomy is an important factor in individual system design.

In order to achieve autonomous behaviour, the autopilot-equipped UAV, described in the current paper, is supplemented with another embedded system for handling high level control of the UAS. We call this embedded system the Pilot Control Module (PCM). Similar modular architecture has also been developed in [6] where UAS is equipped with a high level planner and a low level autopilot as separate modules. The PCM handles both internal communication with autopilot as well as external communication with other systems by receiving subscriptions for image data from UGS and providing the acquired data. The PCM must maintain an adequate level of situation awareness to evaluate the ability of the UAS to provide the requested services as the service requests arrive. When the UAS receives a subscription for image data, the PCM evaluates the UAS situation and plans a mission to acquire images from target area if this can be facilitated. The PCM also relays control communication between autopilot and Ground Control Station (manned by a human), which is used during testing for legal and safety reasons and also in order to monitor the progress of the tests. The implemented UAS prototype was developed considering the requirements of the realistic tactical scenario within the context of the ongoing European Defense Agency project IN4STARS in the Research Laboratory for Proactive Technologies (ProLab) at Tallinn University of Technology.

II. SEVICE BASED SYSTEM OF SYSTEMS ARCHITECTURE

Modern solutions are expected to be able to operate in dynamic, open and unpredictable environments and also in the context of a changing system configuration. In order to efficiently cope with these aspects, the System of Systems (SoS) paradigm has been developed during the lasts decades. In a SoS the systems making up the SoS interact autonomously with each other in order to collaborate and achieve a the higher level SoS goal, a goal that would not be achievable by components alone. The interactions form a crucial part of a SoS configuration as the higher level functionality can be only achieved via a collaboration between the systems. We have chosen to apply a subscription (or service) based data exchange model [7] for building the SoS described in current paper. The interactions between SoS components are mediated using a service oriented proactive middleware (ProWare, developed at the Research Laboratory for Proactive Technologies [8]). ProWare offers the services of provider discovery, on-line data validation and service contract agreements between data providers and consumers. Such an architecture facilitates predictable operation also in a changing SoS configuration.

Unlike a system with a fixed structure, where the functionality of the components and their interaction patterns are well controlled and predictable, in a dynamic SoS the interactions are not fixed as the system configuration itself is not fixed. In the context of nondeterministic interactions between systems the validity of the data that is exchanged is crucial for ensuring correctness of the outputs of algorithms using the data as an input. The use of ProWare enables to set validity constraints for data that is requested from other systems and check the validity of the data in the context of the constraints on-line, while the data is being exchanged. The data consumers do not subscribe for service/data from specific producer, but to service/data constrained by type, time and location. The producer (e.g., the UAS) decides on its own if it is able to provide data that satisfies the resources and constraints. On Figure 2 the concept of applying ProWare in a distributed fusion scenario is depicted. Every system that is part of the SoS has a ProWare component in it, which is responsible for interactions between the systems. The ProWare components makes requests for data (in the form of subscriptions) to other systems and delivers data generated by the local system to other systems.

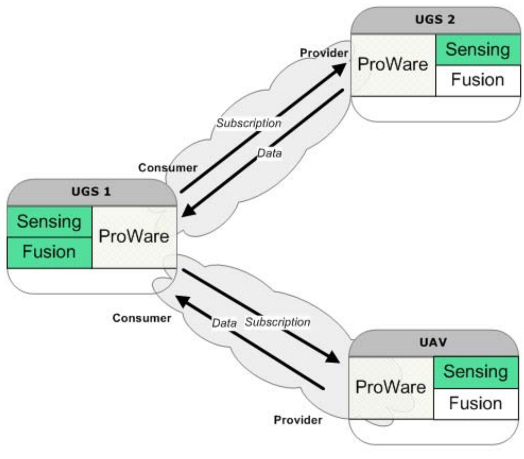


Fig. 2. ProWare in data exchange scenario

This paper focuses on the autonomy aspect in designing the UAS needed for collaboration with UGS network. The objective is to create a SoS, which is assembled from autonomous systems, eliminating the need for long term planning in deployment and operation and also for central coordination. As the UAS is operating in an open and dynamic environment it is not just enough to automatically execute a pre-programmed sequence of steps but the UAS must autonomously be able to decide between choices presented by the perceived situations and adapt to changing environment in order to achieve optimal efficient results.

III. THE UGS NETWORK

The Unattended Ground Sensors (UGS) network for the SoS is a tactical WSN, which is tasked with detecting anomalous or illegal movement in the monitored area and to classify the types of the detected objects. The UGS use various sensor modalities, including PIR, acoustic, seismic and magnetic field, being also able to perform classification in a collaborative manner, performing fusion of collected sensor data within the network. We assume that the sensors are aware of their locations and they exchange information in a form of situation parameters [7] in order to develop and maintain a local situational awareness picture. The use of ProWare for distributed detection is described in more detail in our previous work in [9]. The UGS that will be used for current project, being developed by ProLab in cooperation with an Estonian company Defendec are depicted on Figure 3. The design and tests of UGS are more details described in [10].

Fig. 3. Defendec Smartdec device on the right, experimental magnetic field sensor on the bottom left and experimental sensors in the forest in the left top

When an object of interest is detected and classified, the UGS transmits a subscription for visual information to the UAS. Once the image from the area has been received by the UGS, it is combined with the classification results obtained by the UGS network and communicated to the information consumers who had requested information from the area. In case the UGS receives a negative response from a UAS (e.g. UAS is not able to reach the area in time) or does not receive any response in a certain timeframe from UAS at all, the response is sent to the original subscriber without imaging information.

IV. UAS PLATFORM

The Unmanned Aerial System (UAS) is designed and built using pre-existing systems and components. The main components of UAS are depicted in Figure 4. The main components of the UAS are an autopilot, a set of sensors for flight dynamics, embedded system (Pilot Control Module) for high level control, a communication system for communicating with the other SoS components and an imaging sensor.

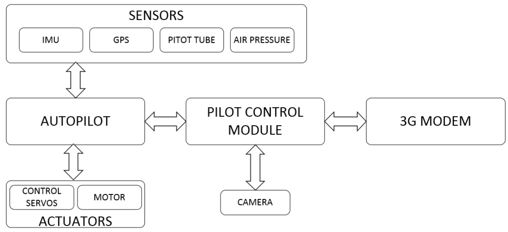


Fig. 4. UAS components

A. UAV airframe

UAV chosen for this project is a small tactical fixed wing aircraft. It weighs around 2 kg, has a wingspan of 1.5 m and its average flight time is 40 minutes. The top speed of the UAV is around 100 km/h. The UAV prototype used for outdoor experiments can be seen on Figure 5.

Fig. 5. UAV used in live tests

B. Autopilot

For control and stabilization of the airframe and for waypoint following the UAS is equipped with an autopilot provided by an Estonian SME "Threod Systems", see Figure 6. It consists of two parts, the autopilot itself and the sensor board, which is mounted on top of the autopilot. The sensors include MEMS accelerometers, gyroscopes and air pressure sensor[s] for altitude. The pitot tube used for air speed measurement is mounted in a wing and connected to autopilot via A/D interface. The GPS module (Ublox NEO-6M GPS module with update rate of 5Hz) is mounted on top of the aircraft, the interface to the autopilot being a serial connection. The sensors board also contains a Secure Digital (SD) memory card reader.

The autopilot is designed to use the Micro Air Vehicle Link (MAVLink) communication protocol to communicate with the Ground Control Station. While in the standard UAV configuration the commands for the autopilot are provided by

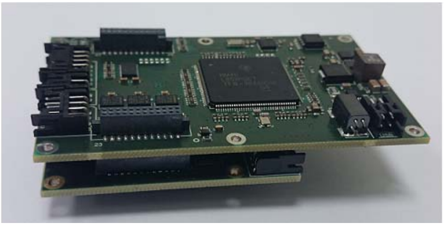


Fig. 6. UAV autopilot

the ground station, in the UAS configuration used in the current project, the commands for the autopilot are provided by the PCM. For reasons of safety the PCM also relays autopilot telemetry to real Ground Control Station so that UAV can be monitored during the outdoor testing.

C. *Embedded system for high level control (PCM)*

After the evaluation of the available COTS embedded computing systems, the system chosen for the PCM was Raspberry Pi (RPI) model B. The main reason for the choice was its good availability and good software support that make RPI very good platform for prototyping. In order to interface the RPI with the autopilot, the serial interface was used. The imaging sensor is also directly interfaced to the PCM.

D. *UAS imaging sensor*

The UAS is equipped with a camera that operates in the visible light frequency. Initially a GoPro or a similar camera was considered, but after initial tests it was concluded that the quality of the RPI standard camera (OmniVision OV5647 image sensor) is sufficiently high for testing purposes, enabling validation of the UAS operation in the desired manner. The planned flying height of such tactical mini UAS is typically less than 120m, at which distance the images provided by the RPI camera are detailed enough.

E. *UAS – UGS communication*

As transfer of images from UAS to UGS requires higher data rates that can typically provided by sub-gigahertz modems, a 3G connection was used for this data link. To enable easy integration of the components a client-server model was utilized for this communication, adding a separate server that facilitated communication between SoS components. The intended communication architecture for such an SoS should be based on a dynamic ad-hoc local wireless network principles and not rely on the internet and client server-model, but this will be implemented in the next phase of the project. The replacement of the communication link (direct link versus a client-server based model) does not affect the underlying high-level design of the SoS in question, as for the components of SoS the communication model is transparent. As long as the UAS is in communications range of any of the SoS components, the subscriptions and images between UAS and UGS will mediated using the service oriented middleware as described above in section II.

V. Design of high level control module in order to integrate UAS to SoS

The PCM implemented on the RPI embedded system must satisfy the following functional requirements:

- Ability to receive and handle the image data subscriptions from UGS and decide if the mission according to received subscription is feasible.
- Ability to dynamically compute a flight trajectory to the imaging positions and to follow the computed plan.
- Ability to develop local situation awareness and dynamically re-evaluate the generated plan in case the observed situation differs from the expected.
- Ability to control its camera during the mission.
- Ability to send images back to the subscriber.

On Figure 7 a general mission scenario flow is depicted.

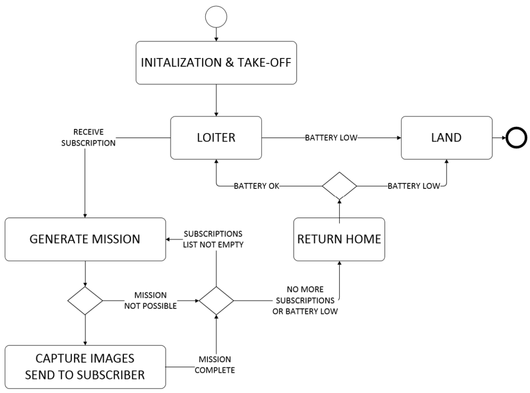


Fig. 7. Scenario description

After the UAS has completed the initialization, it can be switched to full autonomous mode and launched from hand. The UAS then starts loitering above its initialization position called HOME. When UAS receives a subscription for an image capturing mission and if battery voltage has not dropped below pre-set threshold and the distance to the target position indicated in the subscription is realistic, the UAS will proceed with the mission. If UAS receives more subscriptions, they will be handled by order of priority and the order of arrival. When the UAS has no more missions or battery voltage threshold has been exceeded it will return to HOME position and land.

VI. Pilot Control Module (PCM) Software

A. General architecture of PCM software

The general architecture of the software of the PCM is depicted on Figure 8. The PCM implemented on the RPI exercises high level control over internal and external UAS communication, message handling and mission control modules. The PCM creates both a serial connection to autopilot and a network connection to the remote network server, parsing relevant information from internal and external communication, maintaining the values of vehicle state variables and acting according to the current situation and the goal function. The data read from network socket can either be a stream of MAVLink messages from a Ground Control Station or a subscription from UGS (for safety reasons, the UAS can also be controlled and monitored over a Ground Control Station (GCS)). The stream of MAVLink messages is relayed directly via the serial connection to the autopilot.

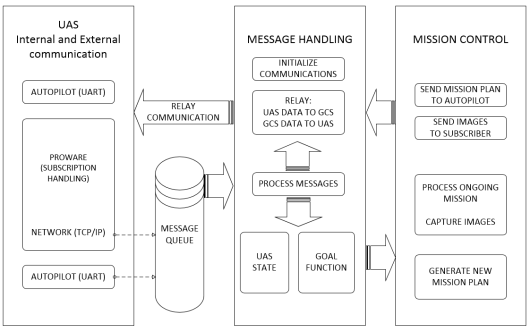


Fig. 8. Pilot Control module

In case the PCM receives a subscription over network from a UGS, the ProWare module will first check the data validity and then the PCM will call the relevant software functions from Mission Control module, update the goal function and produce a new mission plan which then is communicated to autopilot. The data read from serial connection is a stream of MAVLink messages from autopilot. It contains the UAS vehicle telemetry, system info, GPS data, mission related requests and reports. The MAVLink messages are parsed by the PCM and based on its type the information is used. From GPS data the PCM extracts current position and memorizes it until it is updated. Mission request and report messages from autopilot are used by the PCM for controlling the UAS mission. (Starting new mission, aborting the mission, control of camera, etc.). The vehicle telemetry and system info contains various information about vehicle attitude, movement (including airspeed and heading) and also battery voltage and current. All data read from serial port is also relayed to the Ground Control Station. Finally the Pilot Control module logs everything to a local file.

B. Mission Control Module

The Mission control module's tasks upon receiving appropriate commands from PCM are generating a new mission plan, uploading it to the autopilot, execution of the plan and managing the camera control. In order to generate a mission plan the following arguments from received subscription and the UAS state are used: Target position, its movement vector and the time of detection, the current location of the UAS, course and speed. The module produces a list of MAVLink mission plan items where the first item is a current position and the rest are items which will command the UAS to fly over the given target and capture the images. The navigational formulas for calculating distances, angles and new positions in WGS84 (World Geodetic System 1984) coordinate system rely on spherical geometry.

The Mission Control Module goes through 4 distinct steps during the generation of UAV mission:

1. A trajectory type for imaging mission is chosen.
2. GPS positions are calculated for the trajectory and imaging commands are added.
3. Positions for transit route to target area are computed.
4. The waypoints positions and imaging commands are transformed to MAVLink messaging format.

In the first step, the trajectory type for imaging mission is chosen. Next, the route to the target is generated. Currently it is a simple straight line to the target area. An appropriate path planning algorithm to find the optimal transit route, which would take no-fly zones or preferred routes into consideration, will be added in the future. The choice of the trajectory type is currently pre-fixed before UAS take-off, only the direction depends on the UAS approach. During the testing various trajectory types from straight line over target with several photo positions to an area coverage were tested. This approach is very robust as the both the size and direction of the trajectory types can be dynamically chosen. This remains for the future work.

C. UAS Payload/Camera Control

While the PCM relies on the autopilot telemetry and mission related communication, it can parse and memorize the mission commands that are designed for camera control and when autopilot reports back that a certain mission item has been reached and next waypoint item has been set to "current" i.e. autopilot starts following next waypoint in the list, the Mission Control module will take action and command the camera capturing according to the previously memorized messages.

VII. Tests Results

The test results described in current paper involve only subscriptions with stationary targets and the ground sensor system being emulated by a regular personal computer. Tests with live UGS devices communicating directly with the UAS will be conducted in the next phase of the project.

The purpose of the tests described in current paragraph is to validate the integration and use of UAS as a high fidelity information provider for the SoS of wireless ground sensor network.

Fig. 9. Captured image of stationary target.

The challenges encountered in the final outdoor tests showed what could have been predicted: such a system is greatly affected by the physical environment and most of these aspects cannot be easily simulated or predicted in a laboratory environment. An example typical image from the live outdoor tests is depicted on Figure 9 and Figure 10 indicates an emulated target position on Google Earth map.

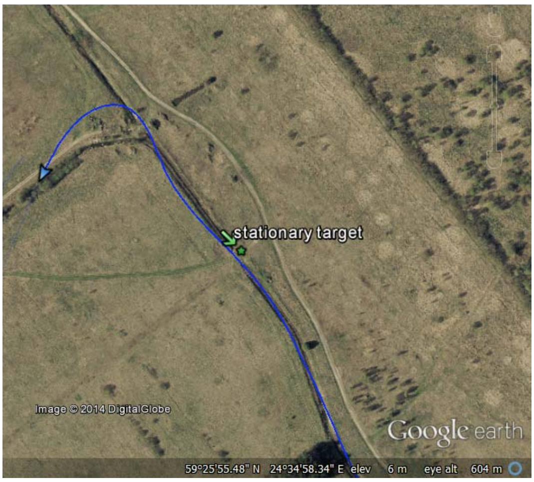


Fig. 10. Emulated stationary target and UAV track (blue line) on Google Earth map.

Although most of the time the results were stable meaning that UAS captured images on targets as indicated on Figures 9 and 10, but there were also difficulties. For example strong wind has an effect on UAS turn radius and speed of arrival at the target area, making it difficult to estimate the precise time of arrival to the target area. Another example is that outdoor environment and UAS movement also had an impact on the 3G modem communication. During laboratory tests we never experienced any communication interruptions, on the field on the other hand the 3G connection sometimes broke down and a routine had to be implemented to re-establish communication to the relay-server automatically.

The outdoor tests and analysis of the flight paths from several executed missions with UAS speed set to 20 m/s showed that its average turn radius is minimum 50 m. In case distance to target was more than 200 m and UAS had sufficient time to manoeuvre and fly directly over a stationary targets it had no problem capturing images from the positions indicated by UGS. On the other hand, in cases where UAS was receiving several subscriptions nearly simultaneously and the target positions in subscriptions were around or less than 200 m from each other and off the UAS immediate flight trajectory, see Figure 11, the UAS had to make very steep turns and the target was often not captured on images due to aircrafts tilt during the turns.

Fig. 11. UAS flight path in case of two consecutive subscriptions involving targets very close to eachother (targets are at wp-s 4 and 8, rest of the waypoints are generated for UAS guidance).

This problem will be handled in the future by installing small 1-axis gimbal on UAS and improving the planning dynamics and algorithm. In extreme cases one could imagine that UAS may miss the waypoint altogether, but this is handled by setting the waypoint achieved radius around waypoint position. This must be large enough considering UAS manoeuvrability and small enough considering UAS flying height and camera lens angle. Even if the UAS misses the waypoint, we saw during the tests that it will simply fly around and make another attempt.

VIII. Future Work

The work described in the paper was performed during the first year of the IN4STARS project, as the total duration of the project is three years, many continuing activities have been planned. It must be added also that the developed UAS architecture and the resulting SoS consisting of UGS and UAS establishes a good basis for planned upgrades and activities.

Additional field tests with the UAS platform are needed to increase the confidence in the platform. Live tests with operational UGS network can be started once the fielded UGS nodes are able to offer all the required functionality (i.e. classification and tracking of mobile objects of interest), which currently works in a lab environment and to a limited extent in the field. Tests with moving targets will follow the initial tests with stationary objects. An appropriate path planning algorithm for transit route to mission area and dynamic ability to choose between the different trajectory types (i.e. types for direct flight over a target or area coverage) for imaging mission depending on the precision estimation of the target position is also to be added in future. Also the design of a single axis gimbal for controlling the angle of the payload of the UAS was completed, but due to the time constraints this functionality was not implemented within the context of the work described in this paper.

Although the SoS communication principles have been validated in the lab and in field tests, the real validation of the communication principles described in the paper can be only performed in the field with real UGS devices operating in the physical environment. These tests are planned for the summer of 2015.

IX. Conclusion

In this paper we have explored the possibility to create a SoS solution where an autonomous UAS is integrated as one of the systems in a tactical SoS in order to carry out perimeter control. The contemporary UAV are mostly remotely operated, not autonomous and are not used as autonomous components in larger systems in a way described in current paper. We have developed a concept for integrating the UAS as an autonomous component in a SoS and shown the feasibility of the concept by conducting live tests with an implementation of the concepts on a real UAS. The created UAS solution is a robust and extendable platform for future work.

Reference

[1] S. Teh , L. Mejias , P. Corke and W. Hu "Experiments in integrating autonomous uninhabited aerial vehicles (UAVs) and wireless sensor networks", *Proc. Australasian Conf. Robot. Autom.*, 2008

[2] Pignaton de Freitas, E., et al. "UAV relay network to support WSN connectivity." Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010 International Congress on. IEEE, 2010.

[3] Pearre, Ben, and Timothy X. Brown. "Model-free trajectory optimization for wireless data ferries among multiple sources." GLOBECOM Workshops (GC Wkshps), 2010 IEEE. IEEE, 2010.

[4] P. Corke, R. Peterson, and D. Rus, "Coordinating aerial robots and sensor networks for localization and navigation," in Proceedings of the 7th International Symposium on Distributed Autonomous Robotic Systems, DARS 2004, Toulouse, France, June 2004, pp. 281-29.

[5] S. Bhattacharya, N. Atay, G. Alankus, et al., "Roadmap Query for Sensor Network Assisted Navigation in Dynamic Environments," in Proceedings of DCOSS, 2006.

[6] How, Jonathan P., et al. "Increasing autonomy of UAVs." *Robotics & Automation Magazine, IEEE* 16.2 (2009): 43-51.

[7] Preden, J-S., et al. "Data exchange for shared situation awareness." Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), 2012 IEEE International Multi-Disciplinary Conference on. IEEE, 2012.

[8] Motus, Leo, Merik Meriste, and Jurgo Preden. "Towards middleware based situation awareness." *Military Communications Conference, 2009. MILCOM 2009. IEEE*. IEEE, 2009.

[9] Riid, Andri, Jurgo Preden, and Sergei Astapov. "Detection, identification and tracking of mobile objects with distributed system of systems." System of Systems Engineering (SOSE), 2014 9th International Conference on. IEEE, 2014.

[10] Astapov, Sergei, et al. "Object detection for military surveillance using distributed multimodal smart sensors." Digital Signal Processing (DSP), 2014 19th International Conference on. IEEE, 2014

Appendix 4

IV

J.-S. Preden, **J. Kaugerand**, E. Suurjaak, S. Astapov, L. Motus, and R. Pahtma. Data to decision: pushing situational information needs to the edge of the network. In *Proceedings of the 2015 IEEE international multi-disciplinary conference on cognitive methods in situation awareness and decision*, pages 158–164. New York: IEEE, Orlando, FL, USA, 9-12 March 2015

Data to decision: Pushing Situational Information Needs to the Edge of the Network

Jurgo Preden, Jaanus Kaugerand, Erki Suurjaak, Sergei Astapov, Leo Motus, Raido Pahtma
Research Laboratory for Proactive Technologies
Tallinn University of Technology
Tallinn, Estonia
jurgo.preden@ttu.ee

***Abstract*—Obtaining a high level of situation awareness while maintaining optimal utilization of resources is becoming increasingly important, especially in the context of asymmetric warfare, where information superiority is crucial for maintaining the edge over the opponent. Obtaining an adequate level of situational information from an ISR system is dependent on sensor capabilities as well as the ability to cue the sensors appropriately based on the current information needs and the ability to utilize the collected data with suitable data processing methods. Applying the Data to Decision approach for managing the behavior of sensor systems facilitates optimal use of sensor assets while providing the required level of situational information. The approach presented in the paper combines the Data to Decision approach with the *Fog Computing* paradigm, where the computation is pushed to the edge of the network. This allows to take advantage of Big Data potentially generated by the sensor systems while keeping the resource requirements in terms of bandwidth manageable. We suggest a System of Systems approach for assembling the ISR system, where individual systems have a high level of autonomy and the computational resources to perform the necessary computation tasks. To facilitate a composition of a System of Systems of sensors for tactical applications the proactive middleware ProWare is applied. The paper presents the results on the implementation of a sensor solution that facilitates on-line sensor cueing and collaboration between sensors by building upon the *Fog Computing* paradigm and utilizing the Data to Decision concepts in the context of the European Defense Agency project IN4STARS.**

Keywords—ISR; situation awareness; middelware; System of Systems; data to decision; fog computing

I. INTRODUCTION

Good situation awareness at all levels in military operations is more critical than ever as in the asymmetry of operations we have to rely on information superiority to maintain an advantage. Modern technology offers us tools for collecting abundant amounts of sensor data with relatively reasonable costs. The challenge is aggregation of the data, abstraction of data to information and identifying the data sources needed for generating situational information on the area and topic that is needed. To deliver an operational solution, one must look at the entire information processing chain from the sensor sources to the information consumer not just at the operation of individual components. As we must strive to provide the best situation awareness possible to every warfighter, the information consumers may be diverse, starting from dismounted soldiers to commanding officers in a base, the ISR system must be able to cope with the needs of these diverse users. It has previously been suggested that for efficient system operation one should start looking at the communication chain from the information consumer side, identifying the situational information needs of the information consumer. As the information needs of individual users change over time depending on their location and the type of mission the need to accomplish the system must adapt to the changing needs of the user.

The high availability of ground sensor assets and communication technologies presents the opportunity to use the sensor assets for very high granularity sensing, achieving high quality of data with a large number of sensing nodes with relatively mediocre capabilities. However, pursuing this approach presents many theoretical and practical challenges, such as bandwidth allocation, asset management and coordination of data flows. One possible approach to be used is to push the computation to the edge of the network, thereby reducing bandwidth requirements and computational capabilities needed at central locations.

The concept of Data to Decision (D2D) is used to characterize decision making scenarios, where potentially the data sources are able to provide an abundant amount of information and where it is difficult to assemble the appropriate collection of data for rapid decision making [1]. D2D highlights the collection and fusion of actionable information to provide adequate situational information for assessing options, threats and consequences of decisions [2]. Although D2D concepts can be applied at all levels of an organization, these concepts are in particular applicable to individuals on the edge, who, with the aid of modern mobile information and communication platforms, can potentially have access to real-time actionable information. In addition to military applications, providing actionable information to operators in the field is also very critical in emergency response and law enforcement, where adequate situation awareness (based on real-time correct information) to aid rapid decision making is critical.

Combining the D2D approach with the *Fog Computing* paradigm proposed by Cisco [11] and also previously as a distributed computing approach by the authors of the current

The work presented in this paper was partially supported by the Estonian Ministry of Defence and the European Defense Agency project IN4STARS..

paper [12] enables to take advantage of a large number of sensing nodes while overcoming technical challenges and provide to the consumers the required situational information needed with optimal use of resources.

Optimizing the information flows using the D2D approach and delivering only information currently needed reduces the overload for the operator, which is a serious challenge with the flood of data provided by modern ISR systems [7].

Pushing computation to the edge has many clear system level benefits, such as optimized use of sensor resources as well as reduced bandwidth requirements, which is a serious factor in modern systems [6].

One cannot assume that the systems making up an ISR SoS are developed and deployed as a complete system, instead the individual systems may be deployed at different times and they may not be owned by the same actor, e.g., in case of a coalition operation there may be a desire to cross-use sensor assets between coalition partners. The coordination between the individual systems must be realized at the individual system level as involving central coordinating authorities may be quite complex as one can't expect to have global knowledge of individual assets and their capabilities. This means that the individual systems must be able to use services from other systems without the need for manual configuration.

In the European Defense Agency's IN4STARS project the Research Laboratory for Proactive Technologies is developing a sensor system, which is able to adapt its configuration and behavior and provide data according to the consumer needs. This follows the data to decision approach, where the data delivered (and the level of sensing performed) by the sensor system behavior depends on the information needs as expressed by the user.

The paper is organized as follows: Section II presents the various challenges for ISR systems, section III presents the system concept suggested by the authors, section IV describes the components of the experimental system used to validate the concepts and section V describes the architectural approach for the system.

II. ISR CHALLENGES

A. Aggregating diverse data

A modern ISR system may be composed of a diverse set of data and information sources, including both hard and soft data sources. In order to provide the decision maker or an operator with an adequate level of situational information the information from these data sources must be harmonized, synchronized and validated for combining it into usable information.

With these diverse data and information sources these ISR systems can be called Systems of Systems (SoS) with humans in the loop, as humans are an important part of these SoS, both using the information provided by the SoS as well as feeding their analysis results back to the SoS. The SoS provides the resulting information to both humans and machines.

In [1] the authors suggest to use CNL (Controlled Natural Language) to task the information processing chain for information required by the information consumer. Such an approach makes the human an active part of the SoS, where the human makes additional information requests based on information received from the SoS components, thereby influencing their behavior.

In [1] an asset catalogue is used, based on the data and information needs the assets from the catalogue are accessed via services. We propose a more distributed architecture, where the information needs are pushed to the network and within a sub-network the tasks are distributed and the assets tasked. This approach can be used both for hard as well as for soft information sources.

The Recognition Primed Decision (RPD) making model develops the (human) user decision making capability based on the current situation and past experiences [3]. The RPD model shows the goals of the user and the cues that are important, this information can be used to prime the information acquisition and processing systems. In [4] the authors describe the Cognitive Observe-Orient-Decide-Act model as a method of user and team analysis in the context of the Data Fusion Information Group (DFIG) Information Fusion Model. The C-OODA model engages the human into an iterative information processing cycle, which involves the information processing steps performed by machines. The objective is to reduce information uncertainty. In [4] the authors even attempt to apply control theory elements to describe the information processing of the human in the C-OODA loop.

B. Big data challenge

Big Data is not only a challenge in the business world but it has presented a serious challenge in the intelligence domain for some time. The issue has become even more acute with the even greater proliferation of sensor systems [6]. For example US Lieutenant General David A. Deptula (while being the Deputy Chief of Staff for ISR, U.S. Air Force) has stated that the flood of data originating from diverse sensor sources makes the DoD operators and imagery analysts, who have to monitor these information feeds from sensor assets, swimming in sensors but drowning in data [7]. While this challenge may be addressed by adding additional manpower for the analysis task, alternative solutions of automatic information processing offer a more efficient solution.

As various ISR assets are able to provide abundant amounts of information, the question arises who consumes that information and what type of activity is the information consumer interested from a specific region. By tasking the sensor assets (including both ground based and airborne assets) with specific objectives for information acquisition the processing of the data can be also pushed further to the edge of the network, or processed *in situ* on the sensor systems themselves. This would allow to communicate only the higher level situational information from the sensor systems to higher levels of the network, thus reducing bandwidth requirements and the amount of processing and analysis capabilities needed at central locations.

As these challenges are also very relevant in the Internet of Things domain, Cisco researchers have coined the term *Fog Computing* for describing a computing paradigm, where the *Big Data* is processed near the ground (i.e., close to the data sources) as opposed to processing the data in the Cloud, which is the typical approach [11]. As this paradigm is bringing the Cloud computing concepts closer to the ground and a cloud close to the ground is called fog the term *Fog Computing* seems appropriate. Thus, *Fog Computing* makes it possible to utilize Big Data, but as the computation is pushed to the edge of the network the requirements for bandwidth are reduced and the computational need at the central locations is reduced. Cisco foresees that *Fog Computing* is a platform that provides computation, storage and networking between end devices and traditional Cloud Computing nodes [11]. This approach allows to make use of the capabilities of the computational hardware that is part of individual nodes, while minimizing bandwidth requirements as data is processed close to the spot where it is generated. In time critical applications (where the information may be required for rapid decision making) the latency of the computation is critical and by moving the computation to the edge of the network the latency can be reduced as the number of communication hops the data and the information have to travel becomes smaller.

In order to implement the *Fog Computing* paradigm in the ISR context, the data processing that is performed in the field must adapt to the information needs of the information consumer. The reason for this requirement lies in the fact that the combination of all possible information needs is so high that not all the information that may be needed by information consumers can be generated all the time. Thus the information generated (i.e., computed by the edge nodes of the network) must be tailored to the current information needs of the consumers. This can be only done if the information needs are communicated from the consumer to the edge of the network, to the individual sensor nodes.

One benefit resulting from applying the *Fog Computing* paradigm is in the reduced power requirements of the individual systems. As sensing is performed and data is communicated only when there is need for it and the level of abstraction of the information is increased to the highest possible level at the far edge of the network, the reduced amount of data and information communicated also reduces power consumption of involved systems, extending the lifetime of battery powered systems.

A practical example of how the behavior of sensor systems can be adjusted based on situational information needs can be brought in the context of a sensing system on the ground, monitoring a road. Clearly the information needs are highly dependent on the current situation. If the road has been cleared from IEDs the previous day for a convoy passing the next day, any movement information (human or vehicle) on that road that occurs after the counter IED team has finished its operation is relevant until the convoy has safely traversed the road. If suspicious movement is detected, the sensor system on the ground can task a UAS to provide additional information on the detected object, thus providing better situational information to the information consumer. After the convoy has reached its destination, the operators are probably not interested in receiving information on normal civilian traffic occurring the road. Similarly, in case of a public road with regular traffic, the operators might be interested in an abnormally high number of heavy vehicles passing along that road. So the sensing systems deployed on that road can process the data locally and send high level information on only these detected events, which the information consumers have designated as events of interest. The behavior of the sensor systems adapts to the information needs as expressed by the information consumer, so while still capturing large amounts of data we can take advantage of the Big Data, but we constrain it to the edge of the network.

C. Sharing high demand, low availability assets

While the ground based sensor assets have oftentimes high availability due to low price and simple deployment, some of the more capable airborne assets, which are in high demand have lower availability due to high price and demanding support infrastructure required for them. Sharing of these high demand, low availability assets presents a serious challenge. In [5] this aspect of ISTAR (Intelligence, Surveillance, Target Acquisition and Reconnaissance) is discussed. One possible solution for alleviating this problem is dynamic management of the high demand ISR. In [5] the authors suggest the use of an ISTAR Manager for managing the operation of the assets. The authors suggest that use of visualization and situation awareness tools by ISTAR Manager, combined with decision support tools would improve the utilization of ISTAR assets. Extending this concept the authors of the current paper suggest that every low availability asset could posses a manager component, which manages its operation based on the priorities of the information requests received by the asset.

As missions can be highly dynamic, the information needs and priorities of the individual information consumers may change as the mission progresses. The needs of the individual consumers should reach the sensor assets providing the information as the needs are changing. This does not only concern the sensor modalities involved in generating information from sensor data, but also the data reduction and processing techniques used at the start of a mission may not be appropriate for later stages in the same mission [6]. The added complexity of potentially several parallel missions by multiple coalition partners sets even higher demands to the information acquisition, processing and communication systems.

III. SYSTEM CONCEPT

The section presents the concept for the ISR SoS prototype developed in the context of the IN4STARS project, discussing the roles of the individual systems and the flow of information between the systems.

As stated in the introduction, the systems making up the SoS are assumed to have a high level of autonomy and they are assumed to provide the collected data or generated information via services. This means that information consumers can have access to the information generated by the ISR system by subscribing to appropriate services. If no service subscriptions have been made to a system, it will not send out any information, being in a power saving mode instead.

The consumer must not specify exactly from which asset the information must originate, it may request just information with the optimal granularity and the ISR SoS may decide what sensor sources, what modalities and what algorithms it will apply to provide the situational information.

Unlike most systems that assume a central coordinating agent, the sensor system architecture we apply builds upon a System of Systems approach, where the individual systems are autonomous. When a request for information is made to the SoS, any node that is capable to provide the requested information with an acceptable cost will respond to it. The specific sensor modalities needed for providing the requested information (e.g. detection and identification of tracked vehicles) need not be co-located with the system providing the information, instead the information may be fused from several sources. To enable this kind of operation the nodes must maintain a certain level of self awareness as well as awareness of the SoS, in order to find the required sensor sources for generating the information requested by the information consumer. In order to enable this kind of system operation the individual systems must be able to communicate directly and to request services from other systems. The conceptual system configuration is depicted on Figure 1.

A. *Information flow*

Applying the D2D approach in a Fog Computing paradigm means that the requests for situational information made by the information consumer can be directed to the sensor assets in the field, closest to the area of interest. The routing of information to the specific information provider may be done using many alternative methods, e.g., geo-routing, using a central service directory or some other service discovery mechanism. The requests may be passed through a server, if an architecture requires that, but there is no need for a central server or coordinator. Based on the information requests, the algorithms are primed in the computing device providing the information service (e.g., sensor or fusion node). Service requests are made to the data sources (sensor nodes) from which data is needed for computing the requested situational information. Once the information has been computed it is provided to the consumer.

The data and information flows set up based on situational information requests may involve several sensor modalities and sensor nodes. Let's consider a case where the information on tracked vehicle movement is requested from a specific area. The initial detection of a vehicle can be performed by a movement sensor using PIR technology, which has extremely low power consumption requirements. Once an object has been detected by the movement sensor, it can trigger the operation of an acoustic array. The acoustic array is much more capable in terms of object type and location identification, so it is able to identify the type of object, its speed and approximate location. If the object is of type that is of interest to the information consumer, the acoustic array node can notify a UAS to provide visual information on the detected object. Depending on the availability of the UAS resources, it can fly to the indicated area, acquire images from the area and provide them to the acoustic array on the ground, which can provide it then to the consumer(s) that have requested information from the specific area.

Figure 1 Conceptual system configuration and information flows

The information consumer in this scenario can be for example an analyst located far away from the area of operations or a dismounted soldier, conducting an operation in the area of question. The paths that the information must take to reach the intended recipient may be complex but we assume that modern routing methods are able to cope with this dynamically. The dismounted soldier has low latency requirements and clearly he will be using the information for tactical purposes, forming local situation awareness. The analyst in this case will have more complete information as the analyst may have access to additional information sources from adjacent areas and from other information sources (e.g., HUMINT, OSINT).

IV. SYSTEM COMPONENTS

The sensor systems used in the prototype ISR system assembled for the IN4STARS project feature several modalities. The sensor systems can be categorized to ground sensors and airborne sensors, below both types are described and the operation of the systems discussed.

A. Ground sensors

The ground sensor systems are of the following modalities: movement, image and acoustic sensors. All sensor systems are autonomous, enabling collaboration between sensor systems using the proactive middleware ProWare, discussed in more detail in section XX. While the behavior of the movement sensor and the image sensor is quite simple - these sensors systems just provide a specific type of output, the collaboration between the individual acoustic arrays is more complex.

The object localization solution based on acoustic arrays utilizes autonomous acoustic arrays working together for localizing detected objects. The same arrays can be also used for acoustic classification using any of the available classification methods as we have also presented in our previous work [13]. For acoustic localization on the ground the UGS systems are placed in the horizontal plane and localization is performed by estimating object coordinates in the plane (x,y) of objects emitting sound. Each system is equipped with two acoustic sensors spaced at a specific distance l from one another, forming a small acoustic array. Estimation of the angle of the observed object in relation to the array is based on estimating the time delays of acoustic wave arrival to the sensors, also called Time Difference of Arrival (TDOA). The Direction of Arrival (DOA) of sound from a specific acoustic source is calculated using TDOA. The individual acoustic UGS systems form a SoS, which is able to estimate the regions where the observed object is located using DOA information from several systems.

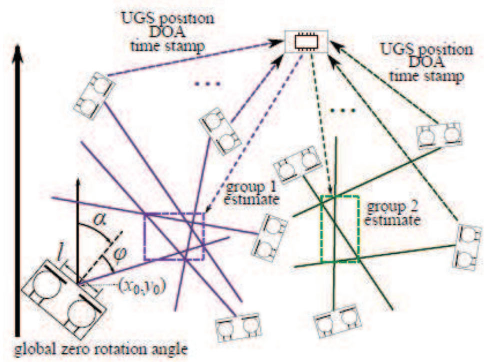


Figure 2 Acoustic source localization with a SoS consisting of acoustic arrays

The acoustic array systems are partitioned into groups, each group having a common Field of View (FOV), i.e. all arrays in a single group must observe the same area as depicted on Figure 2. Group partitioning is performed by clustering, taking two aspects into consideration. Firstly, UGS must be facing in the common direction as the considered localization procedure uses a directional approach. In this regard, the observed area is not necessarily enclosed by UGS, as shown on Figure 2, but may be observed from one or several sides. Secondly, a group must have certain homogeneity. UGS located too far from the group's centroid may be useless to the localization effort in low Signal to Noise Ratio (SNR) environments or when the sound emitted by the source of interest is too weak. Furthermore, non-homogeneous groups present additional challenges for wireless communication.

The acoustic arrays for a small SoS, which can be triggered as a group, providing the compound computation result to the information consumer requesting the information.

The operation of a UGS group can be triggered by a movement sensor (as described previously), which has detected movement in a specific area, if a request for information has been received from an information consumer.

B. UAS

The UAS employed in the scenario is fully autonomous, requiring human assistance only for takeoff. The authors acknowledge that operational UAS-s do not feature this level of autonomy and will not in the near future, but the approach used for cueing the UAS and for exchanging the information can be also used in case of a UAS with a man in the loop.

The UAS used in the experimental system is a micro UAS with a maximum payload of 200 grams and a top speed of 100 km/h. The UAS is equipped with a camera that operates in the visible light frequency. The UAS is capable of autonomous operation including computing a flight trajectory, following the trajectory and acquiring images at desired locations. Any information consumer (including a ground based sensor system requiring additional information) can subscribe to information from the UAS. As the UAS is just one of the sensor systems in the SoS, communication to and from it in respect to the sensor data follows the same scheme as with other sensors - in case visual information is needed to augment the situation assessment, a ground sensor system that requires the information may request the image from the UAS. If the high-level control module in the UAS decides to serve (the priority of the request is higher than the priorities of previous requests) the incoming information request a new mission is planned and loaded to the mission control module. Once the mission is executed and the images have been acquired at the specified location, the acquired images (or situational information inferred from the images) are delivered to the ground sensor system, which is then able to deliver the visual information together with the classification and tracking information to any of the information consumers that have subscribed to situational information from the specific region

The architecture of the UAS software is depicted below on Figure 3.

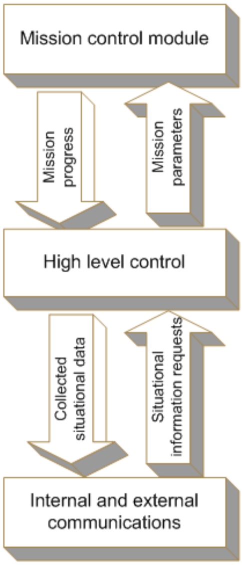


Figure 3 UAV software architecture

The high level control module plans the missions based on the incoming information requests and manages communication with sub-systems of the UAS. The high level control module also decides when and what collected information should be communicated to which information consumer.

The internal and external communication module maintains the message queue from the various sub-systems of the UAS as well as the external systems. The high level control module uses the information delivered by the messages to maintain adequate situation awareness and plan the operation of the UAS accordingly, for example the .

The mission control module executes the current mission as planned by the high level control module. The mission control module is responsible for controlling the trajectory and flight dynamics of the UAS, providing feedback to the high level control module on the progress of the mission.

V. SYSTEM ARCHITECTURE

A. Representation of Situational Information

Building upon the situation awareness model introduced by Endsley [8] we have proposed a situation awareness model for a distributed computing system in [10].

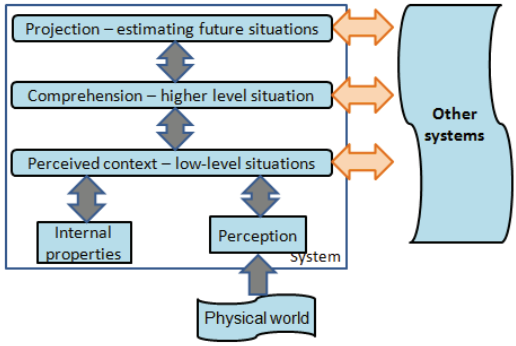


Figure 4 Exchange of situational information in a hierarchy of situations

The diagram on Figure 4 illustrates how the sensor data as well as the intermediary situational information computation results (situation parameters) can be exchanged with other systems. Both the end result of the situational information processing as well as the intermediary computation results can be used as triggers for sensor systems or as input to the information fusion process.

B. Proactive middleware

The proactive middleware ProWare [10] is used to realize the interactions between the individual systems making up the ISR SoS. ProWare relies on the concept of data mediators (ProWare components located on very system that is part of the ISR SoS) to ensure the correctness of data and the resilience of the SoS. The mediator associated with every system is responsible for communication of data to and from the system. A subscription-based data exchange model is used, as due to the unknown structure of the SoS, the data exchange partnerships must be formed dynamically in the form of subscriptions. While discovering a data provider and subscribing to data from that provider, the data consumer also communicates the temporal and spatial constraints for the data. These constraints are observed both by the producer mediators before the data is communicated from the producer to the consumer. In a similar way the mediator component at the consumer side validates that the data received (still) satisfies the (temporal, spatial and others) constraints specified by the consumer process [10]. The ProWare mediator configuration and data exchange setup is depicted below on Figure 5. Any system can assume the role of a consumer or a produced depending on the data and information needs of the system and the data and information produced by the system.

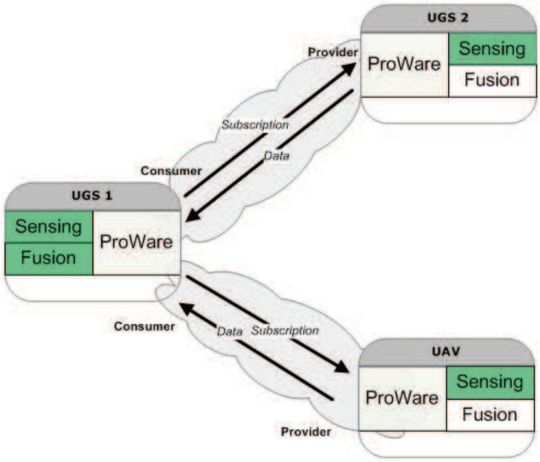


Figure 5 ProWare in data exchange scenario

We have shown the viability of this data mediation approach and its ability to ensure temporal and spatial correctness of data in simulations [5], [6]. As only data that satisfies the requirements of the detection, identification and tracking algorithms is communicated from the producer system to the consumer system, also the bandwidth requirements are potentially reduced.

In the work done in the context of the IN4STARS project, presented in the current paper, the ProWare components have been on embedded nodes, including UGS and UAS and on the information consumer side.

CONCLUSION

The work presented in the paper builds upon years of previous work on the topic. While the solution presented in the paper is an experimental one, the principles can be clearly applied for the creation of an ISR system where the computation is pushed to the edge of the system, yielding the benefits described in the paper. By combining D2D concepts with the *Fog Computing* paradigm it is possible to optimize the utilization of sensor assets and network resources, while providing high quality situational information to the consumer.

REFERENCES

[1] Preece, A. ; Pizzocaro, D. ; Braines, D. ; Mott, D. ; de Mel, G. ; Tien P., "Integrating hard and soft information sources for D2D using controlled natural language", 15th International Conference on Information Fusion (FUSION), 2012

[2] B. Broome, "Data-to-decisions: a transdisciplinary approach to decision support efforts at ARL," in Proc Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR III (SPIE Vol 8389). SPIE, 2012.

[3] R. Breton and R. Rousseau, "The M-OODA : Modelling of the OODA loop as a modular functional system." Def. Res. and Dev. CA-Valcartier, DRDC TM 2008-130, May 2008.

[4] Blasch, E.P. ; Breton, R. ; Valin, P. ; Bosse, E., "User information fusion decision making analysis with the C-OODA model", Proceedings of the 14th International Conference on Information Fusion (FUSION), 2011, pp 1 - 8

[5] Pearson, G; "Dynamic management of layered ISR systems", Proc. SPIE 8389, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR III, 2012.

[6] Hershey, P.; Wang, M; "Composable, distributed system to derive actionable mission information from intelligence, surveillance, and reconnaissance (ISR) data", IEEE International Systems Conference (SysCon), 2013, pp 460 - 467

[7] Lt. Gen. David A. Deptula, Keynote Address, GEOINT 2009, Oct. 2009.

[8] Endsley Mica R and Garland Daniel J "Situation awareness analysis and measurement", London : Lawrence Erlbum Associates, Publishers, 2000.

[9] Preden, J.; Motus, L.; Meriste, M.; Riid, A. (2011), "Situation Awareness for Networked Systems", IEEE International Multidisciplinary Conference on Cognitive Methods in Situation Awareness and Desicion Support - IEEE COGSIMA 2011, Miami, USA, Feb. 22-24, 2011

[10] Preden, J; Llinas, J.; Rogova, G.; Pahtma, R.; Motus, L., "On-line data validation in distributed data fusion.", SPIE Defense, Security and Sensing, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR IV. (edit) T. Pham; M. A. Kolodny; K. L. Priddy. SPIE, 2013, (Proceedings of SPIE; 8742), 1 - 12.

[11] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli, "Fog computing and its role in the internet of things", In Proceedings of the MCC workshop on Mobile cloud computing (MCC '12). ACM, New York, NY, USA, 13-16.

[12] Preden, J.; Motus, L.; Pahtma, R.; Meriste, M. "Reducing Bandwidth Requirements and Optimizing Data Flow in Distributed Data Acquisition and Processing", 2013 IEEE International Multi-Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), San Diego, Ca, USA. IEEE Publishing , 2013, 178 - 185.

[13] S. Astapov and A. Riid, "A hierarchical algorithm for moving vehicle identification based on acoustic noise analysis," in Proc. 19th Int Mixed Design of Integrated Circuits and Systems Conf, 2012, pp. 467–472. (2012)

Appendix 5

V

S. Astapov, J. Berdnikova, J. Ehala, **J. Kaugerand**, and J.-S. Preden. Gunshot acoustic event identification and shooter localization in a wsn of asynchronous multichannel acoustic ground sensors. *Multidimensional Systems and Signal Processing*, 29(2):563–595, 2018

Multidim Syst Sign Process
DOI 10.1007/s11045-017-0479-7

Gunshot acoustic event identification and shooter localization in a WSN of asynchronous multichannel acoustic ground sensors

S. Astapov[1] · **J. Berdnikova**[2] · **J. Ehala**[1] · **J. Kaugerand**[1] · **J.-S. Preden**[1]

Received: 17 February 2016 / Revised: 20 December 2016 / Accepted: 15 February 2017

Abstract Gunshot acoustic localization for military and civilian security systems has long been an important topic of research. In recent years the development of Wireless Sensor Network (WSN) systems of independent Unmanned Ground Sensors (UGS) performing distributed cooperative localization has grown in popularity. This paper considers a shooter localization approach based on gunshot Shockwave (SW) and Muzzle Blast (MB) event time and Direction of Arrival (DOA) information. The approach accounts for acoustic events Not-of-Interest (NOI), such as target hit noise, reflections and background noise. UGS perform gunshot acoustic event detection and DOA estimation independently; the information regarding every detected shot instance is sent through the WSN to the fusion node, which performs event identification and calculates the shooter's position. The paper presents a solution to identifying SW and MB among NOI events at the stage of information fusion. The considered approach treats the information gathered from different UGS separately, and thus does not require precise synchronization between the UGS. For DOA estimation, an

This research was supported by the European Defense Agency project IN4STARS and the Estonian IT Academy program.

✉ S. Astapov
sergei.astapov@ttu.ee

J. Berdnikova
julia.berdnikova@ttu.ee

J. Ehala
johannes.ehala@ttu.ee

J. Kaugerand
jaanus.kaugerand@ttu.ee

J.-S. Preden
jurgo.preden@ttu.ee

1 Laboratory for Proactive Technologies, Department of Software Science, Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia

2 Thomas Johann Seebeck Department of Electronics, Tallinn University of Technology, Ehitajate tee 5, 19086 Tallinn, Estonia

Published online: 23 February 2017

algorithm designed for circular microphone arrays is proposed and compared with the SRP-PHAT localization algorithm. It is shown to provide adequate DOA estimates, while being more computationally effective. The proposed shooter localization approach is tested on real signals, acquired during three live shooting experiments. It is shown to succeed in localizing the shooter's position with a mean accuracy of 0.87 m for 30 shots at the range of 35 m, and just above 7 m for 37 shots at the range of 100 m.

Keywords Shooter acoustic localization · Circular microphone arrays · DOA estimation · SRP-PHAT · Wireless Sensor Networks

1 Introduction

Active development of shooter acoustic localization systems has continued for more than three decades. Numerous different gunshot detection and direction estimation systems are currently available for military applications of sniper and covert enemy force positioning, and are also used in law enforcement for gun violence reduction and forensics (Aguilar 2013). The devices currently available are generally standalone systems, composed of a single microphone array, e.g., the vehicle-mountable Boomerang system (Mazurek et al. 2005). Individual gunshot detectors, developed for military and law enforcement personnel (George and Kaplan 2011; Sallai et al. 2013; George et al. 2014), consist of compact shoulder-carried, helmet or uniform mounted sensors. Such individual systems increase local situation awareness, however, for large area coverage a different approach is required.

Modern Military Intelligence, Surveillance and Reconnaissance (ISR) systems apply distributed Unmanned Ground Sensors (UGS) interconnected through a Wireless Sensor Network (WSN) for large area coverage. UGS perform local situation assessment, and through data fusion a global assessment over the whole monitored area is made. A distributed system configuration expands UGS collective Field of View (FOV) and thus is well suited for shooter localization. The state of the art in this area suggests either synchronous (Sallai et al. 2011), or asynchronous (Damarla et al. 2010) gunshot acoustic event detection and subsequent shooter localization based on UGS collective information. The majority of the proposed approaches are based on the supersonic bullet's shockwave (SW) and muzzle blast (MB) analysis (Millet and Baligand 2006). Most methods employ single-sensor UGS which identify the gunshot events and estimate the shot geometry under different initial assumptions, e.g., the known caliber of the fired projectile in Sallai et al. (2011), or a certain ballistic shockwave acoustic model in Aguilar et al. (2007). However, initial assumption inconsistency and the presence of acoustic events Not-of-Interest (NOI) may significantly reduce localization accuracy (Ash et al. 2010). (By NOI events we denote residual gunshot acoustic events and various noise produced by other sources.)

Employing multichannel smart sensors for gunshot localization allows to additionally estimate the Direction of Arrival (DOA) of gunshot event acoustic waves. Knowing the DOA aids in acoustic event identification and allows to reduce the number of initial assumptions, which, in turn, makes the localization process more robust. In this paper we propose a method of shooter localization based on gunshot event DOA and Time Difference of Arrival (TDOA) information. The method is intended for operation in a WSN which consists of interconnected UGS, equipped with sensor arrays, and information fusion nodes. Each UGS independently performs gunshot acoustic event detection, computes the DOA and fixates event occurrence time in its own local time. The fusion node gathers DOA and time information from all the UGS which it governs, performs identification of SW and MB among NOI events, calculates the TDOA between SW and MB, and estimates the shooter position based on the UGS known positions. The distribution of computational tasks among the UGS and fusion nodes reduces the risk of any network component being overloaded, and the use of several fusion nodes eliminates the single point of failure and bottleneck effects. The TDOA are calculated per each UGS and no cross-UGS delays are used, thus node synchronization is not required (however, node clock divergence still needs to be roughly estimated for the fusion node to be able to distinguish between shot instances). An asynchronous approach is explicitly targeted due to the fact that long-lasting precise node synchronization cannot be guaranteed in WSN, especially in ones adopting the dynamic ad-hoc topology. For DOA estimation we apply a reduced computational cost approach presented by us in Astapov et al. (2015a), and a well known, effective, but computationally expensive localization algorithm of Steered Response Power (SRP-PHAT) for comparison.

Circular microphone arrays were chosen for the UGS implementation to allow for a 360° horizontal Field of View (FOV). Two prototype versions were created: the first one employs six condenser microphones and an exterior Data Acquisition Device (DAQ); the second one employs six MicroElectroMechanical Systems (MEMS) microphones and a BeagleBone Black as a DAQ and processing unit. The proposed method is tested on signals acquired during three live shooting experiments. The first experiment was performed at a small outdoor shooting range with a shooter-target distance of 35 m. The signals were acquired by four UGS of prototype 1. The second and third experiments were performed at a larger outdoor shooting range with a shooter-target distance of 100 m. The signals were acquired by six UGS of prototype 2. The experimental results indicate the feasibility of the proposed localization method in terms of gunshot event detection, NOI event elimination and shooter position estimation.

The remainder of the paper is structured as follows. Section 2 introduces the applied gunshot geometry model. Section 3 discusses problems situated with shooter acoustic localization, while examining several gunshot scenarios and localization approaches. Section 4 handles the proposed shooter localization method, reviewing the gunshot acoustic event detection, DOA estimation and information fusion procedures. Section 5 presents the UGS prototypes and experimental results. Section 6 is devoted to the discussion and thoughts on future developments. Finally, Sect. 7 concludes the paper.

2 Gunshot acoustic components

For our shooter localization approach we adopt a planar gunshot acoustic event geometry model (i.e., the sensor and the trajectory of the traveling bullet are situated in the horizontal plane). Figure 1 portrays the acoustic events produced by a gunshot at point Z, as observed

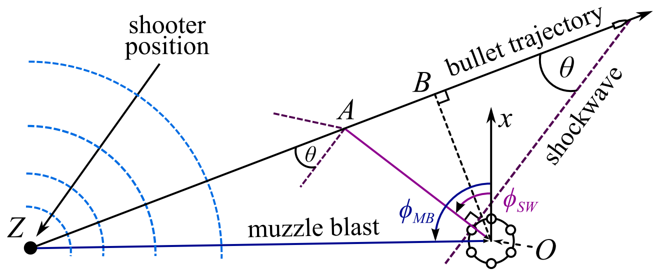


Fig. 1 Gunshot acoustic event geometry in the horizontal plane

at point O. For simplicity purposes we assume straight bullet trajectory, not accounting for effects considered in exterior ballistics (Carlucci and Jacobson 2010). A gunshot is characterized by the shockwave, produced by a supersonic projectile, and the muzzle blast of the fired weapon. SW produces a conical wavefront at an angle θ to the bullet's trajectory. The angle θ depends on the speed of sound c in air and the bullet velocity v:

$$\theta = \sin^{-1}\frac{c}{v}. \tag{1}$$

The waves of MB, on the other hand, propagate spherically at speed c in all directions.

The initial bullet velocity is equal to the muzzle velocity v_0 (i.e., the velocity at which the bullet leaves the muzzle of a gun), which depends on the bullet caliber and cartridge type and can be approximated for different firearm types (Carlucci and Jacobson 2010). Bullet velocity v decreases with flight distance due to air friction. It can be expressed as a function of traveled distance d_f as

$$v(d_f) = \left(v_0^{\eta} - 2\eta C_b^{-1} d_f\right)^{1/\eta}, \tag{2}$$

where, C_b is a ballistic constant, which depends on the bullet's type, and η is the exponent value, usually set at 0.5. We assume function (2) to be unknown and rather estimate the bullet velocity using the procedure described in Sect. 4.3.3. For small firearms (e.g., rifles) the decrease in the $v(d_f)$ curve can be considered linear and ultimately insignificant for the travel distance of 100–200 m (Carlucci and Jacobson 2010). Thus, for the rest of the paper we denote the bullet velocity as a range-invariant parameter v. The speed of sound in air c, on the other hand, depends on the ambient temperature. For an open environment it is calculated as

$$c = 331.45\sqrt{1 + t^{\circ}/273}, \tag{3}$$

where, t° is the temperature in degrees Celsius.

At line-of-sight, the sensor at point O detects MB at the time

$$t_{MB} = t_{shot} + \frac{d_{Z,O}}{c}, \tag{4}$$

where, t_{shot} is the time of shot, and $d_{Z,O} = \|Z - O\|$ is the Euclidean distance between points Z and O. Acoustic waves of SW originate from the bullet itself and not from the muzzle. SW travels outwards from the bullet's trajectory and is approximated as a planar wavefront in the horizontal plane. As the bullet has reached point A at speed v, the SW wavefront propagates from point A at speed c and reaches point O at the time

$$t_{SW} = t_{shot} + \frac{d_{Z,A}}{v} + \frac{d_{A,O}}{c}. \tag{5}$$

Point A here is such a point on the bullet's trajectory, from where SW will travel directly to point O at an angle θ relative to the bullet's trajectory (see Fig. 1).

The TDOA between SW and MB acoustic events can then be expressed as

$$\Delta t = t_{MB} - t_{SW} = \frac{d_{Z,O}}{c} - \frac{d_{Z,A}}{v} - \frac{d_{A,O}}{c}. \tag{6}$$

The distance from the sensor at point O to the bullet's trajectory ($d_{O,B}$ in Fig. 1) is called the miss distance. Whether Δt is positive depends on the bullet's velocity and the miss distance. In case of a shot fired from a rifle (average bullet velocity near or greater than mach 2) in the sensor's direction with the miss distance small enough, Δt is expected to be positive, as SW will most likely reach the sensor before MB.

The DOA of MB and SW for the sensor at point O are defined in the horizontal plane as azimuth values ϕ_{MB}, ϕ_{SW}, relative to the sensor's local coordinate system (x-axis in Fig. 1). Here the azimuth ϕ_{SW} is the angle of incidence of a wavefront traveling from point A, and ϕ_{MB} is the angle of incidence of a wavefront traveling from point Z.

3 Problem statement

Knowing t_{SW} and t_{MB}, gunshot acoustic localization may be performed by estimating the angle θ and the miss distance. Angle θ may be estimated by applying a shockwave acoustic model to the duration of the SW transient (Aguilar et al. 2007), or calculated under known bullet caliber assumption (Sallai et al. 2011). Then, using multiple measurements of t_{SW} and t_{MB} from K synchronous single-sensor UGS, the miss distances can be approximated and point Z located via a search procedure proposed by Sallai et al. (2011). UGS synchronization plays a crucial role in such approaches and heavily influences the bound parameters of the bounded search procedure, as well as the overall localization accuracy, as discussed by Lindgren et al. (2009). Alternatively, using multiple measurements from K asynchronous single-sensor UGS and assuming θ to be known, it is possible to iteratively estimate MB DOA, miss distances, the bullet's trajectory and, consequently, point Z via a multistage optimization procedure proposed by Damarla et al. (2010). If UGS clocks are sufficiently synchronized, a mutual reference moment t_{shot} can be established for all UGS via (5), and Z can be estimated by multilateration, using time delays t_{MB} from (4). Multilateration and its application to shooter localization is discussed further in the "Appendix".

Unfortunately, if gunshot events include NOI events, such as reflections and target hit (TH) noise, MB cannot be unambiguously selected from numerous events following SW. Consider, for example, Fig. 2, which presents six fundamental gunshot scenarios. Scenarios I–III do not contain NOI events and are most commonly considered in the majority of state of the art approaches. In Scenarios I and II the bullet either passes through or beside the UGS cluster, and no TH is detected. The localization is then performed using pure SW readings (arrows pointing from one or both sides towards the bullet's trajectory) and MB readings (arrows pointing towards the shooter's position). Scenario III assumes that only MB are detected. This makes it a trivial localization problem which can be solved using conventional localization methods, e.g., multilateration. Scenarios IV–VI, on the other hand, assume the presence of NOI events and the masking effect. Here either SW or MB may be corrupted or masked by TH (Scenario V), or either SW or MB may be corrupted or masked by each other (Scenarios IV and VI). Furthermore, NOI such as reflections and background noise may be present for all scenarios and must be accounted for accordingly. NOI events can be eliminated by identifying MB and SW by their acoustic properties (Libal and Spyra 2014) or applying statistical assignment (Osborne et al. 2014), however, these do not solve the masking problem.

The shooter localization algorithm presented in this paper assumes Scenario V of Fig. 2, where the UGS form a look-out perimeter around the potential target, that is very likely to be hit inside or near the UGS cluster. Scenario V implies that either SW or MB may be corrupted or masked by TH, and UGS situated behind the target may not detect SW altogether. As Scenario I is a special case of Scenario V (the bullet passes through the cluster and no TH is detected), the localization rules intended for Scenario V will also be applicable for Scenario I.

The paper also considers several acoustic event detection problems situated with varying shot range and influence of NOI events. At a sufficient shot range the TDOA between SW and

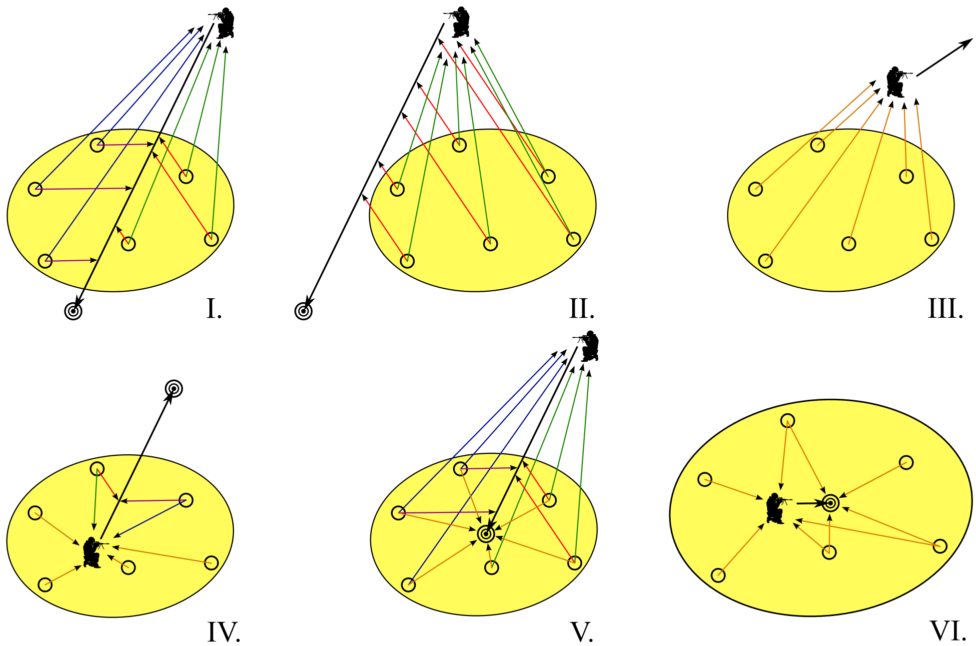


Fig. 2 Six fundamental gunshot scenarios: the bullet passes through the UGS cluster (I); the bullet passes beside the UGS cluster (II); a shot is fired away from the UGS cluster (III); a shot is fired from inside the UGS cluster (IV); the bullet hits the target in the vicinity of the UGS cluster (V); a shot is fired and the bullet reaches its target inside the UGS cluster (VI)

MB acoustic transients makes the events well distinguishable (Borzino et al. 2014). In one of our experiments we study a short range case, where event separation is not straightforward due to short TDOA. In our detection method we account for all gunshot acoustic events, as the MB transient is not guaranteed to strictly follow the SW transient.

4 Proposed approach to shooter localization

The proposed approach is intended for application in WSN with a dynamic ad-hoc topology. This implies node synchronization complications and a varying number of active nodes at any given time. Thus, we focus on an asynchronous, size-invariant solution. The WSN consists of UGS, equipped with acoustic sensor arrays, and one or several information fusion nodes. The approach consists of the following steps:

1. Each UGS detects a gunshot, separates its acoustic events, marks the time and computes a DOA value per each event.
2. Per each detected shot, each UGS sends an information packet to the fusion node, containing its position, steering angle and acoustic event parameters $\{\mathbf{x}, \beta, \mathbf{t}, \Phi\}$.
3. The fusion node performs event identification and shooter localization based on the information provided by active UGS.

The packet of UGS $k = 1, \ldots, K$ contains: UGS coordinates $\mathbf{x}_k = (x_k, y_k)$; UGS steering angle β_k; gunshot event times $\mathbf{t}_k = \left[t_1, \ldots, t_{E_k}\right]$; event DOA $\Phi_k = \left[\phi_1, \ldots, \phi_{E_k}\right]$, where E_k is the number of detected events of k-th UGS. As each UGS operates in its own coordinate system, the steering angle β_k is used to specify UGS local coordinate system steering from a global zero-rotation angle (which is defined by Earth's magnetic north). While receiving packets from UGS, the fusion node maintains a validity interval, beginning at the moment of arrival of the first packet. This way the expired packets, or the ones corresponding to another shot are dealt with separately.

For the sensor configuration we choose Uniform Circular Arrays (UCA) because they provide full horizontal FOV with a simple geometry. Each array consists of $M = 6$ microphones with an angle between two successive microphones, relative to the array center O, of

$$\alpha = \angle m_i O m_{i+1} = \frac{2\pi}{M}, \ (1 \leq i < M). \tag{7}$$

The arrays are designed to be compact, since the application field requires UGS to be covert, if hidden in the monitored environment. For the UCA experimental prototypes we use circular shells with a radius of $r = 7.5$ cm (prototype 1) and $r = 10$ cm (prototype 2).

4.1 Gunshot acoustic event detection and separation

Gunshot acoustic event detection for a general case (i.e., comprising of all scenarios of Fig. 2) is an intricate task. Amplitude-based methods are well suitable in case of Scenarios IV and VI, where both SW and MB are detected inside the UGS cluster as high-energy transients and are, therefore, distinguishable from background noise. The same holds for Scenarios I–III and V if the range is short enough for MB to be detected. Otherwise, MB can have an insufficiently high amplitude to be detected, or it can be masked by background noise. Another approach lies in identifying SW and MB by the shape of their acoustic signals. Aguilar et al. (2007) examine the N-shaped pattern of SW, and Libal and Spyra (2014) try to distinguish SW and MB from reflections by applying classification. This may work well for Scenarios I–III, where no TH or overlapping events occur and the task lies in eliminating reflections. For Scenarios IV–VI and, in our case specifically, Scenario V these methods are not guaranteed to perform well.

Shooter distance plays an important role in acoustic event separation as well. In case of a significantly short distance, acoustic event separation poses a challenge due to an extremely short TDOA between SW and MB (Freire and Apolinario 2011). Figure 3 presents an example of a normalized gunshot signal acquired 16.2 m away from the shooter. Here the TDOA between SW (at 4 ms) and MB (at 11 ms) is only 7 ms. Figure 5, on the other hand, portrays a normalized gunshot signal acquired 97.5 m away from the shooter. Here the TDOA between SW (at 25 ms) and MB (at 150 ms) is already 125 ms, which is twice as long as the whole gunshot signal of Fig. 3. If the detection algorithm treats the closely spaced events as a single event, MB may be lost in the SW transient. On the other hand, analyzing every closely spaced signal peak will waste computational resources and produce a large number of unwanted results.

Another problem lies in separating gunshot instances in case of burst-mode and automatic fire at close ranges. Consider Fig. 3, where the TDOA between SW and MB is 7 ms with post-blast events (TH and reflections) starting to occur at the 40th millisecond. Neglecting these post-blast events may seriously harm the detection process in case of burst-mode fire. For example, an AK-47 in burst mode can fire 600 rounds per minute and an M-4 fires at 950 rpm, which constitutes approximately 1 bullet every 100 ms and 63.2 ms, respectively. In this case consecutive SW and MB may be mistaken for post-blast events, and vice versa for a single shot case.

In our approach to acoustic event detection and separation we consider both short (20–40 m) and medium (100–200 m) shot distances. We establish all acoustic events by

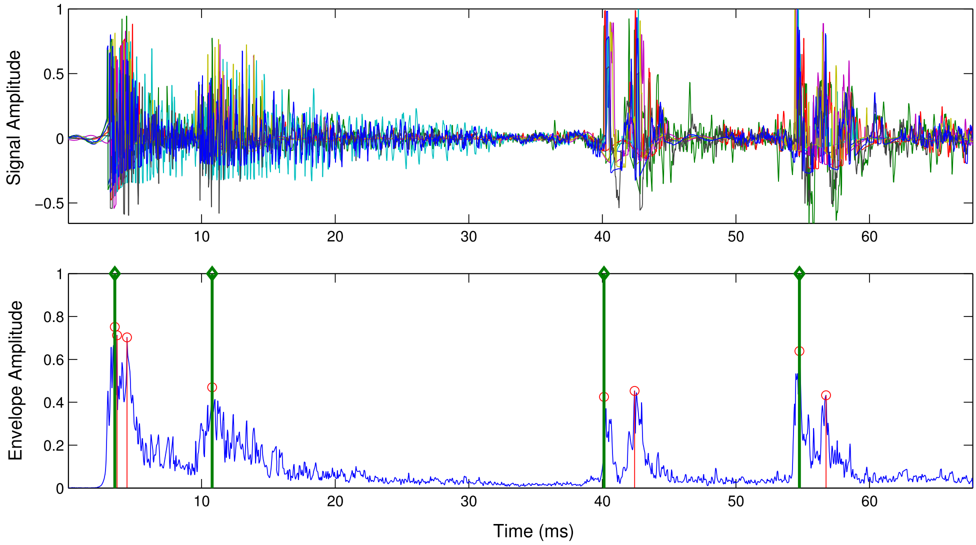


Fig. 3 Gunshot acoustic components acquired by UGS S_3, Experiment 1, at 48 kS/s (*top*). Collective envelope and times of detected events (*bottom*). *Red stems* results of peak detection; *green stems* event establishing peaks

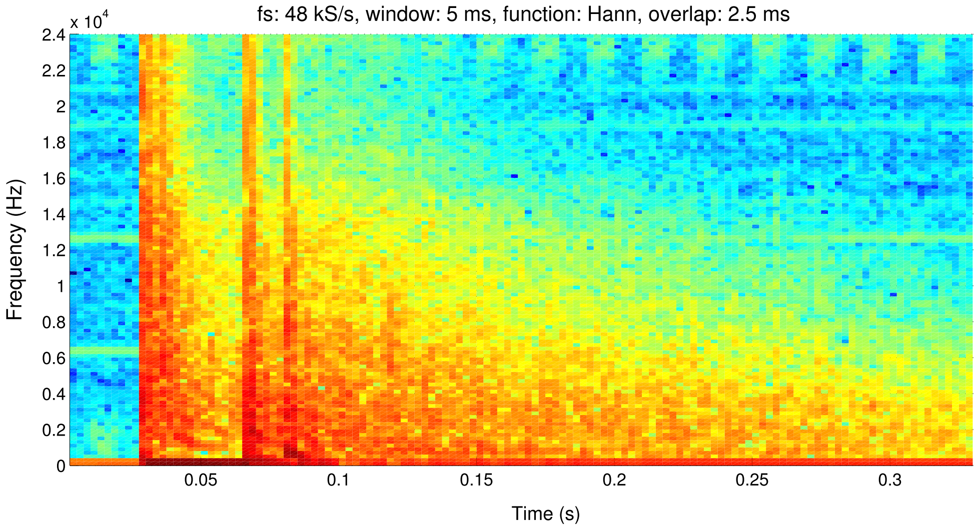


Fig. 4 Spectrogram of the gunshot signal presented in Fig. 3. Acoustic components presented in Fig. 3 are located at approximately 30–95 ms.

the following procedure. First, a collective envelope is computed using the signals from all microphones. At sampling time n, the envelope of samples $x_1[n], \ldots, x_M[n]$ is

$$s_{\text{env}}[n] = \max\left(|x_1[n]|, \ldots, |x_M[n]|\right). \tag{8}$$

Event detection is performed on the differential collective envelope

$$\Delta s_{\text{env}}[n] = s_{\text{env}}[n] - s_{\text{env}}[n-1]. \tag{9}$$

The differential envelope $\Delta s_{\text{env}}[n]$ is passed through peak detection, and peaks within an interval of $t_W/2$ seconds, where t_W is the predefined length of event window, are grouped

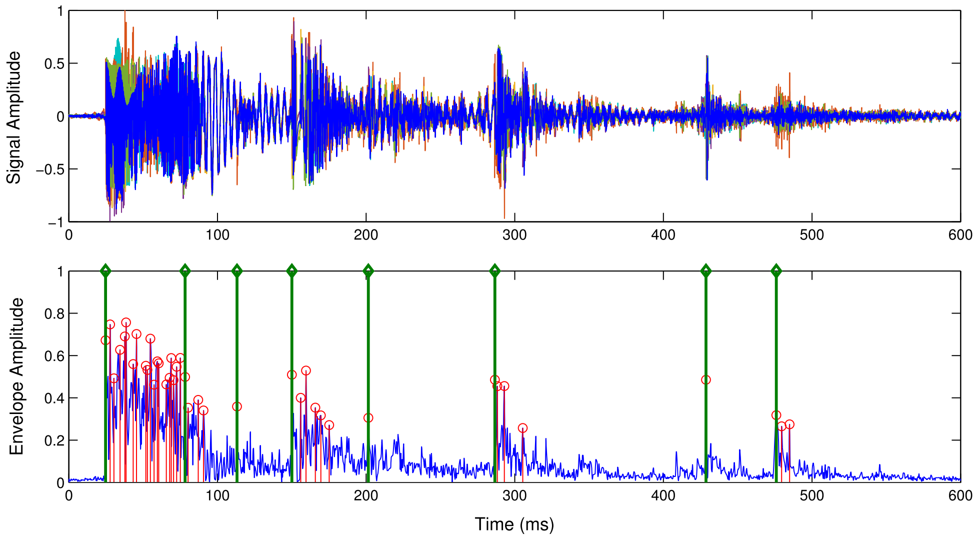


Fig. 5 Gunshot acoustic components acquired by UGS S_1, Experiment 3, shooter position 1, at 20 kS/s (*top*). Collective envelope and times of detected events (*bottom*). *Red stems* results of peak detection; *green stems* event establishing peaks

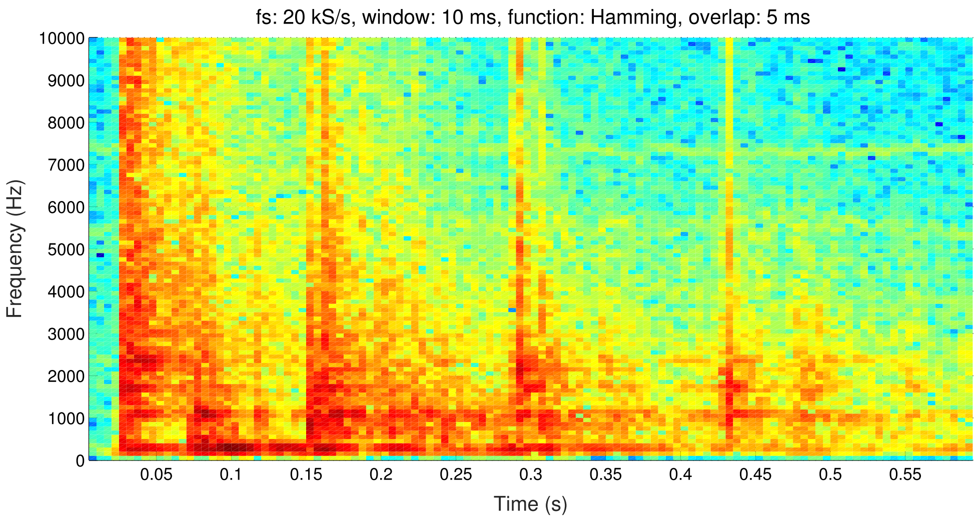


Fig. 6 Spectrogram of the gunshot signal presented in Fig. 5

together and one (the first) peak per event is chosen. An example of separation of four events is presented in Fig. 3 (lower) and of eight events—in Fig. 5 (lower). One frame of duration t_W is retrieved from the multichannel signal buffer per each event peak such, that event beginning is included in the frame and adjacent events are strictly separated. This means that if the events do not overlap, the event is windowed from the beginning of its signal's envelope rise for the duration t_W; if the events do overlap (event establishing peaks are approximately $t_W/2$ seconds apart), the first event is windowed leftward from the beginning of the second event, and the second event is windowed rightward from it's beginning.

Event identification is performed during the information fusion stage. As NOI events can also be transient in nature, they are hard to identify during event detection. Frequency analysis does not offer a straightforward solution either, as NOI events such as TH possess highly uniform spectral densities as well as SW and MB (see Figs. 4, 6). Figure 5 also portrays event overlapping at 25–110 ms. Here SW is overlapped with its own ground reflection, which results in two additional peaks being detected before MB. In this situation the identification of SW by its shape and duration will likely produce inaccurate results.

4.2 Direction of arrival estimation

At the time of shot detection, k-th UGS produces E_k multichannel signal frames of length $N = f_s t_W$, where f_s is the sampling frequency. A separate DOA estimate is then computed per each frame by applying SRP-PHAT (for reference) and our proposed lightweight method (Astapov et al. 2015a).

4.2.1 SRP-PHAT

Steered Response Power with Phase Transform is one of the most effective acoustic DOA estimation methods, proposed by DiBiase (2000). The SRP $P(\mathbf{a})$ is a real-valued functional of a spatial vector $\mathbf{a}$, the maxima of which indicate the direction to the sound source. $P(\mathbf{a})$ is computed as the cumulative Generalized Cross-Correlation with Phase Transform (GCC-PHAT) across all pairs of sensors at the theoretical time delays, associated with the chosen direction. Consider a pair of signals $x_k(t), x_l(t)$ of an array consisting of M microphones. The time instances of sound arrival from a point $a \in \mathbf{a}$ for the two microphones are $\tau(a, k)$ and $\tau(a, l)$, respectively. Hence the time delay between the signals is $\tau_{kl}(a) = \tau(a, k) - \tau(a, l)$. The SRP-PHAT for all pairs of signals is then defined as

$$P(a) = \sum_{k=1}^{M} \sum_{l=k+1}^{M} \int_{-\infty}^{\infty} \Psi_{kl} X_k(\omega) X_l^*(\omega) \mathrm{e}^{j\omega\tau_{kl}(a)} \mathrm{d}\omega, \tag{10}$$

where $X_i(\omega)$ is the spectrum (i.e., the Fourier Transform) of signal $x_i(t)$, $X_i^*(\omega)$ is the conjugate of that spectrum and Ψ_{kl} is the PHAT weight, defined as

$$\Psi_{kl} = \left(\left|X_k(\omega) X_l^*(\omega)\right|\right)^{-1}. \tag{11}$$

In a general case the spatial vector $\mathbf{a}$ partitions the FOV into a planar or volumetric discrete spatial grid. An SRP value is then computed for every point of that spatial vector. This approach requires a significant amount of computational resources and is ultimately unneeded in our planar case. To reduce the number of SRP-PHAT computations we divide the horizontal plane into n_h possible azimuth angles. A single angle increment is calculated, similarly to (7), as $\phi_h = \frac{2\pi}{n_h}$. The evaluation points are chosen in the planar FOV along a circle with a radius r_{FOV}. The SRP-PHAT evaluation is performed over the entire circumference $[0, 2\pi)$ for the points $a_{h,i} = \left(x_{h,i}, y_{h,i}\right)$:

$$\begin{aligned} x_{h,i} &= r_{FOV} \cos\left(i\phi_h\right), (0 \le i < n_h), \\ y_{h,i} &= r_{FOV} \sin\left(i\phi_h\right), (0 \le i < n_h). \end{aligned} \tag{12}$$

The azimuth is estimated in the direction of elevated SRP values $P(\mathbf{a}_h)$. For a single source case the final azimuth is equal to

$$\phi = \arg\max\left(P(\mathbf{a}_h)\right) \cdot \phi_h. \tag{13}$$

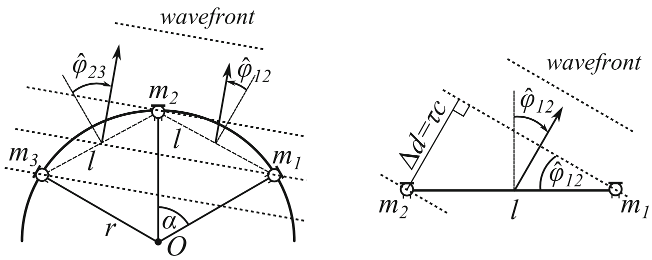


Fig. 7 Azimuth estimation in the far field for consecutive microphone pairs of the circular array (*left*). Geometry of a single microphone pair (*right*)

4.2.2 Optimized DOA estimation algorithm

Even with a reduced functional, SRP-PHAT still requires significant resources and processing time, because it performs cross-correlation between all pairs of microphones and for all specified directions. We focus on reducing the number of microphone pairs for cross-correlation and the number of discrete directions per each pair (Astapov et al. 2015a).

Our proposed method takes a directional DOA estimation approach. According to our design the microphones are embedded in a solid circular shell; therefore the DOA opposite to the common direction of any given microphone pair are not considered for analysis. The pairs of microphones for azimuth estimation are chosen such, that their inter-sensor angle is less than $\frac{\pi}{2}$: $\alpha_{ij} = \angle m_i O m_j < \frac{\pi}{2}$. The set of these pairs is

$$A = \left\{ \left(m_i, m_j\right) \subseteq S_2^M \;\middle|\; \alpha_{ij} < \frac{\pi}{2} \right\}, \tag{14}$$

where S_2^M is the set of all combinations of microphone pairs, $\left|S_2^M\right| = \binom{M}{2}$. A separate azimuth estimate $\hat{\varphi}_{ij}$ is made under the far field assumption for every pair of microphones $\left(m_i, m_j\right) \subseteq A$. For any pair $\left(m_i, m_j\right)$ of consecutive microphones (see Fig. 7), the azimuth estimate is obtained by

$$\hat{\varphi}_{ij} = \sin^{-1}\left(\frac{\tau_{ij} \cdot c}{l}\right) = \sin^{-1}\left(\frac{\Delta n_{ij}/f_s \cdot c}{l}\right), \tag{15}$$

where l is the distance between two consecutive microphones, calculated as

$$l = 2r\sin\left(\frac{\alpha}{2}\right) = 2r\sin\left(\frac{\pi}{M}\right), \tag{16}$$

and τ_{ij} is the TDOA of the wavefront to microphones m_i and m_j. For non-consecutive microphones, l is calculated by substituting α in (16) with its multiple. The TDOA is always limited to $\tau \in [-\tau_{\max}, \tau_{\max}]$, where $\tau_{\max} = l/c$ is the delay of sound traveling directly from one microphone to the other (i.e., at $\pm\frac{\pi}{2}$). In (15), τ_{ij} is also represented in terms of delay in samples Δn_{ij} and the sampling frequency f_s. To estimate Δn_{ij} we apply cross-correlation to the pair of signals:

$$R_{ij}(\Delta n) = \sum_{n=0}^{N-1} x_{m_i}[n] \cdot x_{m_j}[n - \Delta n], \;\; (i < j), \tag{17}$$

where N is the length of the signals in samples. The maximum of the cross-correlation then defines the TDOA: $\Delta n_{ij} = \arg\max\left(R_{ij}(\Delta n)\right)$. The quality of the estimate $\hat{\varphi}_{ij}$ is measured as cross-correlation peak distinctness from its mean level:

$$q_{ij} = \max\left(R_{ij}\left(\Delta n\right)\right) - \text{mean}\left(R_{ij}\left(\Delta n\right)\right). \tag{18}$$

Each estimate $\hat{\varphi}_{ij}$ is made for the middle point of the inter-microphone distance and takes the values of $\hat{\varphi}_{ij} \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$, negative if the source is situated to the left, positive—if the source is situated to the right, and zero—if it is in front of the microphone pair. Thus individual $\hat{\varphi}_{ij}$ are adjusted to the array's common angle coordinates: $\hat{\varphi}^*_{ij} = \hat{\varphi}_{ij} + ((i-1)\alpha + (j-1)\alpha)/2$. After that coherent directions are found among the estimates. This is done by applying a partitioning procedure, similar to the one we presented in Astapov et al. (2013). It performs the task of clustering the $\hat{\varphi}^*_{ij}$ estimates such, that the coherent estimates must lie within sectors with a central angle of no more than $\varphi_{\max}$. For example, if $\varphi_{\max} = \frac{\pi}{6}$, then each cluster's coherent estimates must lie no more than $\left[-\frac{\pi}{12}, \frac{\pi}{12}\right]$ from the cluster's centroid.

The resulting clusters Φ_p, $p = 1, \ldots, P$, where P is the number of clusters, each contain n_p estimates $\hat{\varphi}_k$, $k = [1, n_p]$, and the associated quality q_k. The clusters are evaluated in order to find the largest cluster, containing estimates of best quality (Astapov et al. 2015a). Algorithm 1 handles the final azimuth calculation for the single source case. The real-valued parameter $\sigma = (0, 1)$ is the threshold of tolerance and the integer parameter $n_{\min}$ is the lower bound for the largest cluster size. The final azimuth estimate ϕ cannot be made if there are insufficient coherent estimates, or if they are of low quality.

Algorithm 1 Final azimuth ϕ estimation for a single source

Require: Φ_p, q_k of every $\hat{\varphi}_k \in \Phi_p$, $p = 1, \ldots, P$
get largest cluster size $|\Phi|_{\max}$, maximum quality $q_{\max}$
if $|\Phi|_{\max} = n_{\min}$ **or** $q_{\max} <$ allowed **then**
  return $\phi \leftarrow \varnothing$ ▷ initial criteria not met
else if Φ_p of size $|\Phi|_{\max}$ contains $\hat{\varphi}_k$ with $q_{\max}$ **then**
  return $\phi \leftarrow \sum_{k=1}^{n_p} q_k \hat{\varphi}_k / \sum_{k=1}^{n_p} q_k$ ▷ weighted mean
else
  for $i = |\Phi|_{\max} - 1$ to $i > n_{\min}$ **do** ▷ search in smaller $\Phi_p, n_p > n_{\min}$
    if $\exists q_k \geq \sigma \cdot q_{\max}$ for any $\hat{\varphi}_k \in \Phi_p, \left|\Phi_p\right| = i$ **then**
      return $\phi \leftarrow \sum_{k=1}^{i} q_k \hat{\varphi}_k / \sum_{k=1}^{i} q_k$
    end if
  end for
  return $\phi \leftarrow \varnothing$ ▷ estimates of sufficient quality not found
end if

An example of final azimuth ϕ estimation based on the intermediate estimates $\hat{\varphi}^*_{ij}$ is presented in Fig. 8. Coherent directions are first established by applying the partitioning procedure with the $\varphi_{\max}$ parameter. The resulting clusters Φ_1, Φ_2 and Φ_3 contain only one azimuth value because they do not lie within a sector with the central angle less than $\varphi_{\max}$, which means that the coherency condition is not met for these azimuth values. The clusters Φ_4–Φ_6, on the other hand, do contain coherent estimates. Then, according to Algorithm 1, the largest cluster containing the estimates of the highest quality is established. Cluster Φ_6 is the largest cluster which also contains the estimates of highest quality $q_{\max}$, therefore, the final azimuth is calculated as the weighted mean of the estimates contained in this cluster. Cluster Φ_4, on the other hand, does not meet the lower bound of allowed cluster size $n_{\min}$, while cluster Φ_5 is of sufficient size, however, it does not contain estimates of sufficient

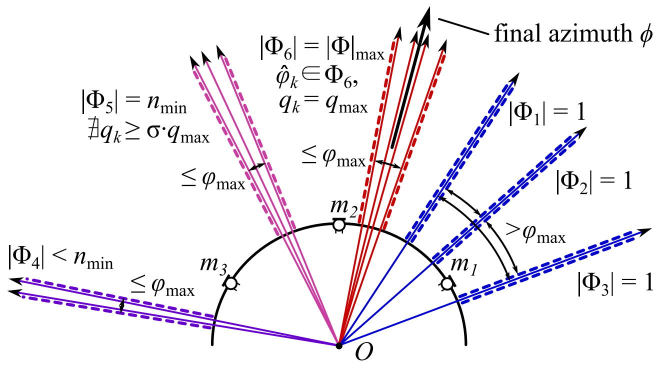


Fig. 8 A graphical example of the partitioning procedure for finding coherent directions among intermediate azimuth estimates and the estimation of the final azimuth estimate according to Algorithm 1

quality. Thus, these two clusters do not meet the criteria of Algorithm 1 and are omitted from analysis.

To determine the increased computational efficiency of our proposed method, we quantify the reduction in the number of cross-correlations required for computing SRP-PHAT and our method, as cross-correlation is the most resource-demanding operation in both methods. SRP-PHAT will calculate $n_h \cdot \binom{M}{2}$ cross-correlations; our method will calculate $\delta \cdot |A|$ cross-correlations, where $\delta = \sum \delta_{ij}$ is the total number of shifts required for calculating cross-correlations for all microphone pairs $\left(m_i, m_j\right) \subseteq A$. As the time delay τ is bounded by τ_{max} and τ is expressed in delay in samples Δn, then Δn is also bounded by a maximal sample shift: $\Delta n \in [-\Delta n_{max}, \Delta n_{max}]$, where Δn_{max} is calculated as

$$\Delta n_{max} = \left\lfloor l \cdot \frac{f_s}{c} \right\rfloor, \tag{19}$$

where $\lfloor \cdot \rfloor$ denotes rounding to the largest previous integer (i.e., the floor function). Consequently, cross-correlation $R_{ij}\,(\Delta n)$ will require $\delta_{ij} = 2\Delta n_{max}(i, j) + 1$ shifts to cover all possible TDOA values. In our experiments we set $M = 6$ and $n_h = 500$ for both UCA prototypes. The number of cross-correlations per each SRP-PHAT computation is then equal to $500 \cdot \binom{6}{2} = 500 \cdot 15 = 7500$. According to (14), in case of $M = 6$ the proposed method utilizes $|A| = 12$ pairs of microphones: 6 consecutive pairs $m_i m_{i+1}$ and 6 pairs over one microphone $m_i m_{i+2}$. Assuming $c = 340$ m/s, for prototype 1 UCA ($r = 7.5$ cm, $f_s = 48$ kS/s) the number of cross-correlations per each DOA evaluation using the proposed method is then equal to $6 \cdot 21 + 6 \cdot 43 = 384$. For prototype 2 UCA ($r = 10$ cm, $f_s = 20$ kS/s) the number of cross-correlations is equal to $6 \cdot 11 + 6 \cdot 23 = 204$. Therefore the number of resource-demanding operations is reduced by more than one order of magnitude.

4.3 Information fusion and shooter localization

As a result of shot detection, the fusion node receives K packets $\{\mathbf{x}, \beta, \mathbf{t}, \Phi\}_k, k = 1, \ldots, K$, where K is the number of active UGS, which have detected at least one gunshot event. The number of detected events E_k may vary per UGS. The DOA estimates Φ_k are first steered to the global coordinate system, $\Phi_k \leftarrow \Phi_k - \beta_k$, and information fusion is then conducted in the following steps: identification of SW and MB DOA; estimation of shot geometry; estimation of miss distance and distance to shooter for each UGS; shooter localization.

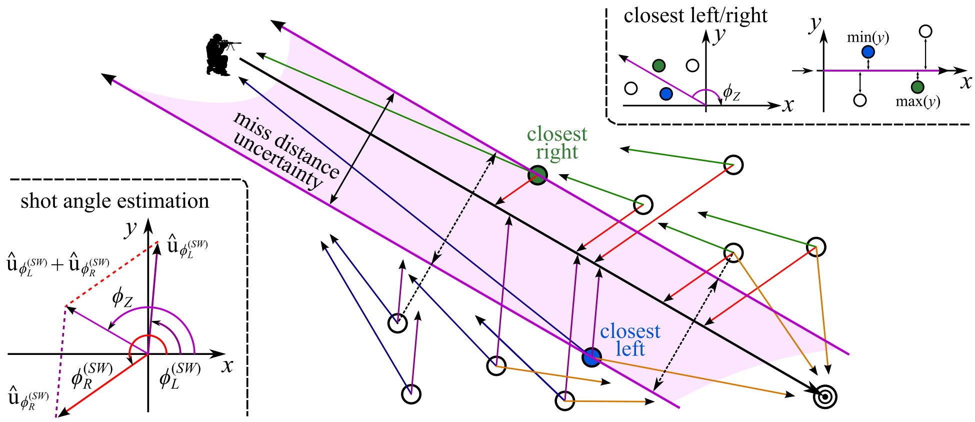


Fig. 9 Shot angle and miss distance uncertainty interval estimation by UGS groups, situated to the left and to the right from the bullet's trajectory

Information fusion is performed on multiple fusion nodes which can govern a single UGS group or several either intersecting or separate groups. Furthermore, each UGS may be permitted to act as a fusion node if its computational resources allow for it. As a result, several position estimates may be produced for the same shot instance. This paper does not concern the further steps at higher levels of data fusion, where these various estimates are analyzed. This section presents the solution for shooter localization performed on a single fusion node.

4.3.1 DOA coherency

Consistent DOA are established by analyzing all $\Phi = \{\Phi_k \mid k = 1, \ldots, K\}$ estimates. To locate coherent estimates, the angular values in Φ are clustered in a manner, similar to the one described in Sect. 4.2.2. If coherent estimates exist, we obtain P clusters Φ_p, $p = 1, \ldots, P$, each containing n_p estimates ϕ_i, $i = [1, n_p]$.

Assuming Scenario V (Fig. 2), Φ_p will contain SW DOA corresponding to the detected SW of UGS situated to the left and to the right from the bullet's trajectory, MB DOA, and other readings, like DOA of TH, various reflections and noise. The DOA of SW vary only slightly (due to DOA estimation error and natural variation of angle θ) and do not depend on the distance to shooter; MB DOA, on the other hand, depend on the distance to shooter and UGS cluster dimensions. If the distance to shooter is significantly larger than the width of the UGS cluster, MB DOA will be roughly parallel for all UGS. At a closer distance the UGS situated on the opposite sides of the bullet's trajectory will have their MB DOA significantly skewed towards the trajectory in the shooter's direction. A principle diagram of coherent DOA for Scenario V is presented in Fig. 9.

4.3.2 Event identification and shot geometry estimation

To reduce the error of individual DOA estimates, event identification is performed on the mean values of clusters Φ_p: $\bar{\phi}_p = \frac{1}{n_p} \sum \Phi_p$, $p = 1, \ldots, P$. To identify SW DOA, all $\bar{\phi}_p$ are analyzed pairwise. For each pair $\bar{\phi}_i, \bar{\phi}_j$, $i = [1, P-1]$, $j = [i+1, P]$, a central angle ϕ_Σ is first calculated as the angular component of the sum of their corresponding unit vectors $\hat{\mathbf{u}}_{\bar{\phi}_i} + \hat{\mathbf{u}}_{\bar{\phi}_j}$ (see Fig. 9). SW DOA are then identified under the assumptions that SW events are detected first, and at least one SW DOA was detected to the left and to the right from the bullet's trajectory. Thus $\bar{\phi}_p$ are searched for such $\bar{\phi}_i$, $\bar{\phi}_j$, that meet all the following conditions:

$$\begin{gathered}\frac{\pi}{2} - \varphi_{\max}^{(SW)} < \left|\phi_\Sigma - \bar{\phi}_i\right| < \frac{\pi}{2} - \varphi_{\min}^{(SW)}, \\ \frac{\pi}{2} - \varphi_{\max}^{(SW)} < \left|\phi_\Sigma - \bar{\phi}_j\right| < \frac{\pi}{2} - \varphi_{\min}^{(SW)}, \\ \forall_{\mathrm{t}_k} \mathrm{ind}\left(t_{\phi_k} \mid \phi_k \in \Phi_i\right) = 1,\ \forall_{\mathrm{t}_k} \mathrm{ind}\left(t_{\phi_k} \mid \phi_k \in \Phi_j\right) = 1.\end{gathered} \tag{20}$$

We define ind as the operation that determines the index of a specific element in a vector of values. $\left(\varphi_{\min}^{(SW)}, \varphi_{\max}^{(SW)}\right)$ is the interval of SW propagation angle θ (see Sect. 2) expected values, accounting for variance and measurement error. For example, if $\theta \approx 25°$ and $\pm 5°$ measurement deviation are expected, this interval is set to $\left(\frac{\pi}{9}, \frac{\pi}{6}\right)$. If the conditions are met, $\bar{\phi}_i$, $\bar{\phi}_j$ and $\phi_k \in \Phi_i \cup \Phi_j$ are labeled $\bar{\phi}_i^{(SW)}$, $\bar{\phi}_j^{(SW)}$ and $\phi_k^{(SW)}$, respectively. For $\bar{\phi}_i^{(SW)}$, $\bar{\phi}_j^{(SW)}$, condition (20) also implies that they were measured on the opposite sides of the bullet's trajectory. Consequently, we adopt their central angle ϕ_Σ as the shot angle ϕ_Z estimate (i.e., the angle, at which the bullet travels towards the UGS cluster; see Fig. 9).

Having estimated ϕ_Z, the UGS S_k that have detected SW are placed either into the "left", or "right" groups G_L, G_R:

$$\begin{aligned}\phi_k^{(SW)} &< \phi_Z \Rightarrow S_k \in G_L, \\ \phi_k^{(SW)} &> \phi_Z \Rightarrow S_k \in G_R.\end{aligned} \tag{21}$$

To estimate the miss distance, $S_k \in G_L \cup G_R$ closest to the bullet's trajectory are first located. This is done by steering the S_k coordinates $\mathbf{x}_k$ by ϕ_Z towards the x-axis around the UGS common spatial centroid $\bar{\mathbf{x}} = \frac{1}{K}\sum \mathbf{x}_k$ as

$$\begin{pmatrix} x_k' \\ y_k' \end{pmatrix} = \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix} + \begin{pmatrix} \cos(\phi_Z) & \sin(\phi_Z) \\ -\sin(\phi_Z) & \cos(\phi_Z) \end{pmatrix} \begin{pmatrix} x_k - \bar{x} \\ y_k - \bar{y} \end{pmatrix}. \tag{22}$$

Then, as portrayed in Fig. 9, "closest left" and "closest right" UGS $\breve{S}_L$, $\breve{S}_R$ are defined as

$$\begin{aligned}\breve{S}_L &= S_i,\ i = \mathrm{ind}\min\left(y_k'\right),\ S_k \in G_L, \\ \breve{S}_R &= S_j,\ j = \mathrm{ind}\max\left(y_k'\right),\ S_k \in G_R,\end{aligned} \tag{23}$$

and the distance between them, perpendicular to the shot angle, $\phi_Z - \frac{\pi}{2}$, is referred to as the miss distance uncertainty interval. Inside this interval the exact miss distance cannot yet be estimated at this point. We approximate it at a later stage of shooter localization.

To identify the DOA corresponding to MB events, $\bar{\phi}_p$ are searched for such $\bar{\phi}_i$, $i = [1, P]$, that meet the following condition:

$$\left|\phi_Z - \bar{\phi}_i\right| < \varphi_{\max}^{(MB)},\ \bar{\phi}_i \neq \bar{\phi}_i^{(SW)}. \tag{24}$$

During MB DOA identification preference is given to $S_k \in G_L \cup G_R$, because SW detection implies that the bullet has passed the UGS, and thus TH will likely not come from the same direction as MB. This way TH DOA will most certainly be avoided. NOI events caused by different noise, on the other hand, are seldom acquired with consistent DOA by a significant number of UGS, and thus their corresponding clusters Φ_p are significantly smaller and the estimates more dispersed. At this stage they are easily separable from the estimates considered for the MB label. Incidental acoustic sources arising in the FOV can be identified and excluded from analysis by general acoustic monitoring and source tracking techniques, e.g., as in Astapov et al. (2013). As a result of MB DOA identification, $\phi_k \in \Phi_i$ meeting condition (24) are labeled $\phi_k^{(MB)}$.

4.3.3 Distance to shooter estimation and shooter localization

Having identified $\phi_k^{(SW)}$ and $\phi_k^{(MB)}$, $k = 1, \ldots, K$, where K is now the number of UGS with both detected events, it is possible to accurately compute the TDOA between MB and SW, Δt_k as

$$\begin{aligned} \Delta t_k &= t_{k,i} - t_{k,j}, \\ i &= \operatorname*{ind}_{\Phi_k}\left(\phi_k^{(MB)}\right),\ j = \operatorname*{ind}_{\Phi_k}\left(\phi_k^{(SW)}\right). \end{aligned} \tag{25}$$

Based on Δt_k and the k-th UGS miss distance estimate $\hat{d}_{miss}^{(k)}$, it is possible to assess the distance to shooter from the k-th UGS using a closed form solution, proposed by Sallai et al. (2011):

$$d_{S_k,Z} = \frac{1}{2\left(c^4 - v^4\right)}\left(A - 2\sqrt{B}\right), \tag{26}$$

where

$$\begin{aligned} A &= -2v^3\hat{d}_{miss}^{(k)}\sqrt{v^2+c^2} - 2\Delta t_k c^3 v^2 + 2c^2\hat{d}_{miss}^{(k)} v\sqrt{v^2+c^2} - 2\Delta t_k c v^4, \\ B &= -2c^4v^4\left(\hat{d}_{miss}^{(k)}\right)^2 + 2\left(\Delta t_k\right)^2 c^6 v^4 \\ &\quad +2\left(\Delta t_k\right)^2 c^4 v^6 - 2c^7\hat{d}_{miss}^{(k)}\Delta t_k v\sqrt{v^2+c^2} + c^8\left(\Delta t_k\right)^2 v^2 \\ &\quad +2c^8\left(\hat{d}_{miss}^{(k)}\right)^2 + 2v^5\hat{d}_{miss}^{(k)}\sqrt{v^2+c^2}\Delta t_k c^3. \end{aligned}$$

Projectile velocity can be empirically estimated by inverting equation (1) as $\hat{v} = c/sin\left(\hat{\theta}\right)$ and applying it to $\hat{\theta}$, which is computed as $\hat{\theta} = \bar{\phi}_L^{(SW)} - (\pi - \phi_Z)$, where $\bar{\phi}_L^{(SW)}$ is the mean value of the set of estimates, labeled as SW and belonging to the left group. For $\hat{d}_{miss}^{(k)}$ estimation, a minimal and maximal miss distance interval $\left[d_{\min}^{(k)}, d_{\max}^{(k)}\right]$ is first established. For every S_k, its minimal miss distance $d_{\min}^{(k)}$ spans from its coordinates $\mathbf{x}_k$ in the direction towards the bullet's trajectory (perpendicularly to ϕ_Z) up to the point, where miss distance ambiguity starts; the maximal distance $d_{\max}^{(k)}$ spans further, up to the point, where miss distance ambiguity ends (see dashed line spanning from UGS of the right group in Fig. 9).

Equation (26) suggests that $d_{S_k,Z}$ rises with $\hat{d}_{miss}^{(k)}$, therefore, $S_k \in G_L$ will give larger, and $S_k \in G_R$—smaller estimates if $\hat{d}_{miss}^{(k)}$ is at the ambiguity start of group G_R, and vice versa if it is at the ambiguity start of G_L. So, the ambiguity interval is iteratively passed from $d_{\min}^{(k)}$ to $d_{\max}^{(k)}$ with a step of d_{step}, the miss distances for K UGS are estimated as $\hat{d}_{miss}^{(k)} = d_{\min}^{(k)} + i \cdot d_{step}$, and distance estimates to shooter $\hat{d}_{S_k,Z}(i)$ at each step are obtained using (26). A shooter position estimate $\hat{Z}_k(i)$ is computed per each UGS, using $\mathbf{x}_k$, $\phi_k^{(MB)}$ and $\hat{d}_{S_k,Z}(i)$. The fitness of $\hat{Z}_k(i)$ point estimates is measured by their average distance from their common centroid $\bar{Z}(i)$:

$$f_{fit}(i) = \frac{1}{K}\sum_{k=1}^{K}\left\|\bar{Z}(i) - \hat{Z}_k(i)\right\|. \tag{27}$$

The minimum of the fitness function f_{fit} indicates the miss distance estimates, closest to the actual value, $\hat{d}_{miss}^{(k)} \simeq d_{miss}^{(k)}$, and the final shooter's position estimate is selected as $\hat{Z} = \bar{Z}(i)$, where $i = \arg\min\left(f_{fit}(i)\right)$.

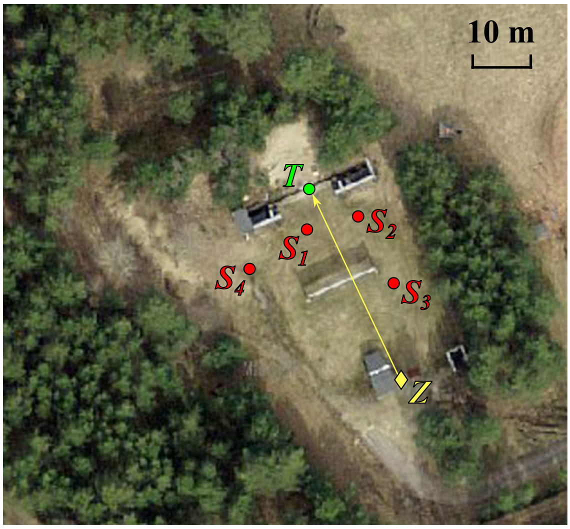


Fig. 10 Layout of Experiment 1. T–target position; Z–shooter position; S_k–UGS positions

5 Experimental results

The proposed shooter localization approach is tested on real gunshot signals, acquired during three separate live experiments at two different outdoor shooting ranges. Experiment 1 was performed at a small shooting range with the shooter-target distance of 35 m. The shooter took one position for the entire experiment. The signals were acquired by 4 UGS. The layout of Experiment 1 is presented in Fig. 10. Experiments 2 and 3 were performed at a larger shooting range with the shooter-target distance of 100 m (from the central shooting position). The shooter took three firing positions during both experiments. The signals were acquired by 6 UGS. In Experiment 2 the UGS were placed in a tight hexagon-shaped cluster, equidistantly positioned 5 m away from the cluster's center. The layout of Experiment 2 is presented in Fig. 11 (left). In Experiment 3, on the other hand, the UGS were distributed more spaciously. The layout of Experiment 3 is presented in Fig. 11 (right). The firearm used in all three experiments was the Husqvarna 8x57JS rifle with the cartridge muzzle velocity equal to $v_0 = 780$ m/s, thus the shockwave is expected to spread approximately at $\theta \simeq 25.8°$ relative to the bullet's trajectory.

UGS latitude/longitude coordinates were measured using a standalone GPS device (Trimble R8 GNSS) since none of the UGS prototypes have GPS locators on board. For data analysis we convert the GPS coordinates into a local planar coordinate system with the target being set as the zeroth coordinate. The steering angle β_k for each UGS is defined as the heading, measured with a high-precision compass. The presented experimental results are already brought to zero steering and the influence of β_k measurement error is not discussed.

Experiment 1 was conducted at a shooting range surrounded by scattered trees. A bullet-catching sand mound is situated approximately 5 m behind the target. The shooter's position is situated beside a small concrete safety bunker, which obstructed direct line of sight of UGS S_4. An overhead horizontal barrier is situated in the middle of the shooting range. The shooter fired 30 shots from a standing position; as the target and all UGS were raised by approximately 1 meter from the ground, each bullet passed the cluster at UGS level or slightly higher. Layout coordinates in meters are presented in Table 1. Weather conditions were the following: temperature $t° \simeq 2$ °C, cloudiness 10%, no precipitation, wind speed 1–2 m/s. Parameters for all steps of the localization process are presented in Table 2.

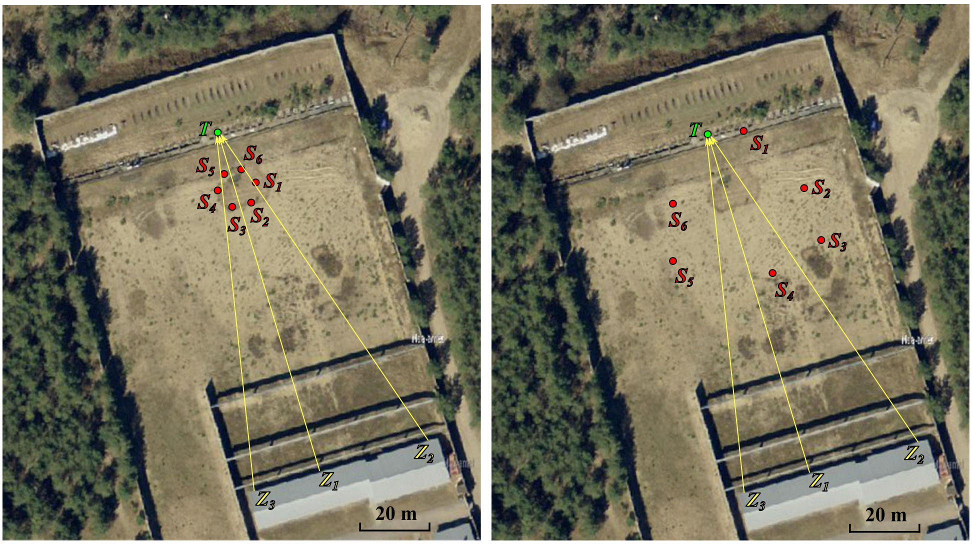


Fig. 11 Layout of Experiments 2 (*left*) and 3 (*right*). T–target position; Z_i–shooter positions; S_k–UGS positions

Table 1 Target $\mathbf{x}_T$, firing point $\mathbf{x}_{Z_i}$ and UGS $\mathbf{x}_{S_k}$ coordinates in meters

Type	Experiment 1	Experiment 2	Experiment 3
$\mathbf{x}_T$	(0, 0)	(0, 0)	(0, 0)
$\mathbf{x}_{Z_1}$	(0, 35)	(0, 100)	(0, 100)
$\mathbf{x}_{Z_2}$	–	(−28.5, 100)	(−28.5, 100)
$\mathbf{x}_{Z_3}$	–	(20, 100)	(20, 100)
$\mathbf{x}_{S_1}$	(4, 6)	(−5, 16)	(−10, 3)
$\mathbf{x}_{S_2}$	(−5.5, 7)	(−2.5, 20.3)	(−20, 20)
$\mathbf{x}_{S_3}$	(−6, 20)	(2.5, 20.3)	(−20, 35)
$\mathbf{x}_{S_4}$	(14, 7.5)	(5, 16)	(−5, 40)
$\mathbf{x}_{S_5}$	–	(2.5, 11.7)	(20, 30)
$\mathbf{x}_{S_6}$	–	(−2.5, 11.7)	(15, 15)

Table 2 Shot detection, DOA estimation and shooter localization parameters

Parameter	Unit	Experiment 1	Experiment 2	Experiment 3
f_s	kS/s	48	20	20
t_W	ms	10	20	20
n_h	–	500	500	500
r_{FOV}	m	0.5	0.5	0.5
$\sigma, n_{\min}$	–	0.8, 3	0.8, 3	0.8, 3
$\left(\varphi_{\min}^{(SW)}, \varphi_{\max}^{(SW)}\right)$	deg.	(21, 31)	(21, 31)	(21, 31)
$\varphi_{\max}^{(MB)}$	deg.	60	40	40
d_{step}	m	0.5	0.5	0.5

Fig. 12 View of the shooting range from the shooter's position, 100 m away from the target (*top*). UGS placement for Experiment 2 (*bottom*). The span of the bottom image is highlighted on the top image with a *red rectangle*

Experiment 2 was conducted at a shooting range, which is entirely fenced by tall concrete walls. A bullet-catching sand mound is situated approximately 15–20 m behind the target. The firing points are situated just outside the shooting range hall. Three overhead horizontal barriers are placed along the first 25 m of the range (see Fig. 12 top). The shooter fired 6 shots from each of the three firing points from a standing position. As the target is elevated from the ground level by 3 m, but all UGS were raised by slightly more than 1 meter from the ground, the bullets traveled above the UGS cluster (see Fig. 12 bottom). Layout coordinates in meters are presented in Table 1. Weather conditions were the following: temperature $t^\circ \simeq 8$ °C, cloudiness 50%, no precipitation, wind speed 5–10 m/s. Parameters for all steps of the localization process are presented in Table 2.

Experiment 3 was conducted at the same shooting range as Experiment 2. The same firing points and target position were used. The shooter fired 6 shots from points 1 and 2, and 7 shots from point 3 from a standing position. The UGS are more widely distributed; UGS S_1 is placed at the target's elevation level, as portrayed in Fig. 13. Layout coordinates in meters are presented in Table 1. Weather conditions were the following: temperature $t^\circ \simeq 6$ ∘C, cloudiness 100%, light rain, wind speed 9–12 m/s with gusts up to 20 m/s. Parameters for all steps of the localization process are presented in Table 2.

Fig. 13 UGS placement for Experiment 3. Shooting range view is presented in Fig. 12

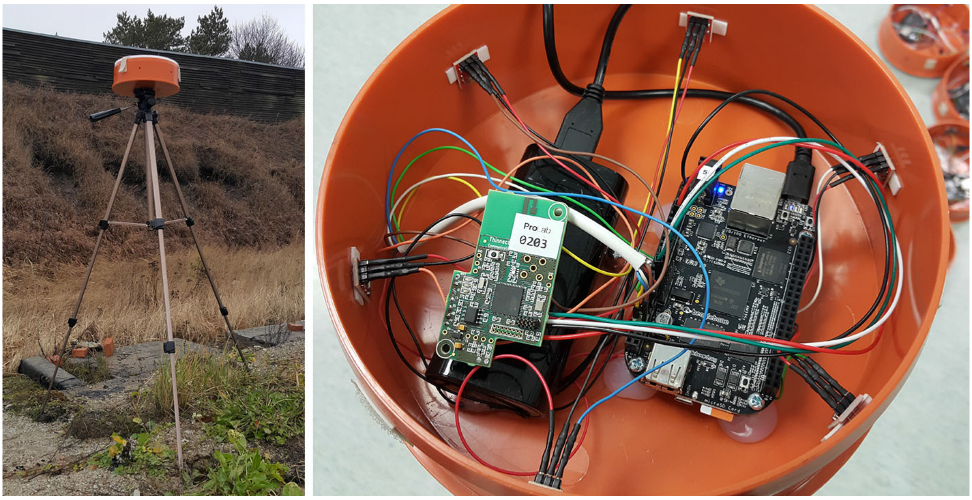


Fig. 14 UGS prototype 2 (*left*). Prototype inner components (*right*)

5.1 Prototype implementation

For the UGS implementation we use Uniform Circular Arrays with $M = 6$ microphones. Two prototypes were created during the course of development. Prototype 1 UGS are used in Experiment 1. Prototype 2 UGS are used in Experiments 2 and 3.

Prototype 1 is composed of a plastic circular shell with the radius of $r = 7.5$ cm, Vansonic PVM-6052 condenser microphones, a multichannel signal amplification circuit and an Agilent U2354A DAQ, connected to a PC running MATLAB. The signals are acquired using the MATLAB Data Acquisition Toolbox at the sampling frequency of $f_s = 48$ kS/s per channel and processed offline. Prototype 1 UGS operate independently from one another, and only rough synchronization is achieved by scheduling the starting moment of data acquisition on each PC. No inter-UGS communication is performed. This cumbersome design is improved upon in prototype 2.

Prototype 2 is composed of an enclosed plastic circular shell with the radius of $r = 10$ cm, ADMP401 MEMS microphones (Pololu Corp., USA), a BeagleBone Black (BBB) development board, a power bank, and a proprietary stand-alone communication module, we call MURP module (see Fig. 14). BBB features two programmable real-time units (PRU) with 32-bit RISC processors, and also an 8-channel 12-bit Analogue Digital Converter (ADC).

This enables the BBB to be used as both a DAQ and processing unit, sampling the data from 6 channels at $f_s = 20$ kS/s separately from the BBB non-real time operating system. The samples produced by PRU are written into a circular memory buffer implemented by the PRUIO library. A circular buffer is used in order to guarantee continuous online signal processing. The binary raw data is also stored on an external SD memory card for later analysis. The sampled data is then fed frame by frame to other software modules, which perform gunshot event detection and DOA estimation. The MURP module (the circuit board found on top of the power bank in Fig. 14) has its own Atmel Atmega256RFR2 chip and IEEE 802.15.4 compliant radio transceiver. A synchronized start time is achieved by broadcasting a sequence of specially timed messages from a control node (six messages counting down from 100 ms with 20 ms intervals), which are used to trigger the concurrent start of signal sampling within the sensor cluster.

The fusion node is implemented on an embedded platform equipped with an Atmel ATmega128RFA1 microcontroller and a IEEE 802.15.4 compliant radio transceiver. In Experiments 2 and 3 the fusion node is used for starting concurrent signal sampling on all the UGS, and no actual data transfer is performed during the experiments, as this paper does not consider the problems of WSN communication. The questions of data validation and network management by a middleware component are discussed in Preden et al. (2013).

5.2 Results of Experiment 1

An example of gunshot event detection by UGS S_3 was presented in Fig. 3. Results show that the applied detection procedure succeeds in detecting gunshot events even with a significantly short TDOA between SW and MB events. During the experiment all 30 shots were detected by all UGS, however, UGS S_4 failed to provide the DOA of seven MB events. Close analysis of signals acquired by UGS S_4 shows that the number of detected events was equal to the number of signal envelope rises per shot. Since the direct line of sight from the shooter to UGS S_4 was obstructed by the safety bunker, the intermediate azimuth estimates did not have sufficient quality to pass the criteria of Algorithm 1 and no final estimates were made. Other UGS detected both SW and MB for every shot; TH was detected in the majority of cases. There were also 13 cases of detection of TH before MB by UGS S_1 and S_2, the reason being their close position to the target. These results clearly indicate the need of gunshot event identification prior to shooter localization.

The two considered DOA estimation methods succeed in establishing a single distinct direction in the majority of cases. A visualization of DOA estimation intermediate results for UGS S_1 is presented in Fig. 15. SRP-PHAT values for every discrete point are scaled to the maximal value of 0.2; the individual pair-vise estimates of the proposed method are ordered by their cross-correlation peak distinctness from the least to the most sharp and depicted as black, blue, green and red lines, respectively; the thick black line denotes the final estimate. It can be seen that both methods produce one distinct beam and several lesser beams, corresponding to DOA of NOI events. The subplots corresponding to SW detection both show a minor beam in the MB direction. This evidently happens due to short TDOA between the two events and their partial overlapping. The MB itself is very evident in the central pair of subplots. Figure 15 clearly shows that the proposed methods produces results highly similar to the ones of SRP-PHAT.

The DOA estimates of four consecutive shots computed by SRP-PHAT are presented in Fig. 16a, and by the proposed method in Fig. 16b (several estimate values are equal and overlap). It can be seen that SRP-PHAT estimates are more dispersed for UGS S_2 and S_3. SW, MB and TH events are well distinguishable for both methods, however, results for UGS

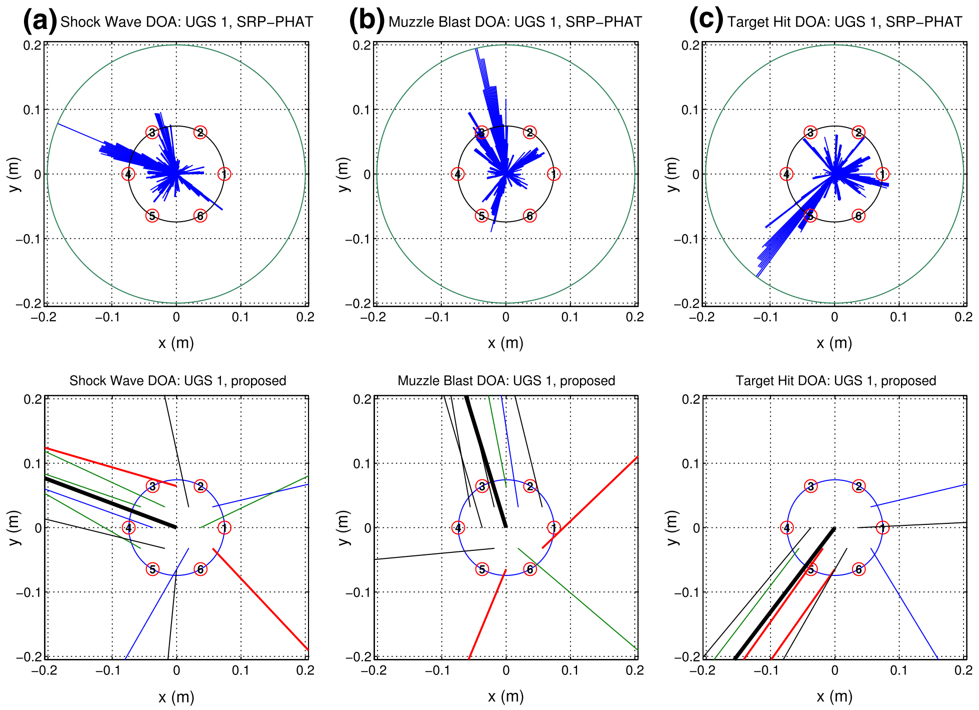


Fig. 15 DOA estimation intermediate results of Experiment 1, UGS S_1. Top subplots–estimation using SRP-PHAT (*blue lines* SRP values of points defined in (12), length normalized by the radius of the green circle). *Bottom subplots* estimation using the proposed method (*black, blue, green, red lines* estimates of microphone pairs defined in (14), with estimate quality (18) increasing by color, respectively; *thick black* final estimate)

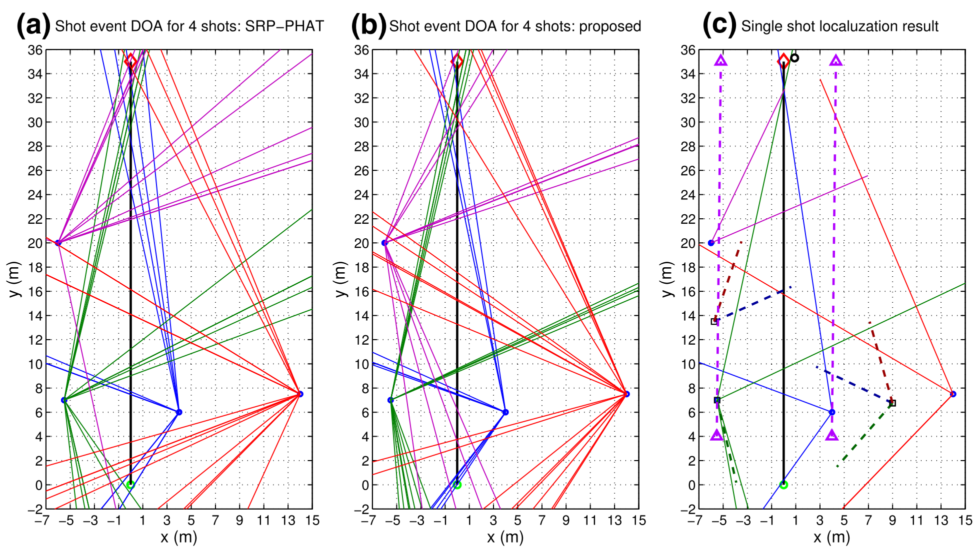


Fig. 16 Experiment 1 DOA estimates for four consecutive shots using **a** SRP-PHAT and **b** the proposed method (*red diamond* shooter true position; *green circle* target; *blue dots* UGS positions; *blue, green, purple, red lines* DOA estimates of UGS S_1–S_4, respectively). **c** Localization result for a single shot (*red*, *blue* and *green dotted* $\bar{\phi}^{(SW)}$, $\bar{\phi}^{(MB)}$ and NOI event DOA of clusters Φ_p; *purple dotted arrow* ϕ_Z and miss distance uncertainty; *black circle* final estimated shooter position)

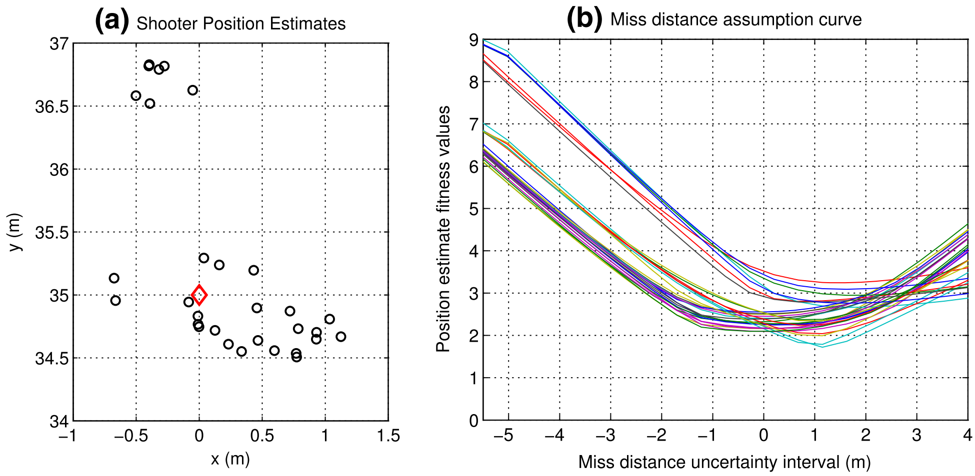


Fig. 17 Experiment 1 results for 30 shots. **a** Estimated shooter positions (*red diamond* shooter true position). **b** Values of f_{fit}, defined in (27), for the miss distance uncertainty interval

Table 3 Shooter position estimate mean error (ME) and standard deviation (SD) in meters

DOA Method	Parameter	Experiment 1	Experiment 2	Experiment 3
SRP-PHAT	$\hat{Z}$ ME	1.12	6.65	8.92
	$\hat{Z}$ SD	0.73	3.53	6.80
Proposed	$\hat{Z}$ ME	0.87	7.08	7.32
	$\hat{Z}$ SD	0.56	3.86	6.15

S_4 are significantly worse due to its larger miss distance and the obstructed line of sight to the shooter.

The intermediate results of localization and the final shooter location estimate for a single shot are presented in Fig. 16c. UGS $\{S_2, S_3\}$ and $\{S_1, S_4\}$, as expected, form clusters of consistent DOA estimates and group into G_L and G_R, respectively. Mean estimates of clustered DOA values are presented in Fig. 16c as dotted lines starting from the spatial centroids of these clusters. The shot angle $\phi_Z \simeq 90°$ is estimated with high accuracy; $\check{S}_L = S_2$, $\check{S}_R = S_1$ are correctly assigned, and thus the miss distance uncertainty interval is properly computed.

Final shooter position estimates (using the proposed method for DOA) are presented in Fig. 17a. To quantify the localization accuracy we use the mean error (ME) metric, calculated as the average Euclidean distance between the known and estimated shooter positions:

$$\mathrm{ME} = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\left(x_Z(i) - x_{\hat{Z}}(i) \right)^2 + \left(y_Z(i) - y_{\hat{Z}}(i) \right)^2 \right)^{1/2}, \tag{28}$$

where N_s is the total number of shots. ME along with its Standard Deviation (SD) for 30 shots is presented in Table 3. It can be seen that using the proposed DOA method results in a slightly smaller ME. Generally, the localization quality for both DOA estimation methods is notably high for Experiment 1. In Fig. 17a a congestion of remote points in the top left corner results from the misdetection of several MB by UGS S_4. Instantaneous bullet velocity estimation (see Sect. 4.3.3) resulted in $\hat{v} \simeq 740$ m/s, which is consistent with the cartridge

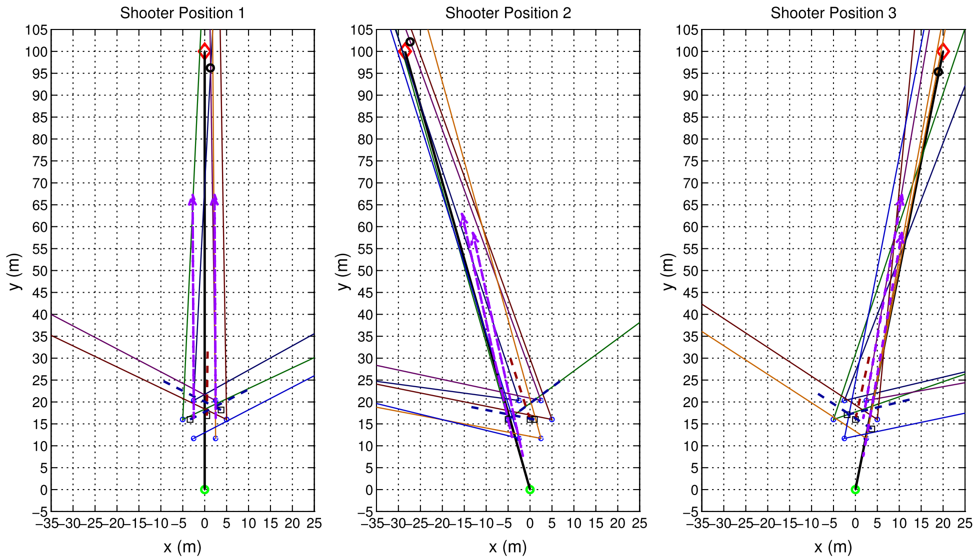


Fig. 18 Experiment 2 localization results for one shot per shooter position (*red* and *blue dotted*—$\bar{\phi}^{(SW)}$, $\bar{\phi}^{(MB)}$ of clusters Φ_p; *purple dotted arrow* ϕ_Z and miss distance uncertainty; *black circle* final estimated shooter position)

specification parameters (i.e., velocity of 753 m/s for ranges under 50 m). The values of the fitness function f_{fit} are presented in Fig. 17b. The function's minimum is situated at ± 1 m from the actual miss distance, and one global minimum of f_{fit} exists for every shot. Thus, miss distance estimation in this case can be performed by a gradient descent method rather than by iterative search.

5.3 Results of Experiment 2

The gunshot acoustic component detection procedure on each UGS succeeded in detecting every shot instance with 5–6 acoustic events per shot on average, occasionally reaching 8–9 events. Acoustic events of Experiment 2 are very similar to the ones of Experiment 3, an example of a single shot signal of which was presented in Fig. 5. The large number of NOI events is caused by numerous reflections of SW, MB, as well as TH off the concrete walls surrounding the shooting range (see Fig. 11). An elevated bullet trajectory, as explained in Section 4.1, causes ground reflections of SW and, consequently, its signal pattern resembles a transient combined with several weaker disturbances. This results in MB being detected as the 3rd or 4th event peak for every shot instance.

The intermediate results of localization and the final shooter location estimate for a single shot case from each of the three firing points are presented in Fig. 18. NOI event DOA are removed from the plots for presentation clarity. For firing point Z_1 all UGS form a single cluster of MB DOA, and UGS $\{S_1, S_2, S_6\}$ and $\{S_3, S_4, S_5\}$ form clusters of SW DOA, detected to the left and right of the bullet's trajectory and group into G_L and G_R, respectively. For firing points Z_2 and Z_3 MB DOA clusters are also formed from all UGS, because the cluster dimensions are significantly smaller compared to the distance between the cluster and the shooter positions, which results in MB DOA being roughly equal. The clusters of coherent SW DOA estimates are formed for Z_2 from UGS $\{S_1\}$ in the left group and $\{S_2, S_3, S_4, S_5, S_6\}$—in the right group. For point Z_3 the left group consists of UGS

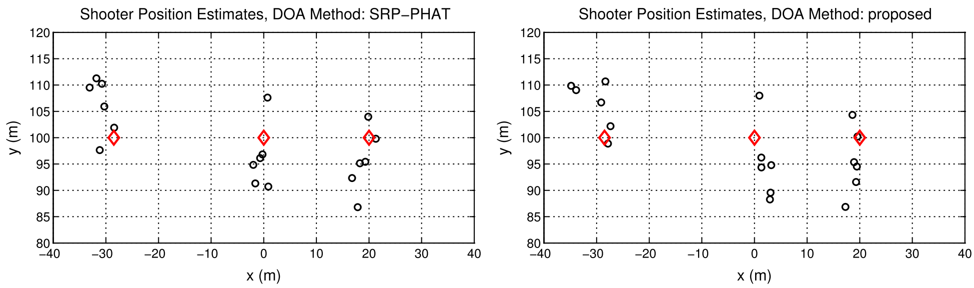


Fig. 19 Experiment 2 localization results for 18 shots with SRP-PHAT (*left*) and the proposed method (*right*) used for DOA estimation. *Black circles* estimated shooter positions; *red diamonds* true shooter positions

$\{S_1, S_2, S_3, S_6\}$ and the right group—of UGS $\{S_4, S_5\}$. As UGS S_1, S_6 and UGS S_3, S_5 are situated nearly along the bullet's trajectory for points Z_2 and Z_3, respectively, their belonging to either the left or right group changes from shot to shot. This does not influence the overall localization accuracy, as the considered UGS cluster is dense enough not to drastically change the miss distance ambiguity interval. The shot angles $\phi_{Z_1} \simeq 90°$, $\phi_{Z_2} \simeq 106°$ and $\phi_{Z_3} \simeq 79°$ are estimated with high accuracy.

Final shooter position estimates for all three firing points are presented in Fig. 19. It can be seen that the estimates are significantly more scattered, when compared to the estimates of Experiment 1. Table 3 shows that the ME for Experiment 2 is approximately 7 m, which is notably higher than a ME of approximately 1 m of Experiment 1. However, taking into consideration that the range set for Experiment 2 is almost three times larger, and prototype 2 UGS use an inferior ADC at $f_s = 20$ kS/s, compared to a standalone DAQ of prototype 1 with a larger bit depth and operating at $f_s = 48$ kS/s, the decrease in localization quality is quite expected and justified. Generally, applying both SRP-PHAT and the proposed method of DOA estimation in the localization procedure yields similar localization quality with SRP-PHAT resulting in slightly more accurate estimates.

Bullet velocity estimation resulted in $\hat{v} \simeq 720$ m/s, which is consistent with the cartridge specification parameters (i.e., velocity of 727 m/s for a range of 100 m). Miss distance estimation via the fitness function f_{fit} is less trustworthy for Experiment 2 due to UGS being very closely positioned to each other, which results in very narrow miss distance ambiguity intervals, especially for firing points Z_2 and Z_3. As a result, if ϕ_Z estimation produces even a slightly inaccurate result, the bullet's trajectory will not fall into the ambiguity interval and true miss distance estimation fails. In our case ϕ_Z estimation performed accurately enough for the bullet's trajectory to be at an edge of the ambiguity interval or very close to it, e.g., firing point Z_2 result in Fig. 18. This means that in the minimal value of f_{fit} appears close to the edge of the ambiguity interval. A more spatially distributed UGS cluster would solve this problem.

5.4 Results of Experiment 3

The number of detected gunshot acoustic events is similar to the one of Experiment 2: 5–6 events per shot on average. The situation with reflections off the surrounding walls is worse for UGS S_1, S_2 and S_3, as they are situated closer to the left and back walls in this case. On the other hand, the effect of SW overlapping with its ground reflection is less evident for the UGS with larger miss distances. Nevertheless, MB is detected as the 3rd peak for 18 out of 19 shot instances.

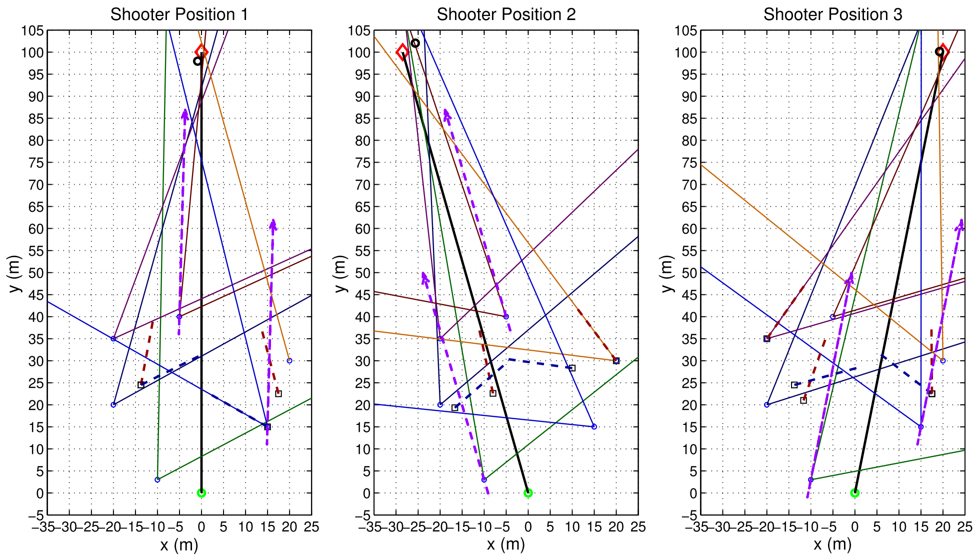


Fig. 20 Experiment 3 localization results for one shot per shooter position (*red* and *blue dotted* $\bar{\phi}^{(SW)}$, $\bar{\phi}^{(MB)}$ of clusters Φ_p; *purple dotted arrow* ϕ_Z and miss distance uncertainty; *black circle* final estimated shooter position)

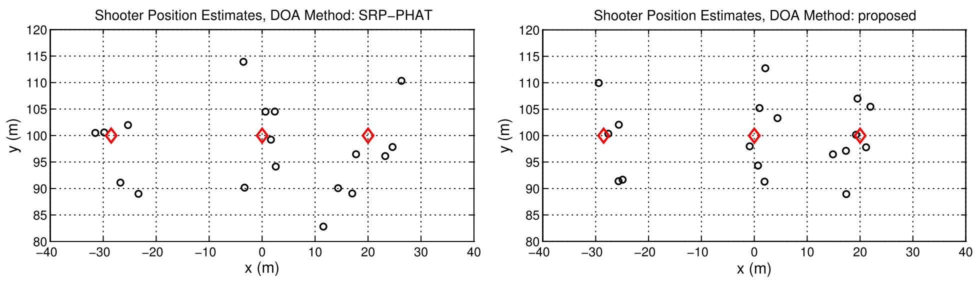


Fig. 21 Experiment 3 localization results for 19 shots with SRP-PHAT (*left*) and the proposed method (*right*) used for DOA estimation. *Black circles* estimated shooter positions; *red diamonds* true shooter positions

The intermediate results of localization and the final shooter location estimate for a single shot from each of the three firing points are presented in Fig. 20. NOI event DOA are removed from the plots for presentation clarity. For point Z_1 UGS $\{S_1, S_2, S_3, S_4\}$ and $\{S_5, S_6\}$ form MB and SW DOA coherent estimate clusters, corresponding to the left and right groups G_L and G_R, respectively. For point Z_2 the UGS belonging to G_L are $\{S_1, S_2, S_3\}$ and belonging to G_R—$\{S_4, S_5, S_6\}$. For point Z_3 the UGS are partitioned as $\{S_1, S_2, S_3, S_4\}$ into G_L and $\{S_5, S_6\}$—into G_R. As the dimensions of the UGS cluster are large enough to be comparable with the distance from the cluster to the shooter, MB DOA do not form a single coherent direction, as was the case in Experiment 2, rather coherent estimates are formed by UGS situated to the left and right of the bullet's trajectory and are skewed towards the shooter's position. Ultimately this can be perceived as a scaled-up version of Experiment 1. The shot angles $\phi_{Z_1} \simeq 90°$, $\phi_{Z_2} \simeq 106°$ and $\phi_{Z_3} \simeq 79°$ are estimated with high accuracy.

Final shooter position estimates for all three firing points are presented in Fig. 21. The estimates are also significantly more scattered, compared to the estimates of Experiment 1. Table 3 presents the ME of localization, calculated using (28). The ME for both Experiments

2 and 3 using the proposed method for DOA estimation is approximately 7 m. The ME of Experiment 3 with SRP-PHAT used as a DOA method is larger, which indicates the supremacy of the proposed method over SRP-PHAT in this case. It can be also noticed from Fig. 21 that Z_2 has only 5 estimates around its true position. This is due to one shot being localized incorrectly and the point residing outside of the figure bounds for both DOA methods. This is a single example of gunshot event identification failure by DOA. If a NOI event has a DOA resembling that of MB and satisfies all the temporal and spatial bounds of the MB check, it can be falsely labeled as MB. Consequently, the TDOA Δt is computed incorrectly and the whole localization procedure can fail. However, this requires the NOI event to corrupt the DOA estimates of several UGS, which is highly unlikely. In our case UGS S_2 and S_3 mistook a NOI event for MB, and their incorrect estimates of distance to shooter steered the cluster's global estimate farther from shooter's true position.

Bullet velocity estimation resulted in $\hat{v} \simeq 725$ m/s, which closely corresponds to the result of Experiment 2. Miss distance estimation via the fitness function f_{fit} operates well for this experiment, as the miss distances for all UGS are sufficient and f_{fit} forms curves, similar to the ones portrayed in Fig. 17, with a single global minimum for the majority of shot instances.

6 Discussion and future work

Although the proposed method of gunshot acoustic component identification using DOA information increases shooter localization robustness, accounting for the destructive influence of various types of NOI events, it has several shortcomings that yet require attention.

The instantaneous bullet velocity estimation via the shot angle needs to be developed into a more general procedure that also accounts for the decrease in bullet velocity with traveled distance. In the experiments the bullet velocity was approximately estimated to be 720–725 m/s, which is significantly less than the 780 m/s muzzle velocity claimed in the cartridge specification. Such velocity reduction even for a 100 m range case can influence localization results. Thus, the degree of this influence needs to be quantified and accounted for in the future.

Alternatively to estimating the distance to shooter by applying (26) in the miss distance ambiguity interval, bearing-only localization methods can be applied. Having identified MB DOA, a least square optimization method, e.g., the bearing-only Total Least Square localization proposed by Dogancay (2005), may be used to estimate the shooter position. However, convergence on the position is doubtful for a tight cluster configuration, like the one used in Experiment 2. Further testing is required to assess the applicability of bearing-only methods under different sensor placement and shooter distance conditions.

The event identification and shooter localization approach needs to be tested in a burst-mode shooting scenario, the peculiarities of which were reviewed in Sect. 4.1. In such a scenario shot instance separation will likely pose a serious problem, so the acoustic event detection procedure will have to be developed further to account for extremely closely spaced shot instances. Also the procedure of sending the shot information to the fusion node is to be reviewed for this case, as sending a large number of packets through the WSN in a very short period of time tends to be problematic.

The problems situated with burst-mode gunshot localization are also related to a case of simultaneous gunshots. If several shots are fired from significantly different shooter positions, the proposed approach in its current state can distinguish between various SW and MB events and produce several position estimates if these gunshot events are not masked by each other and the associated NOI events. The information fusion procedure, however, has to be complemented with additional conditions, which distinguish between several simultaneous SW events in order to assure that the SW events following the one which is detected first are not treated as NOI events.

We also intend to identify the boundaries of application of the gunshot planar geometry model, where either the shooter's or target's elevation above the UGS cluster starts to influence localization accuracy. If the bullet's trajectory does not lie in the same plane as the UGS cluster, the shot geometry cannot be estimated by a planar model, since the conical wavefront of SW cannot be modeled as a planar wavefront, and distance to shooter cannot be estimated by the horizontal projection of the bullet's trajectory. As the results of Experiment 2 have shown, slight elevation of the target does not influence the localization procedure, however, larger elevation levels were not considered in the experiments.

The main problem situated with UGS implementation is situated with the limitations of signal acquisition and processing in real-time. The results of Experiments 2 and 3 show that the reduction of the sampling rate reduces DOA estimation quality and the overall localization accuracy. The influence of applying reduced sampling rates on DOA estimation quality was discussed by us in Astapov et al. (2015b). Therefore, a hardware configuration with a more powerful ADC needs to be developed for future prototypes in order to assure stable sampling at rates equal or higher than the one used in Experiment 1.

Long-term development plans include the expansion of the localization procedure in order to cover all the possible shot scenarios, which were examined in Sect. 3. The specifics of the remaining scenarios are to be researched and a procedure for distinction between the scenarios is to be developed.

7 Conclusion

The paper discussed the absolute need to distinguish SW and MB gunshot events in a scenario with presence of NOI acoustic events, where the MB transient is not guaranteed to strictly follow the SW transient. A shooter localization procedure comprising gunshot acoustic event identification based on DOA information, gunshot geometry estimation and shooter position estimation was presented and verified on real-life data. The main advantages of the proposed localization procedure include its ability to operate asynchronously in a size-invariant WSN, low dependency on gunshot parameter assumptions and increased noise tolerance.

The proposed gunshot acoustic event identification procedure based on DOA information was shown to successfully distinguish the SW and MB gunshot acoustic components from various NOI events. The proposed DOA estimation method was proven to provide DOA estimates, not inferior to the ones produced by one of the most effective DOA estimation methods of SRP-PHAT, while being more computationally effective. The ability of the proposed localization procedure to estimate the shooter's position at a short and medium range with different sensor cluster configurations and under various weather conditions was demonstrated. The proposed localization procedure exhibits high robustness and tolerance to the destructive influence of acoustic NOI events.

Appendix

Multilateration is a technique of estimating object position coordinates based on TDOA information. For the application of shooter localization in the WSN of ground sensors, the shooter's position can be estimated using the TDOA between the MB events, detected by different UGS. As the inter-UGS event time values are used, sufficient node synchronization and temporal, as well as spatial data validation are essential for successful operation of multilateration. Furthermore, the method is applicable only if the MB acoustic events are explicitly identified among other detected gunshot events.

The distance between UGS network node k with coordinates (x_k, y_k, z_k) and the shooter can be defined as a vector length

$$d = \sqrt{(x_k - x)^2 + (y_k - y)^2 + (z_k - z)^2}, \tag{29}$$

where (x, y, z) are the shooter's coordinates and $k = 1, \ldots, K$, where K is the total number of UGS. Thus, knowing UGS positions and times of MB event occurrence t_{MB} for a detected gunshot, the TDOA $\tau_{A,B}$ can be found between two separate UGS A and B. The distance difference between UGS A and the shooter and UGS B and the shooter, $d_{A,B}$ is then calculated as

$$\begin{aligned} d_{A,B} = c \cdot \tau_{A,B} &= c\,(t_{MB}(A) - t_{MB}(B)) \\ &= \sqrt{(x_A - x)^2 + (y_A - y)^2 + (z_A - z)^2} \\ &\quad - \sqrt{(x_B - x)^2 + (y_B - y)^2 + (z_B - z)^2}, \end{aligned} \tag{30}$$

where (x, y, z) are shooter (MB source) coordinates and (x_A, y_A, z_A) are the coordinates of UGS A, and (x_B, y_B, z_B) are the coordinates of UGS B (Liu and Yang 2010). For any group consisting of G UGS the shooter is localizable by the following system of $G - 1$ nonlinear equations:

$$\begin{cases} d_{1,2} = \sqrt{(x_1 - x)^2 + (y_1 - y)^2 + (z_1 - z)^2} - \sqrt{(x_2 - x)^2 + (y_2 - y)^2 + (z_2 - z)^2} \\ d_{1,3} = \sqrt{(x_1 - x)^2 + (y_1 - y)^2 + (z_1 - z)^2} - \sqrt{(x_3 - x)^2 + (y_3 - y)^2 + (z_3 - z)^2} \\ \cdots \\ d_{1,G} = \sqrt{(x_1 - x)^2 + (y_1 - y)^2 + (z_1 - z)^2} - \sqrt{(x_G - x)^2 + (y_G - y)^2 + (z_G - z)^2} \end{cases},$$

where $d_{i,j}$ is the distance difference between the i-th and j-th UGS, and $G \leq K$ is the number of UGS in the group. To estimate the solution to this system of nonlinear equations at least four UGS that have detected MB are needed; this yields three TDOA values $\tau_{1,2}$, $\tau_{1,3}$, $\tau_{1,4}$, and the system is solved by applying a least squares method, e.g., Levenberg-Marquardt. Various practical approaches exist, e.g., as discussed by Bancroft (1985) or by Bucher and Misra (2002). For the ground applications we could simplify the solution with constant z dimension and denote the unknown location of the shooter as (x, y); then we can use the t_{MB} values from only three UGS.

Multilateration methods for WSN highly depend on inter-node synchronization accuracy. Figure 22 presents the results of a simulation of shooter localization using multilateration for the setup identical to that of Experiment 3 (see Sect. 5). The figure illustrates the localization accuracy for all $\binom{6}{4} = 15$ combinations of $G = 4$ UGS groups and $\binom{6}{6} = 1$ combination of $G = 6$ UGS groups with the synchronization error of each UGS randomly chosen from a uniform distribution within the interval of ± 10 ms. The figure shows that larger UGS groups perform with better accuracy than smaller groups with the same degree of node synchronization error. To illustrate the impact of WSN synchronization error on shooter localization accuracy, shooter position estimate mean error (ME), calculated by (28), and its standard deviation (SD) are presented for $G = 4$ and $G = 6$ UGS groups in Table 4. For this simulation we also use the setup of Experiment 3 and assume the WSN clock synchronization error to be in a range of ± 5 ms and up to ± 50 ms. The table shows that in

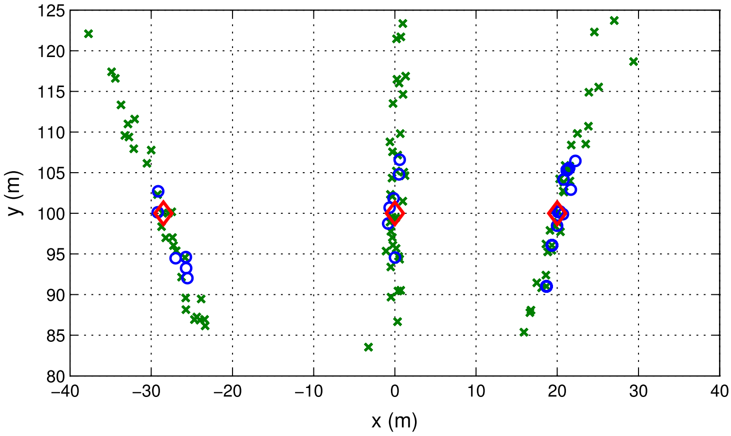


Fig. 22 Shooter localization simulation results for Experiment 3 using multilateration. The theoretical node clock synchronization error is uniformly distributed within the interval of ±10 ms. *Blue circles* shooter positions estimated with $G = 6$ UGS groups; *green crosses* shooter positions estimated with $G = 4$ UGS groups; *red diamonds* true shooter positions

Table 4 Shooter position estimate mean error (ME) and standard deviation (SD) in meters

Node Synch. Error	Parameter	UGS group $G = 4$	UGS group $G = 6$
±5 ms	$\hat{Z}$ ME	6.79	2.57
	$\hat{Z}$ SD	8.43	1.87
±10 ms	$\hat{Z}$ ME	11.29	5.21
	$\hat{Z}$ SD	12.88	3.92
±20 ms	$\hat{Z}$ ME	16.79	11.88
	$\hat{Z}$ SD	17.35	10.01
±50 ms	$\hat{Z}$ ME	23.09	22.45
	$\hat{Z}$ SD	24.20	22.61

order to obtain shooter position estimate accuracy comparable to our proposed method, the $G = 6$ UGS groups should be synchronized to at least ±10 ms, and for $G = 4$ UGS groups the synchronization should be within ±5 ms.

References

Aguilar, J. (2013). Gunshot location systems the transfer of the sniper detection technology from military to civilian applications. In: Proceedings of 47th international carnahan conference on security technology (ICCST), pp 1–6

Aguilar, J. R., Salinas, R. A., & Abidi, M. A. (2007). Acoustical model of small calibre ballistic shock waves in air for automatic sniper localization applications. Proceedings of SPIE 6538, sensors, and command, control, communications, and intelligence (C3I) technologies for homeland security and homeland defense VI 6538:65,381D–65,381D–8

Ash, J., Whipps, G., & Kozick, R. (2010). Performance of shockwave-based shooter localization under model misspecification. In: Proceedings of 2010 IEEE international conference on acoustics speech and signal processing (ICASSP), pp 2694–2697

Astapov, S., Berdnikova, J., & Preden, J. S. (2013). Optimized acoustic localization with SRP-PHAT for monitoring in distributed sensor networks. *International Journal of Electronics and Telecommunications*, *59*(4), 383–390.

Astapov, S., Berdnikova, J., & Preden, J. S. (2015a). A two-stage approach to 2D DOA estimation for a compact circular microphone array. In: Proceedings of 4th international conference on informatics, electronics and vision (ICIEV), pp 1–6

Astapov, S., Ehala, J., & Preden, J. S. (2015b). Performing acoustic localization in a network of embedded smart sensors. *International Journal of Microelectronics and Computer Science (IJMCS)*, *6*(3), 86–95.

Bancroft, S. (1985). An algebraic solution of the gps equations. *IEEE Transactions on Aerospace and Electronic Systems AES*, *21*(1), 56–59.

Borzino, A., Apolinario, J., & de Campos, M. (2014). Estimating direction of arrival of long range gunshot signals. In: Proceedings of international telecommunications symposium (ITS), pp 1–5

Bucher, R., & Misra, D. (2002). A synthesizable VHDL model of the exact solution for three-dimensional hyperbolic positioning system. *VLSI Design*, *15*(2), 507–520.

Carlucci, D., & Jacobson, S. (2010). *Ballistics: Theory and design of guns and ammunition*. Boca Raton: taylor & francis.

Damarla, T., Kaplan, L., & Whipps, G. (2010). Sniper localization using acoustic asynchronous sensors. *IEEE Sensors Journal*, *10*(9), 1469–1478.

DiBiase, J. (2000). A high-accuracy, low-latency technique for talker localization in reverberant environments using microphone arrays. PhD thesis, Brown University, Providence, RI, USA

Dogancay, K. (2005). Bearings-only target localization using total least squares. *Signal Processing*, *85*(9), 1695–1710.

Freire, I., & Apolinario, J. (2011). GCC-based DoA estimation of overlapping muzzleblast and shockwave components of gunshot signals. In: Proceedings of IEEE second Latin American symposium on circuits and systems (LASCAS), pp 1–4

George, J., & Kaplan, L. (2011). Shooter localization using soldier-worn gunfire detection systems. In: Proceedings of 14th international conference on information fusion (FUSION), pp 1–8

George, J., Kaplan, L., Deligeorges, S., & Cakiades, G. (2014). Multi-shooter localization using finite point process. In: Proceedings of 17th international conference on information fusion (FUSION), pp 1–7

Libal, U., & Spyra, K. (2014). Wavelet based shock wave and muzzle blast classification for different supersonic projectiles. *Expert Systems with Applications*, *41*(11), 5097–5104.

Lindgren, D., Wilsson, O., Gustafsson, F., & Habberstad, H. (2009). Shooter localization in wireless sensor networks. In: Proceedings of 12th international conference on information fusion (FUSION), pp 404–411

Liu, Y., & Yang, Z. (2010). *Location, localization, and localizability: Location-awareness technology for wireless networks*. New York: Springer Science & Business Media.

Mazurek, J. A., Barger, J. E., Brinn, M., Mullen, R. J., Price, D., Ritter, S. E., & Schmitt, D. (2005). Boomerang mobile counter shooter detection system. In: Proceedings of SPIE 5778, sensors, and command, control, communications, and intelligence (C3I) Technologies for Homeland Security and Homeland Defense IV, vol 5778, pp 264–282

Millet, J., & Baligand, B. (2006). *Latest achievements in gunfire detection systems*. Tech. rep. DTIC Document

Osborne, R., Bar-Shalom, Y., George, J., & Kaplan, L. (2014). Data fusion from multiple passive sensors for multiple shooter localization via assignment. In: Proceedings of 17th international conference on information fusion (FUSION), pp 1–7

Preden, J. S., Llinas, J., Rogova, G., Pahtma, R., & Motus, L. (2013). On-line data validation in distributed data fusion. *SPIE defense, security and sensing, ground/air multisensor interoperability, integration, and networking for persistent ISR IV, International Society for Optics and Photonics*, *8742*, 1–12.

Sallai, J., Volgyesi, P., Pence, K., & Ledeczi, A. (2011). Fusing distributed muzzle blast and shockwave detections. In: Proceedings of 14th international conference on information fusion (FUSION), pp 1–8

Sallai, J., Volgyesi, P., Ledeczi, A., Pence, K., Bapty, T., Neema, S., & Davis, J. (2013). Acoustic shockwave-based bearing estimation. In: Proceedings of ACM/IEEE International conference on information processing in sensor networks (IPSN), pp 217–228

Sergei Astapov received his Ph.D. degree in the field of Computer System Engineering at the Tallinn University of Technology in 2016. He continues his research at the Department of Software Science at the Tallinn University of Technology and is a member of the Department's Research Laboratory for Proactive Technologies. His research interests include object localization and tracking using wide-band signal analysis, data mining and classification tasks, as well as distributed computing and data fusion algorithms for situation awareness in cyber-physical systems. His recent research concerns object localization and event identification in open environments and acoustic signal based diagnostics of industrial machinery.

Julia Berdnikova received her M.Sc. degree in Telecommunication at the Tallinn University of Technology in 2003. She works as a research scientist and continues her education as a Ph.D. student at the Thomas Johann Seebeck Department of Electronics at the Tallinn University of Technology. Her research interests include digital signal processing, radar and sonar engineering, object classification tasks and distributed sensor networks.

Johannes Ehala received his M.Sc. degree in the field of Computer System Engineering at the Tallinn University of Technology in 2012. Currently he is a Ph.D. student at the Department of Software Science at the Tallinn University of Technology and is a member of the Research Laboratory for Proactive Technologies. His doctoral studies and research interests include self-organization and emergent behavior in cyber-physical-social systems and computational models of distributed computer systems. Currently he is involved in analyzing the temporal aspects of interactions and their implications on the semantic interpretation of interactions in distributed computer systems.

Jaanus Kaugerand received his M.Sc. degree in the field of Computer System Engineering at the Tallinn University of Technology in 2014. Currently he is a Ph.D. student at the Department of Software Science at the Tallinn University of Technology and is a member of the Research Laboratory for Proactive Technologies. His doctoral studies and research interests include dependability in proactive distributed networked systems and emergent behavior. Currently he is involved in analyzing the temporal aspects of interactions and their implications on the semantic interpretation of interactions in distributed computer systems.

Jürgo-Sören Preden received his Ph.D. degree in Computer Science from the Tallinn University of Technology. He is a senior researcher and the head of the Research Laboratory for Proactive Technologies at the Tallinn University of Technology. Jürgo's research interests are focused on distributed computing systems, more specifically on cognition and situation awareness of such systems. His spectrum of activities involves sensing technologies, data processing, computation and communication in ad-hoc sensing systems. Jürgo represents Estonia in NATO STO SCI panel as a national representative.

Appendix 6

VI

A. Riid, **J. Kaugerand**, J. Ehala, M. Jaanus, and J.-S. Preden. An application of a low-cost microwave radar to traffic monitoring. In *2018 16th Biennial Baltic Electronics Conference (BEC)*, pages 1–4. IEEE, 2018

An Application of a Low-Cost Microwave Radar to Traffic Monitoring

Andri Riid*, Jaanus Kaugerand*, Johannes Ehala*, Martin Jaanus† and Jürgo-Sören Preden‡

*Research Laboratory for Proactive Technologies
Department of Software Science
Tallinn University of Technology
Ehitajate tee 5, 19086 Tallinn, Estonia
Email: {andri.riid, jaanus.kaugerand, johannes.ehala}@ttu.ee

†Centre for Intelligent Systems
Department of Computer Systems
Tallinn University of Technology
Ehitajate tee 5, 19086 Tallinn, Estonia
Email: martin.jaanus@ttu.ee

‡Thinnect, Inc.
440 N Wolfe Rd
Sunnyvale, CA 94085, USA
Email: jurgo.preden@thinnect.com

Abstract—**The paper proposes an approach for acquiring traffic data with a microwave technology-based movement sensor. Our aim is to detect the vehicles, their speed and the direction-of-arrival at the observation point, using the Doppler principle and the analysis of sensor signal spectrograms. Experiments on data collected from real traffic confirm the feasibility of the approach, showing near 95% detection rate, near perfect direction-of-arrival detection and adequate velocity estimation.**

I. Introduction

The assessment of traffic flow parameters plays a crucial role in intelligent transportation systems [1]. Real time traffic monitoring, however, presents many challenges, requiring efficient sensors, sensor information processing methods and must consider the cost of the solution. One of the options for monitoring urban traffic flows, currently gaining momentum are Wireless Sensor Networks (WSN). The sensor nodes used as building blocks for the WSN are of low cost, energy efficient and easy to install. They require no additional infrastructure. The technologies used in sensor-nodes for WSN are quickly becoming more capable in detecting the parameters related to traffic flows [2].

Classical (intrusive) sensors applied in vehicle detection include pneumatic road tubes, piezoelectric sensors, magnetic sensors and induction loops which are difficult to install, assume road closures and are liable to damage [3]. Non-intrusive sensor solutions such as active and passive infrared sensors, acoustic and ultrasonic sensors and camera and vision-based approaches are more suitable for WSN-based urban traffic monitoring as they are less troublesome to install and often less costly but are more dependent on weather and/or lighting conditions and may require special maintenance. The solutions based on microwave technology, on the other hand, are less sensitive to light or weather, provide extended range, improved accuracy and are therefore well suited for traffic-monitoring applications [4].

There are two types of microwave radar detectors. The first transmits a waveform with known characteristics, also called a frequency-modulated continuous wave (FMCW) and permits moving as well as stationary vehicles to be detected by measuring the range from the detector to the vehicle [4]–[6]. It also calculates vehicle speed by measuring the time it takes for the vehicle to travel between two internal markers (range bins) that represent known distances from the radar. Vehicle speed is then simply calculated as the distance between the two range bins divided by the time it takes the vehicle to travel that distance.

The second type of microwave radar detector transmits electromagnetic energy at a constant frequency and measures the speed of vehicles within its field of view using the Doppler principle (the difference in frequency between the transmitted and received signals is proportional to the speed of the vehicle) [7]. This type of detector cannot detect non-moving vehicles and variants of this kind of radar are used in speed cameras and police radar guns as well as indoor security systems for intruder detection. The current paper focuses on the latter type of microwave radar sensors and its purpose is to show that those cheap (costing less than 30 Euros) and simple devices can collect rich information from live traffic if coupled with an appropriate signal amplifier and a set of algorithms for the detection of vehicle presence, its direction-of-arrival (DoA) and velocity, which have been developed by the authors.

The reports on using low-cost microwave radars in traffic monitoring applications in scientific literature have been sparse and the purposes of such applications vary. Alimenti et al. [7] developed a 24 GHz Doppler radar and demonstrated its long velocity measurement range with just a couple of vehicles. Fang et al. [8] focused on the detection and classification problems. The radar was installed above two unidirectional lanes of a motorway and 95% detection rate was reported. The classification algorithm was verified with data of 164 cars belonging to 3 vehicle size classes and 94.8% classification accuracy was reported. Misans and Terauds [9] proposed a method for vehicle velocity and length estimation but do not provide much material about the results. Zelenkov et al. [10] considered the same key parameters of the road traffic as in current paper but their experiments involved only 13 vehicles. This implies that the application of such sensors in traffic monitoring is not yet an established methodology.

978-1-5386-7312-6/18/$31.00 ©2018 IEEE

II. The Sensor

Microwaves are the electromagnetic waves whose frequency ranges from 0.3 GHz to 300 GHz. Microwave motion detectors emit microwaves into the specific region, detect the intruder's motion by analyzing the frequency of received microwaves after the reflection from the intruder and trigger an alarm as a consequence if necessary.

The main principle of operation for these sensors is the Doppler effect. A microwave motion detector circuit comprises of the transmitter, receiver and the alarm related circuit. The transmitter sends off microwaves with a specific frequency into the designated area. As soon as they strike an intruder moving with a velocity, the frequency of the signal changes. The frequency shift caused by the Doppler effect can be calculated by

$$F_d = 2V\frac{F_t}{c}\cos\theta, \tag{1}$$

where F_d is the Doppler shift frequency (i.e. the one we register with the sensor), V is the velocity of the target, F_t is the transmit frequency (e.g. 9.35GHz), c is the speed of light ($3{\cdot}10^8$ m/sec) and θ is the angle between the target moving direction and the axis of the module.

If a target is moving straight toward or away from the sensor then the formula reduces to $F_d = 62.333V$, i.e. the speed of 1 m/sec corresponds to the Doppler shift of 62.3333 Hz or 1 km/h corresponds to 17.3148 Hz. The latter rate depends only on the transmit frequency. The sensor compares the transmitted and received signals, producing an output signal.

In a typical movement sensor, external signal processing circuitry amplifies and analyses this signal so that when the specified criteria are met, an output signal can be generated to activate a process such as turning on a light or initiating an alarm. The microwave sensor employed in current project, MDU1740 by Microwave Solutions, with the 9.35 GHz transmit frequency, however, is provided without the external signal processing circuitry. It generates an output signal with an amplitude dependent on the size, distance and reflectivity of the object at a frequency proportional to its velocity. Because the amplitude of the signal is in the range of few microvolts, an amplifier needs to be added to bring the amplitude of the signal up to the range suitable for most ADCs.

III. The Amplifier

We designed a simple two-stage circuit (see Fig. 1). In order to minimize the noise due to large gain (~2500) determined by R1/R2×R3/R4, a precision JFET amplifier (ADA4610-2) has been employed.

The circuit contains two high pass filters consisting of C2, R2 and C3, R4 and two low pass filters consisting of R1, C1 and R3, C4. These filters yield the frequency bounds 3.4 Hz and 2800 Hz, respectively. The latter value establishes the speed measurement limit at 160 km/h, which is more than sufficient for normal traffic conditions.

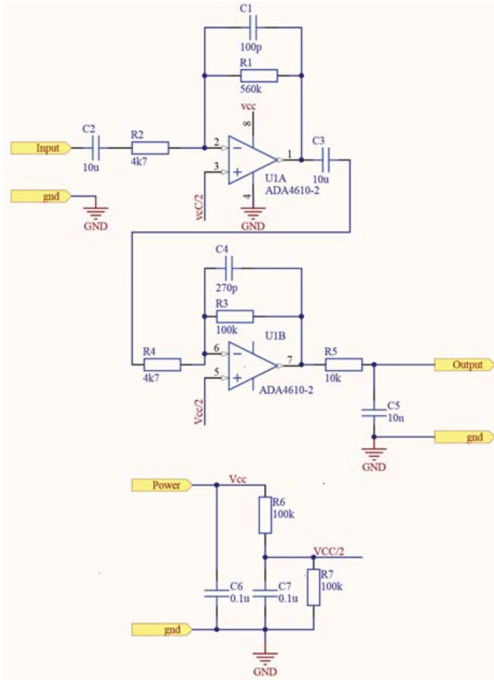


Fig. 1. Circuit of the designed sensor amplifier. Input is connected to the MW sensor (IF) output, output is connected to the ADC. Supply voltage is 5V.

IV. The Detection Algorithm

The sampling frequency (f_s) throughout the experiments is 3000 Hz, ensuring that we are able to detect the speeds up to 86 km/h, sufficient for urban traffic situations. Note that lower sampling rate is preferred because it reduces the computational load and thus makes in-sensor signal processing more feasible. First step of the detection algorithm is to compute a spectrogram of the measured signal with a linear amplitude axis from the raw time signal using the Fourier transform. The signal is windowed into 0.5 sec long segments that overlap by 80%. A long enough window reduces spectral noise and provides sufficient frequency resolution; high degree of overlap, on the other hand, provides good time resolution.

Fig. 2 depicts a raw signal and the corresponding spectrogram generated by the passing of two cars, the one on the right coming from the direction facing the receivers and transmitters of the sensor (in the context of the measurement site, from the left) and another (at a considerably lower speed) from the opposite direction. The amplitude of the raw signal is at its maximum when the vehicle is at its closest to the sensor (its magnitude also depends on vehicle speed and size/length). In the spectrogram, both cars leave characteristic lobes when meeting the sensor, with more focused tails pointing to the direction of arrival that are visible long before or after anything could be registered in the raw signal. The drop in peak frequency in the proximity of the sensor is caused by the increase of the angle of microwave reflection. The raw signal also contains considerable background noise.

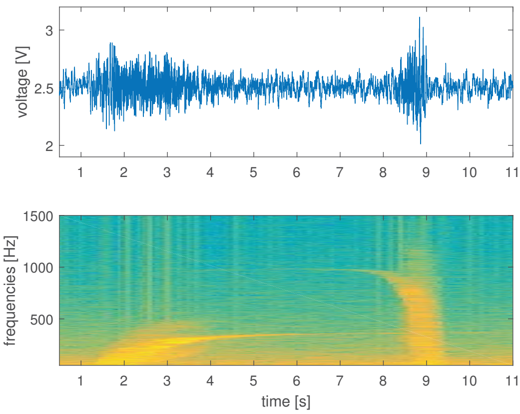


Fig. 2. Raw signal and its (logarithmic) spectrogram

A. Vehicle Detection

The detection of a vehicle is based on the calculation of the total spectral power for each spectrogram window over the 50-1500 Hz bandwidth in the linear scale. We establish a threshold value that defines a detection interval $[a_i, b_i]$ for each passing vehicle where a_i is the instant when the rising power crosses the threshold and b_i is the instant when the fading power crosses the threshold. The threshold value of 0.4 W/Hz provides a reasonable detection accuracy except for the cases where two vehicles arrive at the sensor location simultaneously (Fig. 3).

B. DoA and speed determination

For each detected interval $[a_i, b_i]$ we construct two supporting intervals $[\alpha_i = a_i - \Delta_1,\, a_i - \Delta_2]$ and $[b_i + \Delta_2,\, \beta_i = b_i + \Delta_1]$ at both sides of the original detection interval and observe the sum of total spectral power within these intervals. Note that Δ_1 is chosen relatively small (0.5 secs) in current application so as to capture the tail of the spectral lobe while it is at its freshest (and to better separate the vehicles in a tight sequence) and Δ_2 that acts as a safety margin, so to speak, to avoid the inclusion of irrelevant high spectral power values due to always imperfect placement of the detection interval itself, is 0.2 secs. The basic assumption is that more spectral power is to be found on the side of the vehicle's DoA; that is, if the sum of total power spectrum values within $[\alpha_i,\, a_i + \Delta_2]$ is higher than in $[b_i + \Delta_2,\, \beta_i]$, the vehicle arrived from the left and vice versa.

Once the DoA has been established, we identify the peak frequency in the corresponding interval (and further restricting

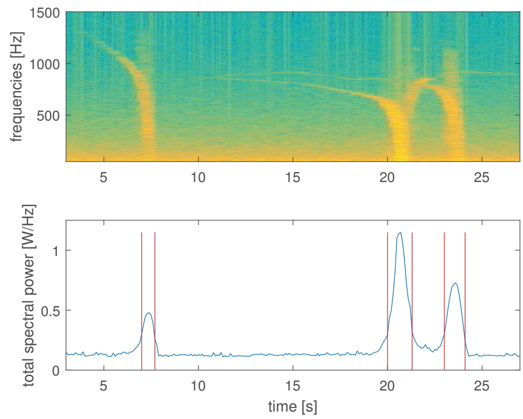


Fig. 3. A spectrogram containing the traces of four vehicles (above), the total spectral power in time and corresponding detection intervals marked by red vertical lines (below). Note that there are two vehicles arriving at the scene at 20 second marker and one of them escapes the detection.

the bandwidth to [250 Hz, 1500 Hz]) which can then be translated to speed in km/h. This approach works well for individual vehicles, however, when cars arrive at trailing distances less than $2\Delta_1$, the task becomes more problematic. For a sequence of N vehicles where $\alpha_{i+1} - \beta_i + < 2\Delta_1$, $i = 1, ..., N-1$, the two supporting intervals are defined as $[\alpha_i,\, a_i - \Delta_2]$ and $[b_N + \Delta_2,\, \beta_N]$. This somewhat reduces the problem but not entirely and information loss in case of sequences consisting of more than two cars is unavoidable. Moreover, if there are two or more traces within the same supporting interval, the vehicle with a stronger signal is always having priority.

V. Experimental Results

Traffic measurements were carried out at Üliõpilaste tee, a two-lane street with relatively quiet traffic in one of the outskirts of Tallinn. The experiment lasted for 30 minutes and involved 96 vehicles, mostly passenger cars, two buses, a large truck and a motorcycle. The sensor was placed at 3 m from the edge of the pavement, and was pointed at the road at 30 degree angle (simulating a situation where the sensor might be attached to an existing street light post). The algorithm was able to detect 91 of 96 vehicles (94.79% accuracy). Of those, the driving direction (49 rightbound and 47 leftbound vehicles) was properly determined in all cases, except one. Considering the speed measurements, there were just 3 instances where errors were encountered, however, the errors were quite drastic. In one case (error 12.2 km/h), a fast moving car overtook a slowly moving car right at the sensor location; in two other erroneous cases (11.1 and 15.1 km/h) two cars coming from opposite directions met close enough to the sensor and the mistake derived from the car in the closer lane taking priority.

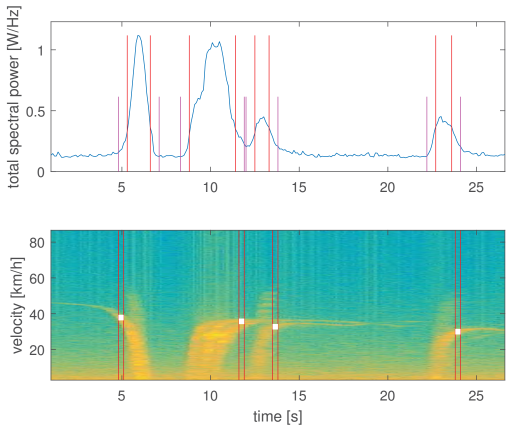


Fig. 4. DoA and speed detection. Each detection interval in the above subplot is coupled with two further intervals at each side of the detection interval (with $\alpha_i$and β_i marked by shorter vertical lines) of which the one containing more spectral power (shown in red in the lower subplot) determines the DoA. The peak frequency within the interval (the white spot) is then used for velocity calculation.

The velocity histogram of all detected vehicles is depicted in Fig. 5. One should note, however, that the velocities are measured from the very proximity of the sensor and are thus underestimated by 2-5 km/h because at this point the microwaves reflect back at a steeper angle. We confirmed this by additional drive-bys at controlled speed. This phenomenon could be compensated, e.g. by dividing the estimated velocity by $\cos(20°)$ but this solution would not be valid for all vehicles because the lengthy ones provide more stable reflection angles (this can be see seen in Fig. 4 where the second vehicle from the left is a bus).

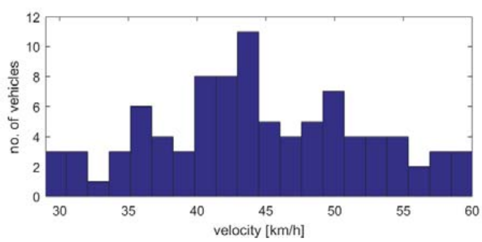


Fig. 5. The velocity histogram, which shows that 23% of detected cars were exceeding the speed limit (50 km/h).

VI. Conclusions

The proposed solution for applying low-cost microwave sensors in traffic monitoring was successful with its nearly 95% detection rate, precise DoA determination and satisfyingly accurate velocity estimation. Although these results have been obtained by offline analysis, the algorithms described in the paper can be implemented in the form of in-sensor software capable of doing on-site work, reporting the results from traffic with an approximately one second delay, principally determined by the Δ_1 parameter of the DoA detection algorithm and the processing power of the computing platform.

A single-sensor solution, however, has its limitations. It is not able to detect the vehicles that arrive simultaneously at the sensor location and has problems in the DoA and/or velocity determination for vehicles that drive at short (less than a second) trailing distances or meet near the location. A multi-sensor solution that will hopefully overcome these problems will be addressed in our further research and its results will be implemented in a pilot project as part of the Smart Environment Networking Technology program (SmEneTe).

Acknowledgement

The research was partially supported by Archimedes Foundation Smart Specialization Program and Thinnect OÜ in the context of the applied research project NSP100 (Smart Environment Networking Technologies 2).

References

[1] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: a deep learning approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 2, pp. 865–873, 2015.

[2] J. Ehala, J. Kaugerand, R. Pahtma, S. Astapov, A. Riid, T. Tomson, J.-S. Preden, and L. Motus, "Situation awareness via internet of things and in-network data processing," *International Journal of Distributed Sensor Networks*, vol. 13, no. 1, pp. 1–21, 2017.

[3] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 187–197, 2015.

[4] T.-J. Ho and M.-J. Chung, "An approach to traffic flow detection improvements of non-contact microwave radar detectors," in *International Conference on Applied System Innovation (ICASI)*, 2016, pp. 1–4.

[5] Y. Xuan, H. Meng, X. Wang, and H. Zhang, "A high-range-resolution microwave radar system for traffic flow rate measurement," in *Intelligent Transportation Systems Conference (ITSC)*, 2005, pp. 880–885.

[6] S.-L. Jeng, W.-H. Chieng, and H.-P. Lu, "Estimating speed using a side-looking single-radar vehicle detector," *IEEE transactions on intelligent transportation systems*, vol. 15, no. 2, pp. 607–614, 2014.

[7] F. Alimenti, F. Placentino, A. Battistini, G. Tasselli, W. Bernardini, P. Mezzanotte, D. Rascio, V. Palazzari, S. Leone, A. Scarponi *et al.*, "A low-cost 24GHz doppler radar sensor for traffic monitoring implemented in standard discrete-component technology," in *European Radar Conference (EuRAD)*, 2007, pp. 162–165.

[8] J. Fang, H. Meng, H. Zhang, and X. Wang, "A low-cost vehicle detection and classification system based on unmodulated continuous-wave radar," in *Intelligent Transportation Systems Conference (ITSC)*, 2007, pp. 715–720.

[9] P. Misans and M. Terauds, "CW doppler radar based land vehicle speed measurement algorithm using zero crossing and least squares method," in *13th Biennial Baltic Electronics Conference (BEC)*, 2012, pp. 161–164.

[10] A. Zelenkov, D. Liepkalns, A. Yershov, and A. Maklakovs, "Development and experimental validation of automatic radar recorder of the number and speed of vehicles for opposite traffic directions," *Automatic Control and Computer Sciences*, vol. 46, no. 1, pp. 25–33, 2012.

Appendix 7

VII

L. Motus, M. Teichmann, T. Kangilaski, J. Priisalu, and **J. Kaugerand**. Some issues in modelling comprehensive situation awareness. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 540–545. IEEE, 2019

2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)
Bari, Italy. October 6-9, 2019

Some issues in modelling comprehensive situation awareness*

L. Motus *Senior Member IEEE*, M.Teichmann, T. Kangilaski, J. Priisalu, and J. Kaugerand *MIEEE*

Abstract: **This paper discusses some issues confronted in the design process of a system creating comprehensive situation awareness for a small country. We strive to merge the experience obtained in modelling and analyzing cyber-physical-social systems and the results published on situation awareness. For that, we develop a reasonably simple and efficient monitoring framework that enables to capture and analyze heterogeneous dynamic phenomena in a country. The final goal is to build a model for comprehensive situation awareness, which merges information from a feasible set of interoperable models, describing operation of country's major institutions; and provides situational information for decision-makers at all required levels.**

INTRODUCTION.

Comprehensive situation awareness (CSA) system supports country's everyday management by collecting and processing information characterising nation-level functioning and interoperating of major political, social and economic institutions, public and private organisations, enterprises, and social networks. Many of the listed interoperating institutions and networks operate semi-autonomously and are coordinated by a set of common goals. The CSA system transforms, processes, and distributes collected information together with the deduced prognoses to stakeholders – in correspondence to their requirements and access rights – for decision-making. The application of stakeholders' decisions closes the observation and decision loop – and the resulting OODA (observe, orient, decide, act) loop is expected to improve the quality of country's management.

Conceptually the task is to create a CSA system able to interoperate with truly complex nation-level System of Systems (SoS) [1]. Those SoS often have time variable composition, and interaction pattern of semi-autonomous constituent systems, which can be cyber-physical systems or cyber-physical-social systems. Large and complex aggregates of components inevitably appear in SoS, and their properties cannot be explained by simple extrapolation of constituent components' properties [2]. Based on this observation, modelling large public organisations (e.g. ministries, large industrial enterprises) so that their models capture enough details of their behaviour, requires building a suite of models -- macro-level model to describe overall functioning of the organisations, and several specific models to understand how and why the macro-level behaviour is generated [3].

Following the classical divide and rule principle, we discuss separately:

- the physically existing interoperating components of SoS (organisations, institutions, enterprises, social and communication networks, stakeholders, etc.) together with their environments -- further called physical-universe, and
- models describing components of the physical-universe, and situated in cyber-space enabling thus monitoring interoperability of components, processing and analysing of collected data, detection of situations, assessment of situation outcome, prognosticate evolution of situations, supporting decision-making (by stakeholders, or by CSA) and analysis of the potential impact of those decisions on physical universe – further called mirror-universe.

Coordination of physical- and mirror-universes and their smooth collaboration has the pivotal role in success of CSA system. The physical-universe has usually a rather static structure and well-defined functionality – here "rather" means that occasionally the country's administration may reorganise the structure and/or functionality of physical-universe. Such changes are considered as caused by Force Majeure and lead to corresponding changes in the mirror-universe.

Comprehensive situation awareness system sets to find, or develop, and interlink together methods to address the following issues:

- delineate the physical-universe, comprehend and observe modus operandi of its components, and as a whole; the physical-universe comprises, for instance, existing political, social and economic institutions, public and private organisations, enterprises, social networks, and other entities of interest
- build a respective mirror-universe in the cyber-space, i.e. develop and implement a sound suite of interoperating (semi-)formal models of the objects comprising physical-universe; mirror-universe may invoke modifications in the physical-universe, if so decided by the stakeholders
- some advanced analytical features are operated in mirror-universe, e.g.:

*Research funded by Estonian Research Agency and co-funded by the European Social Fund.
L. Motus is with Tallinn University of Technology, Department of Software Science, Ehitajate tee 5, 19086, Tallinn, Estonia (corresponding author phone +372 5049 717; e-mail: leo.motus@ttu.ee).

M.Teichmann, T. Kangilaski, J. Priisalu, and J. Kaugerand are with Tallinn University of Technology, Department of Software Science.

- tools providing the ability to check consistency, temporal and spatial integrity of observed data,
- tools for verifying constraints on required interoperability imposed by stakeholders and by objects from the physical-universe
- tools for disseminating obtained situation awareness within the physical- and mirror-universes, strictly following the protocol that permits access to information
- tools for prognosticating (by simulation) the impact of decisions made by stakeholders regarding adjustment of physical-universe
- tools for monitoring and assessing connectivity and functional interoperability of entities in physical-universe, handling anomalies, and resolving cyber incidents,
- tools for harmonising the provided situation awareness with mental models of stakeholders
- tools for detecting pre-defined patterns of interest (situations), with the ability to detect unusual/unexpected changes and assess their potential impact on the system's behaviour

The above list of procedures, methods and tools is further extensible.

Section II surveys research that fosters modelling of cyber-physical-social systems in the context of systems of systems. In particular, related to issues checking temporal and spatial consistency of acquired or measured information, to analyse the impact of autonomy of interoperating systems on their behaviour, ways to increasing self-awareness of constituent subsystems – so as to bolster resilience and reliability of the overall system.

Section III discusses how to delineate the physical-universe, points out major constraints on interoperation of constituent subsystems, provides some guidelines for organising collaboration of constituent subsystems operating under disparate time-systems. Some possibilities to reduce the complexity of the description of physical-universe have been discussed. Section III concludes with discussing modelling requirements on constituent subsystems (organisations, enterprises, social groups, etc,).

I. Related research and observations

In the case of comprehensive situation awareness, the research objects comprise political, social and administrative institutions, organisations, enterprises, and their networks, technological processes, social processes, logistics, etc., as well as climatic and natural processes, and multitude of stand-alone phenomena. The diversity of objects to be considered and modelled is enormous, the reasonable outcome is to select only the most influential phenomena and attempt to capture their most relevant features. In the following, we browse the modelling methods applied to stand-alone objects and/or modelling complex (networked or aggregated) systems built from several stand-alone objects.

Highly sophisticated governance, manufacturing and transport systems, computers embedded in the environment, increased social networking of humans, plus increased social instability, and rapid changes in climate have raised pragmatic interest to managing complex systems. The research domain of complex systems is rapidly expanding – traditional research of systems with natural origin such as climate, biological organisms, ecosystems, or with engineering origin, or stemming from social needs has been expanded to systems that aim to manage country's comprehensive defence – a feasible research can be built upon the highly interdisciplinary notion of cyber-physical-social systems (CPSS). See, for instance [4], [5], [6], [7].

The cognitive and perception mechanisms in CPSS are more susceptible to efficacious in-depth analysis and on-line engineering as compared to those in stand-alone systems of natural or social origin. This enables to consolidate, or combine, methods from a variety of research areas. Such as enhancing perception system, applying non-classical models of computation for data processing, arising methods for detection of emergent behaviour and its mitigation. In many cases, this opens productive research perspectives for better understanding many new aspects in complex systems and fosters development of methods for building situation awareness that improves ability to predict, or manage situations promptly.

Large part of situation awareness (SA) studies has focused on human-human and machine-human context – meaning that the mental model, which is a pivotal tool in decision-making based on situation awareness, resides in a human brain. With the appearance of cyber-physical-social systems concept, the research focus of SA has expanded and today covers often machine-machine context as well. The creator, owner and user of SA may be a smart computer system or some other software intensive device instead of a biological creature. In addition to introducing new research problems, the smart computer systems as carriers of mental models have some advantages – they can be engineered, and re-engineered if necessary, to foster creation and fast sharing of SA in complex systems with incomplete information. For instance, it may be possible to enhance perception tools and procedures, to refocus or readjust cognition procedures and references, strengthen the system's impact on the environment as required by obtaining a well-adjusted mental model, and deduce timely and efficient decisions to improve the situation.

Creating timely situation awareness for a CPSS is not a straightforward process due to persistent evolution of system's composition and its internal as well as external network of interactions. The evolution of CPSS can only partly be understood and managed due to autonomy of many components, due to only partially available information about causal relations, due to partially observable, temporally and spatially sensitive behaviour, and due to occasional appearance of emergent behaviour. This project studies possibilities and methods to mitigate some of the listed obstacles by expedient engineering of cyber-physical-social system during its design and maintenance, and/or during operation by better monitoring its operational characteristics and by obtaining better prognosis for system's evolution by checking the effect of made decisions by simulating their impact on models before employing them in CPSS. Some sources call this process "understanding the meaning of acquired situational information".

On-line engineering of CPSS architecture becomes feasible if we substitute conventional (algorithmic) methods for modelling complex systems with interaction centred multi-agent modelling. This substitution enables to circumvent some limiting features of conventional algorithm-centred modelling that are characterised by "ballistic computations" [8] and stem from Turing machine paradigm. This has led us to studies of systems with architecture that supports self-awareness; we are building on a generic architecture based on a non-classical model of computation [9]. The interaction-centred model and resulting new system's architecture foster close collaboration between control, software, and systems' engineers. This approach also enables to handle quality of service, reliability, safety, and cyber security issues from the early stages of the system's development. This architecture has been further enhanced to support system's self-awareness, which helps to mitigate the impact of emergent behaviour [10].

We consider any implemented cyber-physical-social system as a dynamically varying network, comprising three types of interacting nodes – represented by single humans or social groups, natural and/or artificial processes, and (potentially smart) artefacts. All these components are equally important for expedient operation of the system. Another useful paradigm for describing a wide class of operating cyber-physical-social systems is a cognitive multi-agent system. In complex cases, like situation awareness system for country's comprehensive defence, one might benefit from combining several well-focused operational paradigms.

Prior to applying situation awareness tools, it makes sense to re-engineer the existing physical and social parts of CPSS. For instance, by inserting additional sensors to improve observability; by substituting relevant internal and external direct and/or indirect interactions with mediated interactions [10] to improve systems controllability and transparency of systems' internal structure. To reduce overall complexity of the system we suggest applying modest form of "divide and rule" method by dividing the CPSS into interacting autonomous entities where each entity may exhibit self-X features. The autonomous subsystems increase resilience of the CPSS and improve system's fault tolerance to random misleading messages.

The mathematical models for time-variable objects and attempts to control such systems lead to ill-posed problems that need sophisticated mathematical tools -- see a survey of research in [11]. Ill posed problems give asymptotic solutions, which are not readily usable in case of systems of practical complexity and size.

Pragmatically, in such cases, we rely on simulations based on multi-agent models of CPSS built in cyber space and capturing essential properties of natural and social processes (and their interactions). One successful practical test has been with Sentient World Simulation method, developed at Purdue University; see for instance [12,13]. In this project we build a synthetic mirror of the real world that is persistently co-ordinated with the currently perceived real-world information and simulate real-world processes in the mirror world.

The following illustrates some problems related to building, analysing and engineering a CPSS model in cyber space. The model of a real-world system is typically a multi-agent system represented by a network of interacting, heterogeneous contextually smart agents – whereas several agents may represent physical and social aspects. The network topology in cyber space may need to be changed dynamically – either because the government/parliament has changed physical universe, or because upgraded models are substituting the old models; new agents can be added, existing agents can be substituted, or removed on-line. Agents that interact with the real-world entities usually violate strict rules adopted by conventional (Turing machine paradigm based) computing, also known as "ballistic" computing. To relax the strict rules of conventional computing we stop requiring that the composition of network nodes apply only algorithms that follow requirements assumed by Turing computable functions. Instead, we assume that network nodes are mappings from the domain of definition to value range (that can be implemented by great many of different algorithms), have historical memory, are restricted by quantitative constraints (e.g. execution time, and/or node location). Edges connecting nodes (and describing interactions between nodes) may just transfer data from producer node to the consumer node, and/or map the data in a rather complex way. Those two amendments form the basis of interaction-centred models of computation; see for details [14], [15], and [16].

In the case of CPSS, the interaction-centred model of computation caters for specific system requirements: components should have historical memory, they should not be isolated from the external influence during their execution, they have to possess at least time and location awareness, they may enjoy behavioural autonomy and may exhibit some self-X properties. Due to system's complex structure, autonomy and self-X properties of components, we cannot exclude the occurrence of emergent factors, and their potential impact on the behaviour of system. The emergent behaviour can have either harmful or beneficial impact on the system's behaviour. In the previous century all the emergent factors in embedded systems were assorted as exceptional (i.e. not pre-planned, or not expected) cases and were either shielded and/or eliminated. Emergent behaviour might be extremely useful for CPSS, in many cases, and should be encouraged – just think how ingenious human specialist solves problems. Therefore, the first step is to assess the potential impact of emergent behaviour on systems performance, and the second step is to foster the emergent factor if its impact is positive or supress if it has negative impact. The difficulty is that those decisions are to be made on-line.

In addition to superficial survey of related research, some observations, listed in the random order, might be useful for elaborating models to be used in comprehensive situation awareness system

- Today the term situation awareness (SA) has become a phrase, which indicates wide interest to the topic and in some cases brings in dimming of the true essence of SA. From the positive side the theoretical foundations of SA are maturing [17, 18]. In addition to that, there are some indications that the anthropocentric belief -- SA capability is characteristic to humans only (and may be to certain extent to other biological creatures as well) -- has been relaxed in [18]. Today it is acceptable that smart artefacts (e.g. embedded computers) might be able to develop situation awareness of their own. From the negative side of becoming a buzzword, many conventional data acquisition (or information processing) systems are called situation awareness system, which might confuse the funding organisations.

- Sometimes SA systems are equated to systems for crisis management. What happens is that SA and crisis management systems are closely co-operating – SA system serves as an early warning system that predicts the emerging crisis, prepares initial data for crisis management system, and invokes the crisis management in due time. Prediction is often based on indirect indicators situated outside of the proper crisis domain. Sometimes SA system and crisis management systems are merged to reduce the latent period. However, the composition and basic functionality of SA systems and crisis management systems remain different despite the merger.
- It may be reasonable to distinguish between three types of situation awareness:
    - **passive situation awareness**, we are not interested in systematic acquiring and processing SA, we believe that we can manage e.g. crisis, with incidentally provided/available information
    - **reactive situation awareness**, we acquire SA actively but are worried only about emerged crises and invest only to managing emerged crises
    - **proactive situation awareness**, we are trying to forecast the crises, and actively attempt to mitigate/avoid the crises, and if necessary switch to crisis managing routine.
- Each of these types requires different models – passive SA can live without specific models, minimum requirements for reactive SA are input/output data flows, and some information about resources and other constraints; whereas proactive SA assumes a detailed model of an object/institution and related decision-making processes.
- Estonia has had some experience with virtual situation room (VSR) concept -- a layer of situation awareness system for bringing all the information feeds together and to make them available for collaborative effort of SA and crises/incident management systems; for instance, cyber defence exercises (e.g. Locked Shield), and AbuseHelper project that aims at automatic handling of incidents in CERTs. Check also https://www.isao.org .
- MAJIIC Multi-sensor Aerospace-ground Joint ISR Interoperability Coalition. The primary aim of the project is to improve the commanders' situation awareness through collaborative employment and use of interoperable ISR sensor and exploitation capabilities. MAJIIC enables interoperability between ISR and C2 systems using common interfaces for data formats and exchange mechanisms, leaving the inner workings of each national system outside of the scope of the project.

II. Delineating the Physical Universe

The physical-universe comprises natural and man-made objects, e.g. organizations, institutions, enterprises, their environment, and involved social groups. Performance of some objects depends heavily on human behavior, e.g. parliament, top leadership of large businesses, and is susceptible to semi-formal modelling only with severe reservations. The role of emergent behavior in such objects is remarkable and its modelling requires specific approaches, e.g. [10]. In such subclass of cyber-physical-social systems, one should rely on verbal information provided by well-informed people whereas conventional computers and cyber-physical systems have auxiliary role in collecting, storing and pre-processing information and, in assessing the impact of made decisions.

The other subclass of cyber-physical-social systems where people are working in the loop is increasing – e.g. groups of people have substantial influence on the behavior of natural and/or man-made systems. Typically, people cannot be separated from those systems without seriously altering their functionality, see [5, 6, 7].

Situation awareness studies traditional systems, as well as system of systems (SoS) and covers a wide range of applications, see for example [17, 18], and [19]. Usually the focus of publications is on mental models and on using situation awareness as a set of situational parameter values. The existence of detailed models for cyber-physical-social systems, actual regular functioning of those system, as well as validity and coherence of collected situational parameters and evidences are taken as granted, or their study stays in the background and discussed under different disciplines. In the case of comprehensive SA the existence of some detailed models is essential.

Typically, in teams, we spread the situation awareness by sharing information freely to team-members– so that the whole team has the same information and team member's behaviour depends on his/her mental model. In comprehensive situation awareness system, information spreading is more complicated – parts of a system may have privileged/selective access to situational information. Hence, different parts of a system may have to operate on different subsets of situational information – this is called distributed (or system) situation awareness.

In a truly complex case of comprehensive situation awareness of a country, we may need rather sophisticated procedures that permit access to some parts of information. The same comment applies to information exchange between constituent parts of the SoS, and to archive of the SA information. The comprehensive situation awareness system is highly sensitive to cyber-attacks, and information leakage to unauthorised parties.

A. Multiple self-sustaining time systems

Traditional "anthropocentric" scientific disciplines – e.g. mathematics, physics, biology, psychology, -- manage in principle with one single time (for instance, UTC), although they may have specific requirements on the origin of time, on whether the time is reversible, or strictly increasing. Embedded computers that have their own inner time and typically want to influence the behavior of an autonomous external physical process, which follows its own operational rules and time systems). Over 99% of all computer processors are embedded in different time constraint environments [25]. The designers of such systems face the actual necessity to coordinate simultaneously two disparate time counting systems, or to fail in attempting to influence the behavior of a non-trivial physical process by the computer.

Another, less "exotic", example of simultaneous existence of two time counting systems is the case of two simultaneous crisis (e.g. forest fire). The crisis handling system has to time tag all the information related to this phenomenon – its detection time, its estimated starting time, and all the actions taken to extinguish the fire, arrival of resources, etc. Typically, the fire detection time is recorded in terms of coordinated universal time (UTC), whereas the following events could be recorded, either in topological time (fixing the order of events and actions) or, in a more professional case, in relative metric time with the origin at the detection instant of the main event (e.g. forest fire). The manager of two simultaneous forest fires has to maintain two separate topological times, or two separate relative metric times.

In a comprehensive situation awareness system, we need to manage many occurrences of simultaneous incidents. It is important to analyze the correlation of occurrences happening in temporal proximity of each other, although in different locations, to realize the potential long-term purpose of those occurrences, and to detect some clues for finding the motives/persons related to those occurrences. This becomes possible only if we accept, and are able to handle, simultaneous existence of multiple metric, and/or topologic times, and apply time system similar to that applied in real-time embedded systems [26].

B. Mitigating complexity of models

Behavioural complexity of CPSS can be mitigated by engineering autonomous constituent components of SoS, so as to foster adoption of self-X properties – e.g. self-organisation, self-healing, self-protection, and self-awareness. This would allow paying more attention to managing overall behaviour of the CPSS by focusing on collaboration of autonomous constituent components and pay less attention to details of in-component operations. This principle leads us to preferring the description of CPSS as multi-agent systems comprising, in the ideal case, autonomous smart agents that possess self-X properties and interact with each other and with their environment(s). System's behaviour needs to be checked on-line due to potentially time variable structure and composition of CPSS, and due to strict dependability requirements. Detection of behavioural changes and assessment of their impact on dependability can usually be carried out on-line. The impact of major changes in behaviour should preferably be studied by simulation and may need suspension of normal operation.

In the case of comprehensive situation awareness system, the physical-universe comprises, in principle, a myriad of interoperating heterogeneous components. In many situation awareness cases, e.g. [17, 18], this burden has been alleviated by limiting the extent of mirror-universe that reflects the physical-universe, or by attempting to build less heterogeneous models of the physical universe. A well-functioning methodology for modelling complex systems departs from the idea of architectural frameworks – DoDAF, UAF, etc – that enables to build suitably approximated models.

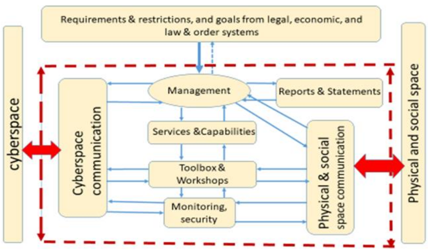
Another possible approximation that contributes to transparency of reasoning is based on separate modelling the functioning of a component and its interoperation with the other components in a SoS. Fog (or mist) computing technology, as subcategory of cloud computing, with the accompanying data fusion, aggregation, and other type of processing illustrates this approach, see [20, 21].

In the previous century, the emergent behaviour was considered as an unexpected and harmful feature that had to be immediately cancelled. Emergent behaviour does not depend on the design and implementation of system's constituent parts but occurs due to not very strict connections between constituent parts. For instance, SoS structure allows some autonomy in establishing interaction between system components, see [22]. Therefore, such behaviour occurs unexpectedly to a designer or observer. Emergent behaviour is characteristic to complex systems, assumes the presence of nonlinear dependencies, and is related to self-organisation processes, see [23].

Computer systems of the 21-st century often exhibit explicit complexity and autonomy of components, the existence of emergent behaviour is acceptable today, as inseparable property of those systems, although the impact of emergent behaviour has to be studied diligently. Emergent behaviour may add a missing "touch of human genius" to the behaviour of embedded computer systems, if detected in due time and responded properly; see [10]. This might be extremely useful in the case of automatic decision-making, automatic analysis of potential evolution of situations, and study of impact of made decisions by simulation.

C. Generic template for a model of SoS constituent component

In order to comprehend and reason about the nation-wide situation we need to monitor the operation, and interoperation, of standing semi-autonomous enterprises, organisations and institutions, plus numerous temporary structures corresponding to missions and task forces. To reduce the complexity of the monitoring task we suggest applying a unified (and simplified) template for modelling every single object of interest. Any modelling starts from collecting information (user stories, statutes of the organizations, etc) – this stage usually provides incomplete verbal information based on interviews, and various other documents, see for instance [24]. Applying DoDAF-type modelling methodology, we get a capability-based template.



A template for functional schema of an enterprise

This template covers the interim step in developing a set of semiformal models that describes country's everyday operation, and facilitates creation, and distribution of dependable situation awareness to authorized stakeholders and enables its use for improving the management of the country.

III. Conclusion

This paper discussed an on-going project and more details regarding the models and related analysis will be published in the future papers.

Acknowledgment

The authors acknowledge the financial support from Estonian Research Agency and from the European Social Fund. The authors appreciate lively discussions within the project consortium, in particular with the Lab for Proactive Technologies, which have led to better understanding of feasible solutions to real problems.

References

[1] C.B. Keating, J.J. Padilla, K. Adams (2008) "System of Systems Engineering Requirements: Challenges and Guidelines", Engineering Management Journal, vol20, no.4, 24-31, DOI: 10.1080/10429247.2008.11431785

[2] P.W. Anderson (1972) „More is Different", Science, vol.177, no. 4047, 393-396

[3] L. Motus, K. Taveter, V. Dieves (2018) Modelling Complex System-of-Systems for Creating Situation Awareness: Late Breaking Report, Proc. 2018 IEEE Conference on Cognitive and Computational Aspects of Situation Management,168-170.

[4] Jing Zeng, Laurence T. Yang, Man Lin, Hunsheng Ning, Jianhua Ma (2016) „A survey: Cyber-physical-social systems and their system-level design methodology", Journal for Future Generation Computer Systems (2016), http://dx.doi.org/10.1016/j.future.2016.06.034

[5] Z. Liu, D. Yang, D. Wen, W. Zhang (2011) „Cyber-Physical-Social Systems for Command and Control", IEEE Intelligent Systems, vol. 26, no.4, 92-96

[6] S. De, Y. Zhou, I. L. Abad, K. Moessner (2017) „Cyber-Physical-Social Frameworks for Urban Big Data Systems: A survey", Applied Sciences, 7(10), 26pp, doi:10.3390/app7101017

[7] S.K. Sowe, K. Zettsu, E. Simmons, F. de Vaulx, I. Bojanova (2016) "Cyber-Physical Human Systems: Putting people in the Loop", IEEE Comp. Society, IT Professional, 18(1), 10-13, DOI: 10.1109/MITP.2016.14

[8] Sloman A. (2002) "The Irrelevance of Turing Machines to AI", In Scheutz, M. Ed., Computationalism: New Directions, MIT Press, Cambridge, MA, 87-127

[9] Motus L. and M.G. Rodd (1994) „Timing analysis of real-time software", Elsevier, 212pp

[10] L. Motus, J. Preden, M. Meriste, R. Pahtma (2012) "Self-aware architecture to support partial control of emergent behaviour", Proc. IEEE 7th International Conference on Systems of Systems Engineering, Genoa, 422-427

[11] Kabanikhin S.I. (2008) „Definitions and examples of inverse and ill-posed problems", J. Inverse and Ill-posed problems, vol.16, 317-357.

[12] A.R. Chaturvedi, D.R. Dolk, P.L. Drnevich (2011) "Design Principles of Virtual Worlds", Management Information Systems Quartely, vol. 35, no.3, 673-684

[13] P. Drnevich, R. Ramanujam, S. Mehta, A. Chaturvedi (2009) "Affiliation or Situation: What drives Strategic Decision-making in Crisis Response", Journal of Managerial Issues, vol. XXI, no.2, 216-231.

[14] Milner R. (1996) "Calculi for Interaction", Acta Informatica 3 (8), 707-737.

[15] Wegner P. and E. Eberbach (2004) "New models of computation", The Computer Journal, 47(1), 4-9.

[16] Goldin D., S.A. Smolka, P. Wegner (Eds.) (2006) "Interactive Computation", Springer, 488 pp.

[17] Endsley, M. R. (2015). "Situation Awareness Misconceptions and Misunderstandings." Journal of Cognitive Engineering and Decision Making 9 (1): 4–32.

[18] N.A. Stanton, P.M. Salmon, G.H. Walker, E. Salas, P.A. Hancock (2017) "State-of-science: situation awareness in individuals, teams and systems", Ergonomics, vol.60 no.4 449-466, http://dx.doi.org/10.1080/00140139.2017.1278796

[19] N.A. Stanton, P.M. Salmon, G.H. Walker, D. Jenkins (2010) "Is situation awareness all in mind", Theoretical Issues in Ergonomics Science, 11:1-2, 29-40, DOI: 10.1080/14639220903009938

[20] J. Ehala, J. Kaugerand, R. Pahtma, S. Astapov, A. Riid, T. Tomson, J. Preden, L. Motus (2017), "Situation Awareness via Internet of Things and In-network Data Processing", International Journal of Distributed Sensor Networks, vo.13(1), DOI: 10.1177/1550147716686578

[21] J. Preden, L. Motus, J. Llinas, R. Pahtma, R. Savimaa, M. Meriste, S. Astapov (2014) "Multisource Data Fusion for providing Situational Information: Improvised Explosive Devices in Asymmetric Conflicts", Chapter 14 in Case studies in System of Systems, Enterprise Systems, and Complex Systems Engineering, CRC Press (Taylor and Francis Group), ISBN 978-1-4665-0239-0, 407 – 443

[22] J. Goldstein (1999) "Emergence as a Construct: History and issues", in Emergence, 1 (1), 49-72, DOI: 10.1207/s15327000em0101_4

[23] T. DeWolf, T. Holvoet (2005) "Emergence versus Self-Organization: Different concepts but Promising When combined", Springer, Lecture Notes on Computer Science 3464, 1-15

[24] H. Bahsi, V. Dieves, T. Kangilaski, P. Laud, L. Motus, J. Murumets, I. Ploom, J. Priisalu, M. Seeba, E. Taks, K. Tammel, P. Tammpuu, K. Taveter, A. Trumm, T-T. Truusa, T. Vihalemm (2019) "Mapping the Information Flows for the Architecture of a Nation-Wide Situation Awareness System", accepted IEEE CogSIMA 2019 conference

[25] G. Che and Jin Y (2009) "Online co-design of feedback control and real-time scheduling for embedded systems with communication delays." Proc. 2009 IEEE Conference on Computer Science & Education, 618-623

[26] L. Motus (2007) "Modelling metric time", Ch.10 in "UML for real: Design of Embedded Real-Time Systems", Eds. L. Lavagno, G. Martin, B.V. Selic, Springer Science &Business, 205 -220.

Curriculum Vitae

1. Personal data

Name	Jaanus Kaugerand
Date and place of birth	29 December 1976 Tartu, Estonia
Nationality	Estonian

lorem ipsum

2. Contact information

Address	Tallinn University of Technology, School of Information Technologies, Department of Software Science, Ehitajate tee 5, 19086 Tallinn, Estonia
Phone	+372 620 2109
E-mail	jaanus.kaugerand@taltech.ee

3. Education

2014–2020	Tallinn University of Technology, School of Information Technologies, Course name, PhD studies
2012–2014	Tallinn University of Technology, Faculty of Information Technology, Computer and Systems Engineering, MSc *cum laude*
2009–2012	Tallinn University of Technology, Faculty of Information Technology, Computer and Systems Engineering, BSc

4. Language competence

Estonian	native
English	fluent
Danish	fluent
Russian	conversational
Finish	conversational

5. Professional employment

2017– . . .	Tallinn University of Technology, Early Stage Researcher
2014–2016	Tallinn University of Technology, Engineer
2011–2014	IB Krates, Software developer
2006–2009	Estonian Navy, ENS Admiral Cowan, Operations Officer
2005–2006	Estonian Navy, Navy Headquarters Operations Department, Junior Staff Officer
2004–2005	Estonian Navy; ENS Wambola, Navigation Officer

8. Honours and awards

- 2015 Best Student Paper Award, "A System of Systems Solution for Perimeter Control: Combining Unmanned Aerial System with Unattended Ground Sensor Network" 9th Annual IEEE International Systems Conference. Vancouver, British Columbia, Canada
- 2014 Estonian Ministry of Defence Research Award

- 2014 ICT student master thesis pre-defence contest, Master category, I place

9. Defended theses

- 2014, A System of Systems Solution for Perimeter Control: Combining Unmanned Aerial System with Unattended Ground Sensor Network, MSc, supervisor Dr. Jürgo Söeren-Preden, Tallinn University of Technology, Department of Computer Control
- 2011, Model-based PLC software testing engine, BSc, supervisor MSc Tõnu Näks, Tallinn University of Technology, Department of Computer Control

Elulookirjeldus

1. Isikuandmed

Nimi	Jaanus Kaugerand
Sünniaeg ja -koht	29.12.1976, Tartu, Eesti
Kodakondsus	Eesti

2. Kontaktandmed

Aadress	Tallinna Tehnikaülikool, Infotehnoloogia teaduskond, Tarkvarateaduse instituut, Ehitajate tee 5, 19086 Tallinn, Estonia
Telefon	+372 620 2109
E-post	jaanus.kaugerand@taltech.ee

3. Haridus

2014–2020	Tallinna Tehnikaülikool, Info- ja kommunikatsioonitehnoloogia, doktoriõpe
2012–2014	Tallinna Tehnikaülikool, Info- ja kommunikatsioonitehnoloogia, Arvutisüsteemid, MSc *cum laude*
2009–2012	Tallinna Tehnikaülikool, Info- ja kommunikatsioonitehnoloogia, Arvutisüsteemid, BSc

4. Keelteoskus

eesti keel	emakeel
inglise keel	kõrgtase
taani keel	kõrgtase
vene keel	kesktase
soome keel	kesktase

5. Teenistuskäik

2017– ...	Tallinna Tehnikaülikool, doktorant-nooremteadur
2014–2016	Tallinna Tehnikaülikool, insener
2011–2014	IB Krates, programmeerija
2006–2009	Eesti Merevägi, EML Admiral Cowan, operatsioonide ohvitser
2005–2006	Eesti Merevägi, Mereväestaabi operatiivsektsioon, nooremstaabiohvitser
2004–2005	Eesti Merevägi; EML Wambola, navigatsiooniohvitser

8. Autasud

- 2015 Parim tudengiartikkel - 1 koht "A System of Systems Solution for Perimeter Control: Combining Unmanned Aerial System with Unattended Ground Sensor Network" 9th Annual IEEE International Systems Conference. Vancouver, British Columbia, Canada
- 2014 Eesti Vabariigi Kaitseministeeriumi teaduspreemia
- 2014 IKT tudengite lõputööde eelkaitsmiskonkurs, Magistri kategooria, I koht

9. Kaitstud lõputööd

- 2014, Autonoomne mehitamata lendav süsteem osana taktikalisest süsteemide süsteemist, MSc, juhendaja Dr. Jürgo Söeren-Preden, Tallinna Tehnikaülikool, Automaatika Instituut
- 2011, Mudelipõhine PLC tarkvara testimise mootor, BSc, juhendaja MSc Tõnu Näks, Tallinna Tehnikaülikool, Automaatika Instituut