TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Karl Matti 166157

AUTONOMOUS ROBOT SHIP – DETECTING OBSTACLES WITH MILLIMETRE RADAR

Bachelor's thesis

Supervisor: Heigo Mõlder
PhD
Co-Supervisor: Gert Kanter
MSc

Tallinn 2020

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Karl Matti 166157

AUTONOOMNE ROBOTLAEV – TAKISTUSTE TUVASTAMINE MILLIMEETERLAINE RADARI ABIL

bakalaureusetöö

Juhendaja: Heigo Mõlder
PhD
Kaasjuhendaja: Gert Kanter
MSc

Tallinn 2020

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Karl Matti

18.05.2020

Abstract

The goal of thesis is to find solution on prototype level for robot ship NYMO that detects obstacles on the sea. Thesis starts with an overview of modern but affordable millimetre wave radars and comparison of similar radars to find the most suitable result that can be applied on the robot ship. Then, work continues with capturing reflected points by radar where for detecting objects, various grouping algorithms are investigated to remove noise signals from the real ones. The result of this thesis is a software which can detect data points with radar and share distances of closest object in each sector with the ship.
This thesis is written in English and is 59 pages long, including 5 chapters, 42 figures and 2 tables.

Annotatsioon

Autonoomne robotlaev – takistuste tuvastamine millimeeterlaine radari abil

Lõputöö eesmärk on leida prototüübi tasandil lahendus robotlaevale NYMO, et tuvastada takistusi merel. Lõputöö algab ülevaatega kaasaegsetest, ent taskukohase hinnaga millimeeter laine radaritest ja erinevate samatüübiliste radarite võrdlusest, et leida nende hulgast sobivaim lahendus, mida robotlaevale külge panna. Seejärel jätkub töö radari poolt tagasi peegelduvate andmepunktidega, kus objektide äratundmiseks uuritakse erinevaid grupeerimisalgoritme, et mürasignaale olulisest eemaldada. Lõputöö tulemuseks on tarkvara, mis oskab millimeeterlaine radariga andmepunkte tuvastada ja jagada laevale lähimate takistuste distantse sektorite vaates.
Lõputöö on kirjutatud inglese keeles ning sisaldab teksti 59 leheküljel, 5 peatükki, 42 joonist, 2 tabelit.

List of abbreviations and terms

ACC	Autonomous Cruise Control
FMCW	Frequency-Modulated Continuous-Wave
MuRPS	Multi-channel Radar for Perimeter Surveillance
FoV	Field of view
RX	Receive channel
TX	Transmit channel
SDK	Software Development Kit
AIS	Automatic Identification System
ROS	Robot Operating System

Table of contents

List of figures

List of tables

1 Introduction

In the current thesis headline, the words “autonomous robot ship” gives this work a main substance meaning. Autonomous robot ship is meant as a ship that can operate without needing a crew [1].
Currently developed ship named NYMO (Figure 1) [2] is a full electric emission free autonomous swimming vessel as described in [3].

Figure 1. Autonomous ship NYMO

It is 2560 mm long and 1100 mm wide which has maximum payload 100 kgs. The vessel could be used for mapping the seabed, delivering parcels, conducting rescue operations or detecting pollution as stated in [4].
The ship is currently using ultrasonic sensors and maritime Automatic Identification System for navigation at the sea and in the harbour as described in [4]. As Indrek Roasto, the lead of robot ship team stated, the LIDAR is too expensive and cannot see anything in foggy weather. In the same time millimetre wave radar is static and sees with any weather.
Considering that different millimetre wave radars are available and different software have been made for detecting objects, then best solution for the problem was not implemented yet. Problem that this work is going to solve is in two parts. First part was selecting a suitable millimetre wave radar for the ship and make sure that radar has a possibility to work with captured data points programmatically in the ship’s controller. Second part that this work solved was applying various clustering algorithms for data points to see which one helps the ship make better navigation decisions in order to not collide with obstacles. Also, the ship’s current sensors are working with seeing sectors which means that the program has to say in which way and how far an obstacle is.
Thesis is divided in three bigger parts. First one is an overview of a millimetre wave radars and comparisons of various millimetre wave radars which were used to make a purchasing decision. Second part is about different grouping methods that were used for grouping data points like Mini Batch K-Means, K-Means, DBSCAN, OPTICS and a

custom boxing method. Third and final part is about the software developed and scenarios that were used to develop the software. The software is used to capture data from radar, save the data, group data points and display them on 2D scatter plot for a better visualization.

The method that these clustering methods could be applied the radar had to go through a pack of scenarios which data were saved in a file. Different type of scenarios was chosen in order to see how radar collects data and how algorithms work in dissimilar situations. Later the data from the file was used to re-live the capturing moment. Then different kind of algorithms were applied for the same data points to validate which algorithm works the best.

2 Millimetre radars

As stated in [5], the millimetre waves are in an electromagnetic spectrum between 30 to 300 GHz where wavelengths are from 10 to 1 mm.
According to [6], first time were automotive radars tested in the end of 50's. Then, in 70's the development of radars started at microwave frequencies when in last decades are concentrated mainly at 17 GHz, 24 GHz, 35 GHz, 49 GHz, 60 GHz and 77GHz.

2.1 Millimetre radar working principles

The working principles of the millimetre wave radar is described in [7]. Where millimetre wave radar is compared with a pulse radar system. Firstly, pulsed millimetre waves are transmitted to the environment. Secondly, the waves are reflecting by an object and received. The time between first two steps are used to measure the distance to the object. If the object is moving, then the frequency of the received waves is different, and the speed can be calculated from the frequency difference.
As described in [8], the millimetre wave radar has 3 fundamentals: range measurement, velocity and angle. The simplified block diagram (Figure 2) of the main components operates in following order:

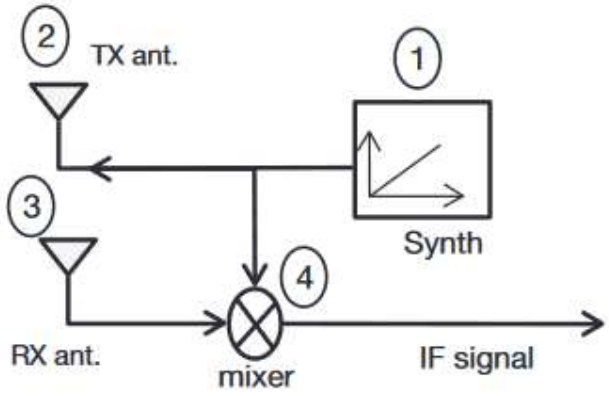


Figure 2. FMCW radar block diagram.

1. A synthesizer creates a signal
2. A transmit antenna sends out the chirp
3. A receive antenna receives a reflected signal from an object
4. A mixer produces an IF signal from sent and received chirps

As described in [9], the millimetre wave radars are accurate. For example, 24 GHz radars can have resolution with 75 cm of accuracy and 77 GHz about 25 cm accuracy for the same active chirping time.
As stated in [5], ultrasonic sensors performance is poor in bad weather, the only viable alternative would be millimetre wave radars or laser sensors. Laser sensors compared with millimetre wave sensors (Table 1) [5] shows how laser is better in

tracking accuracy, classification and imaging while millimetre wave radar works better in volume search and adverse weather.

Table 1. Sensor System Performance Comparison

Sensor Characteristic	**Millimetre wave**	**Laser**
Cost	Poor	Good
Tracking accuracy	Fair	Good
Classification	Fair	Good
Imaging	Fair	Good
Volume search	Fair	Poor
Adverse weather	Fair	Poor
Smoke, dust etc.	Poor	Poor
Dirty Antenna	Poor	Poor

2.2 Examples of millimetre radars in world

According to [6], the first ACC (Autonomous Cruise Control) was commercialized in Japan in 1995 and [10], the first radar-based ACC was introduced by Mercedes-Benz in 1999. As stated in [11], addition to ACC, automatic braking is also used to support driving. What it means, is that the radar captures obstacles on the road and then calculates the time till the collision which is used in system level decisions.
Millimetre wave radars have many applications on other robots. As described in [12], the robots can be controlled using humans hand gestures. Although the human should be on site to control the robots, the controlling can be useful in different situations. Also, detecting other robots with the radar, like Micro Aerial Vehicles is a good approach for perimeter surveillance as stated in [13]. The MuRPS (Figure 3) is a radar working on 94 GHz which has produced reliable results when doing perimeter surveillance.

Figure 3. Photograph of the radar front end of the MuRPS.

According to [14], the millimetre wave radars can be used to monitor vital signs. They collected heartbeat and respiration patterns with prototype from animals and people.
Millimetre wave radars are also successful in scanning and imaging. As described in [5], the millimetre radar can be used for mechanical scanning, electronic scanning and producing two-dimensional images (Figure 4) that are similar to aerial photographs. Those images can be used to predict features that most likely are also on aerial photographs. Images created are reflecting smooth horizontal surfaces which appear darker and roads are lighter due to the corner effects. Big buildings and trees are also reflecting dark because they are considered as smooth surfaces.

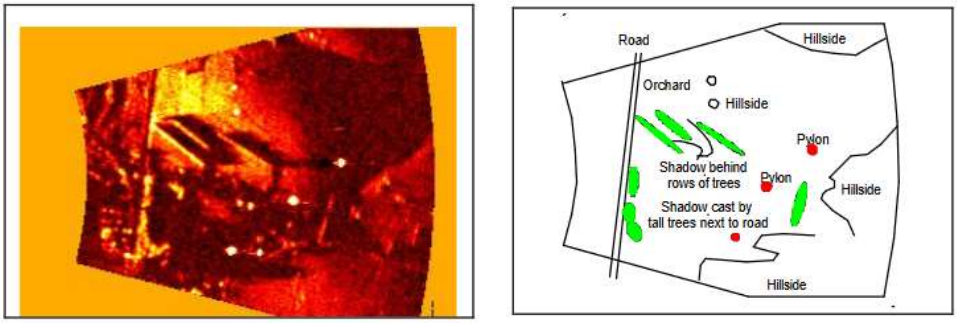


Figure 4. Real Aperture millimetre wave Image

2.3 Hardware selection criteria and technical comparison of different platforms

For the robot ship an optimal millimetre wave radar was intended to be bought. Before buying a comparison of different available millimetre wave radars had to be done. The criteria's that were compared were following: frequency of the radar working on, maximum range distance the radar can see, width range, price and available software.
Robot ship team requested a millimetre wave radar whose maximum object detecting range is between 50 to 250 meters. Since the radar is going to be added to robot ships main controller in the future, the software for the radar should be in Python programming language. Since the program for the radars may have not be done in

Python yet then a possibility to do it was also investigated. Then, after requirements were filled, the price was compared to have a best quality and price relationship. Finally, when requirements and price did not help to decide, the release date was compared in favour of newest radar available.

2.3.1 Technical comparison of different millimetre wave radars

Comparison table (Table 2) of previously collected data is created from comparisons (see Appendix 4 – Comparison of different radars). Different valuable attributes for robot ship according its team was evaluated. The most important attributes were max detection range in meters, price and possibility to create software or use existing for detecting objects at the sea. By considering different values that came up with the research the AWR1843BOOST was selected the most suitable and bought by the robot ship team. AWR1843BOOST had the best price and quality relationship where quality values were amount of different information available along with matching important attributes with robot ship team requests. The deciding advantage for AWR1843BOOST before AWR1642BOOST is that the release of AWR1843BOOST key document is 2019 August [15] which is newer than AWR1642BOOST key document release which is 2018 February [16].

Table 2. Technical comparison of radars by different attributes

Sequence	Name of the device	Small picture	Frequency in GHz	Max detection range in meters	Detection angle in degrees	Price in USD	Software or applications for exact radar
1	AWR1843-BOOST		76-81	150	+- 50 horizontal and +- 20 elevation	299	mmWave studio and mmWave SDK
2	AWR1642-BOOST		76-81	160	-	299	mmWave studio and mmWave SDK
3	AWR1443-BOOST		76-81	120	-	299	mmWave studio and mmWave SDK
4	AWR1243-BOOST		76-81	160+	-	299	mmWave studio and mmWave SDK
5	OPS243		24	1-100	+- 10 horizontal	209	Four applications have been made for the sensor: Traffic Monitoring, Drone Collision Avoidance, Robotic Control and IoT Sensor
6	IWR6843ISK on MMWAVEIC-BOOST carrier		60-64	114.43	+- 54 horizontal and +-22 elevation	324	mmWave SDK
7	Pixhawk V3		24	100	+- 14 horizontal and +- 9 elevation	359	Software which shows plot with detected objects and distance from them
8	ARS408-21		77	0.2 - 250	up to 70 m +- 60 horizontal and +- 20	844	Comes with software which can detect pedestrians and vehicles.

					elevation, up to 250 m +- 9 horizontal and +- 14 elevation		
9	Sensor Fusion Kit		76-81	-	-	1499	Comes with software which can detect different objects and different customization in software's.
10	AoPCB Module		60-64	-	-	120	-

2.4 Hardware connection schema

AWR1843BOOST and Raspberry Pi 3 B+ model was connected (Figure 5) by a supervisor Heigo Mõlder in a box (Figure 6). Raspberry Pi 3 B+ model was chosen because of its popularity and availability which was offered by the robot ship team.
The millimetre wave radar AWR1843BOOST is connected with Raspberry Pi 3 B+ model using Micro USB to USB-C wire. While Raspberry Pi is powering the radar with 5 V, the radar can send data to Raspberry Pi. The data contains number of objects detected, every object x, y and z distance from radar and velocity from radar. Power supply used for the Raspberry Pi is a power bank which gives 5 V with 2100 mA.
A SSH connection through PuTTY [17] is made in a laptop to have an access to Raspberry Pi's command line. That command line is used for activating scripts that will start capturing data with the radar and WinSCP [18] to transfer the data from Raspberry Pi to a laptop. Laptop and Raspberry Pi must be in the same network in order to have a SSH connection. Development environment is in a laptop and Git [19] is a version control system that is used to transfer the changes in a software. Git environment used to store the changes is in TalTech Gitlab [20].
When the radar is activated then it is starting to send out and receive signals like described in chapter 2.1 which will determine the captured data.

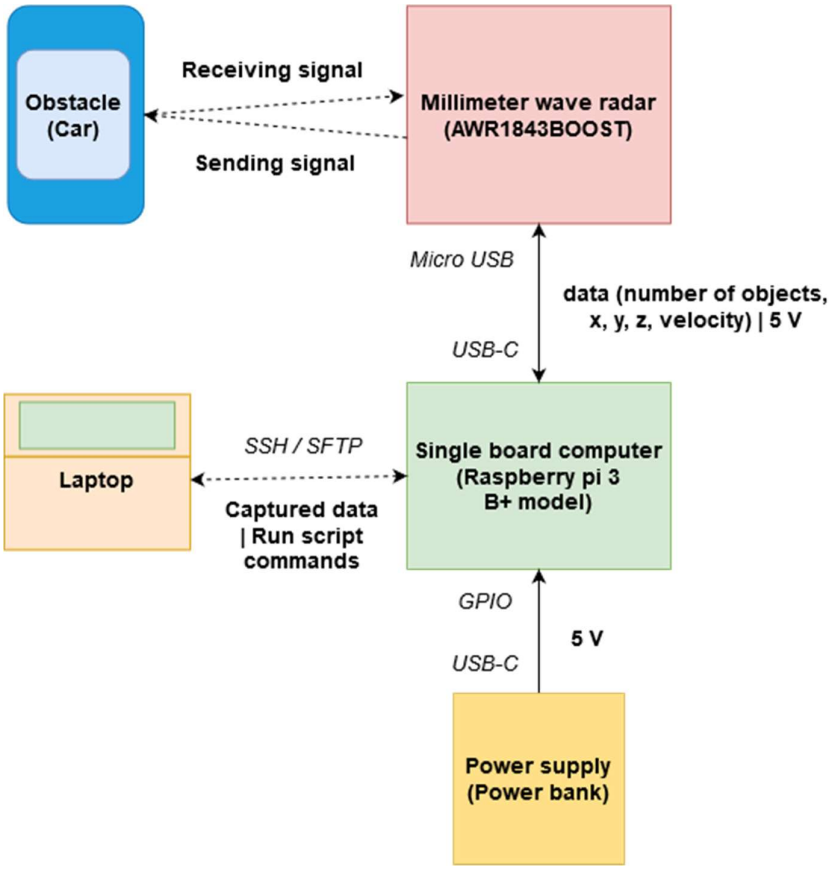


Figure 5. Block schema of connections between obstacle, millimetre wave radar, Raspberry Pi 3 B+ model, power supply and a laptop

Figure 6. AWR1843BOOST and Raspberry Pi 3 B+ in box connected with power bank output 5 V, 2100 mA

2.5 Radar and controller's software

After selecting AWR1843BOOST as the radar, the existing software for controlling it had to be researched and tested. Radars official site refers to mmWave SDK [21] for the development which has out-of-box demos to see how the radar works.
Since the one version of software in Python programming language to read data from AWR1843BOOST and display all the points in 2D scatter plot is made and published in [22] then it was investigated further. Lastly mentioned software could communicate with the radar from Raspberry Pi which was robot ship's criteria. Also, the software would be more lightweight than the official SDK for applying it on a small controller since it has much less possibilities on it.
The mmWave Sensing Estimator [23] is an online application for calculating radar's configuration based on scene parameters. This is used to select AWR1843 sensor and long-range default configuration to replace the unknown configuration. It was used in previously described software in Python to test and implement suitable software from it.
Since the software did not fill the aim of detecting obstacles for the ship entirely then a few parts of the software had to be implemented.
Firstly, single data points had to be saved with a timestamp to a csv file after capturing it in order to replay scenarios (see Scenarios). Secondly, data from saved scenarios should have possibility to play them back on 2D scatter with the correct timing.
Thirdly, single data points had to be grouped for having less noise and having more actual information about obstacles. Then, the made software is not ready to tell the robot ship about obstacles in the right format. Since robot ship works with angles then the viewing sector, which is roughly 120 degrees, must be divided to five sectors each 24 degrees wide. The distance from the closest object in each sector should be returned in list format.
Finally, since the Raspberry Pi is not the main controller, it should send the data over TCP. The listener must be implemented for the main controller and client must be implemented for the Raspberry Pi which is getting the values from the radar.

3 Clustering methods for identifying obstacles

Our millimetre wave radar AWR1843BOOST is detecting single data points which are obstacles for it. The aim is to warn ship with obstacles positions, the horizontal angle from radar and distance between them. To fulfil this aim, different grouping methods were investigated and applied. As stated in [24], commonly unsupervised clustering algorithms are used to fill this aim but these usually fail to identify all the points that belong to the same obstacle. In this work the focus is on unsupervised clustering.
There are many clustering algorithms available to use (Figure 7) [25] like Mini Batch K-Means, K-Means, Affinity propagation, Mean-shift, Spectral clustering, Ward, DBSCAN, OPTICS and etc. As the (Figure 7) shows different clustering algorithms applied on 2D datasets then the third and fifth row are the type of datasets that are the most similar to captured data. From these different type of clustering methods was applied and investigated like K-Means clustering, Mini Batch K-Means clustering, DBSCAN clustering, OPTICS clustering and a custom boxing method.

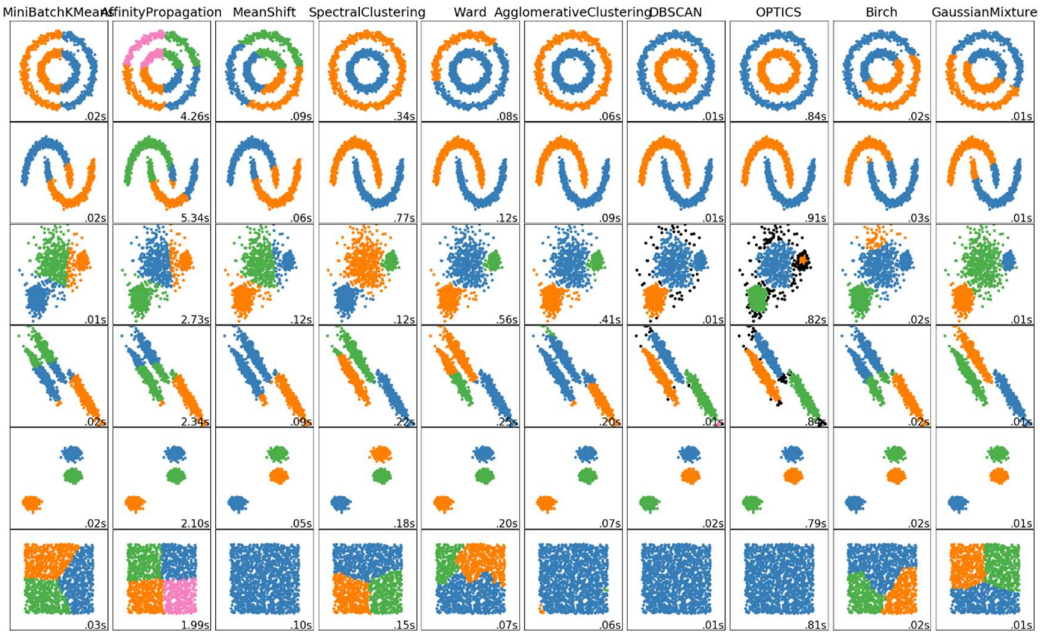


Figure 7. Characteristics of different clustering algorithms on 2D datasets

All the scenarios (see Scenarios) were not saved for the purpose of applying grouping methods on them. Scenarios with walls and cars have been made for applying the grouping algorithm.

3.1 K-Means clustering

K-Means clustering takes in as a parameter the number of clusters as stated in [25]. Its use case is general purpose and flat geometry which gave it a naive idea to apply it on the captured data. Metric for the algorithm is distances between points.
Since the radar is showing noise in 5 m radius, obstacle data points and noise points then cluster number was chosen to be 2. In real life, more than two obstacles can be in

front of robot ship and with the first examination, the cluster number should be calculated dynamically.
The plot (Figure 8) was saved when radar was approaching car (see Scenarios). It shows that some noise points in the upper left corner are separated from real data points and other half are not. Also, the plot shows how data points from car relate to noise what is in front of the radar in 3 m radius.
The plot (Figure 9) of radar approaching the wall (see Scenarios) with K-Means algorithm applied between points has results. Multiple lines referring to the bottom middle points are showing that the wall data points are combined with noise that radar detects.
Since the data points are not correctly connected during the important scenes when approaching the car and wall then this algorithm is not good with given parameters. Also, it is hard to determine how many clusters can the plot have since the radar does not know how many objects there are.

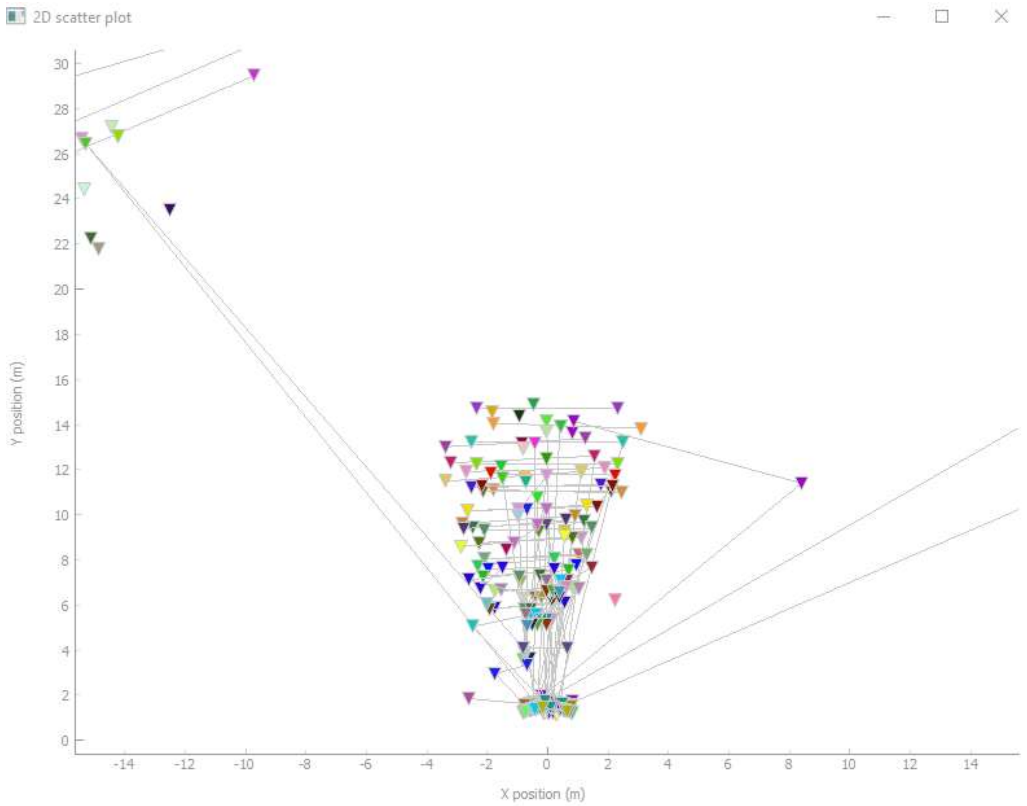


Figure 8. Frames-1012.csv where K-Means is applied, and plot is not cleared after new data came in. One cluster is with the same colour of triangles connected with grey lines. Different clusters are demonstrating how accurate the algorithm with these datasets is.

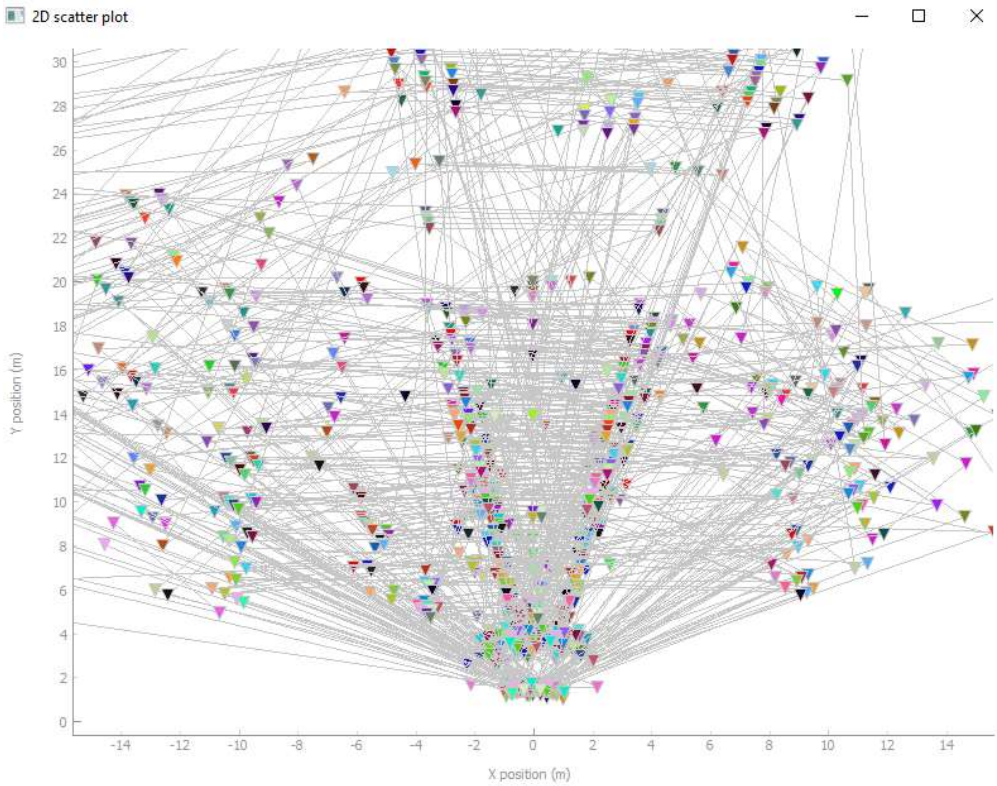


Figure 9. Frames-1103.csv of saved data points when walking towards the wall and applying K-Means algorithm. One cluster is with the same colour of triangles connected with grey lines. Different clusters are demonstrating how accurate the algorithm with these datasets is.

3.2 Mini Batch K-Means clustering

Since the Mini Batch K-Means is faster and gives slightly different results than regular K-Means as described in [25] then it was also applied to see difference in results. Mini Batch K-Means was also examined with cluster number parameter being number 2 to have a better comparison with the regular K-Means clustering.

Differences between Mini Batch K-Means and K-Means is difficult to spot. Also, it is not important because both were not effective due to inaccurate clusterings. As shown in the first plot (Figure 10) when approaching the car from 15 meters then the plot is not changed much more when using regular K-Means (Figure 8). Same results are when approaching the wall (Figure 11) with Mini Batch K-means and with regular K-Means (Figure 9).

Since the Mini Batch K-Means did not change the results when compared to K-Means results then this algorithm was not selected also. Also, it is hard to determine how many clusters can the plot have since the radar does not know how many objects there are.

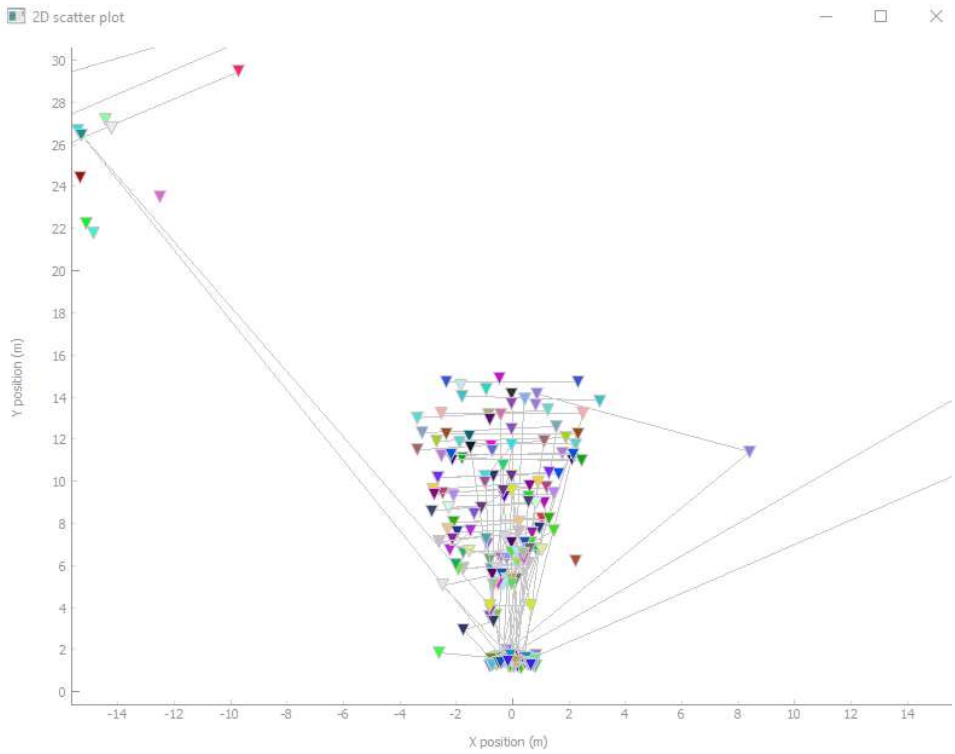


Figure 10. Frames-1012.csv where Mini Batch K-Means is applied, and plot is not cleared after new data came in. One cluster is with the same colour of triangles connected with grey lines. Different clusters are demonstrating how accurate the algorithm with these datasets is.

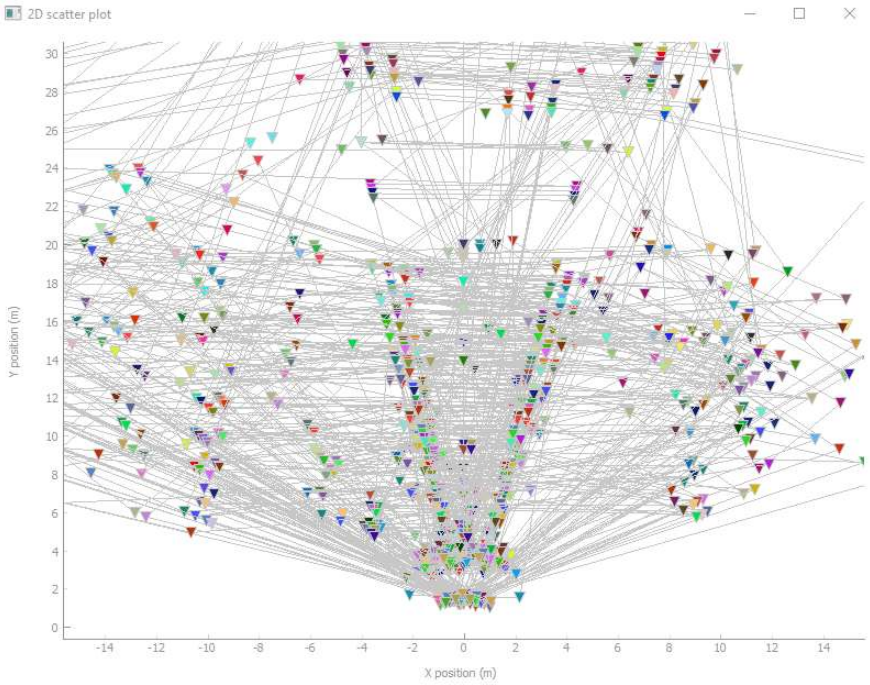


Figure 11. Frames-1103.csv of saved data points when walking towards the wall and applying Mini Batch K-Means algorithm. One cluster is with the same colour of triangles connected with grey lines. Different clusters are demonstrating how accurate the algorithm with these datasets is.

3.3 DBSCAN clustering

DBSCAN is an acronym of Density-Based Spatial Clustering of Applications with Noise as described in [26]. DBSCAN type of clustering takes neighbourhood size as a parameter and for geometry the distance between nearest points is used as stated in [27]. Use case for the clustering method is uneven cluster sizes which is suitable because obstacles can be different sizes.
DBSCAN was used with *eps* parameter being 4 which is maximum distance between two data points in order to group them together. Also, *min_samples* parameter was set to 2 which determines the number of data points needed to create a cluster.
Dataset which was coming from radar was fitted to the DBSCAN grouping method. Each datapoint had value beginning from -1 which was considered noise and 0 and above as clusters where each number represents different cluster. Data was displayed on plot and was visually compared (Figure 12) how accurate the method is with playing actual video of the scenario the radar was going through.

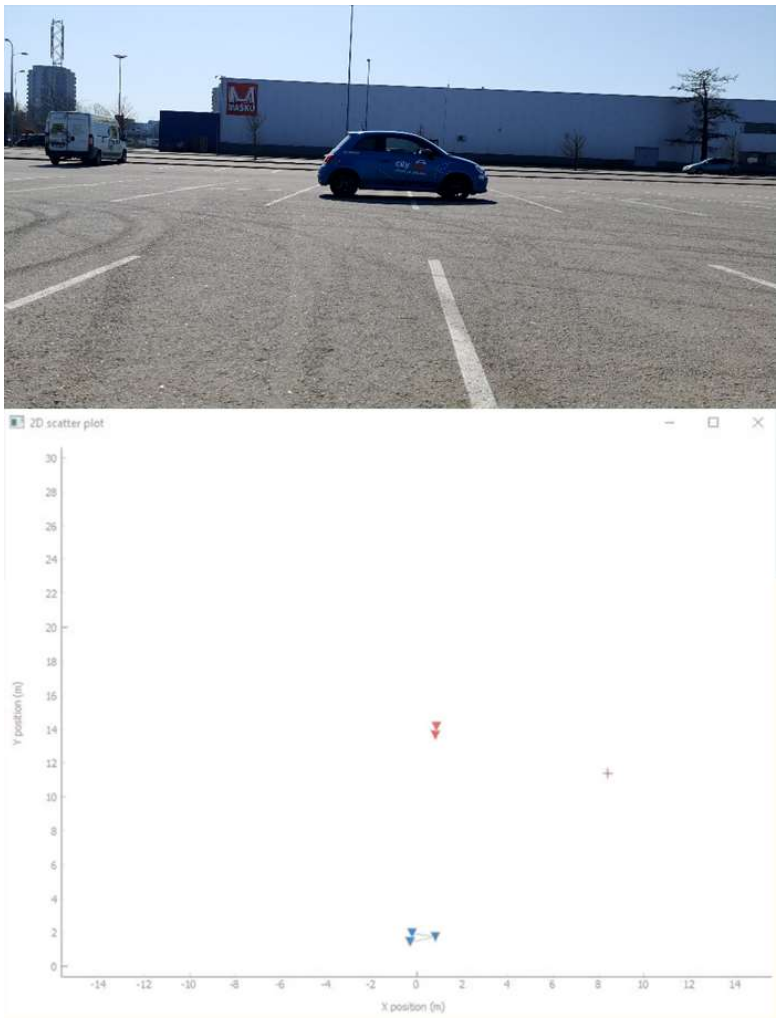


Figure 12. Video of surroundings and 2D scatter plot when radar faces car from 15 m +-1 m from frames-1012.csv. DBSCAN algorithm is applied for data points. Noise is + symbol and clusters are different colours and lines are drawn between them.

Better understanding of the results after applying the algorithm comes when the plot (Figure 13) is not cleared after new data comes in. The plot has clusters where triangles with same colours and lines drawn between them are one clusters and "+" symbols are marked as noise by the algorithm. The plot shows that roughly in 15 m to 10 m the radar could not detect the car much because the distance between the points are more than 4 meters from each other to create a cluster.

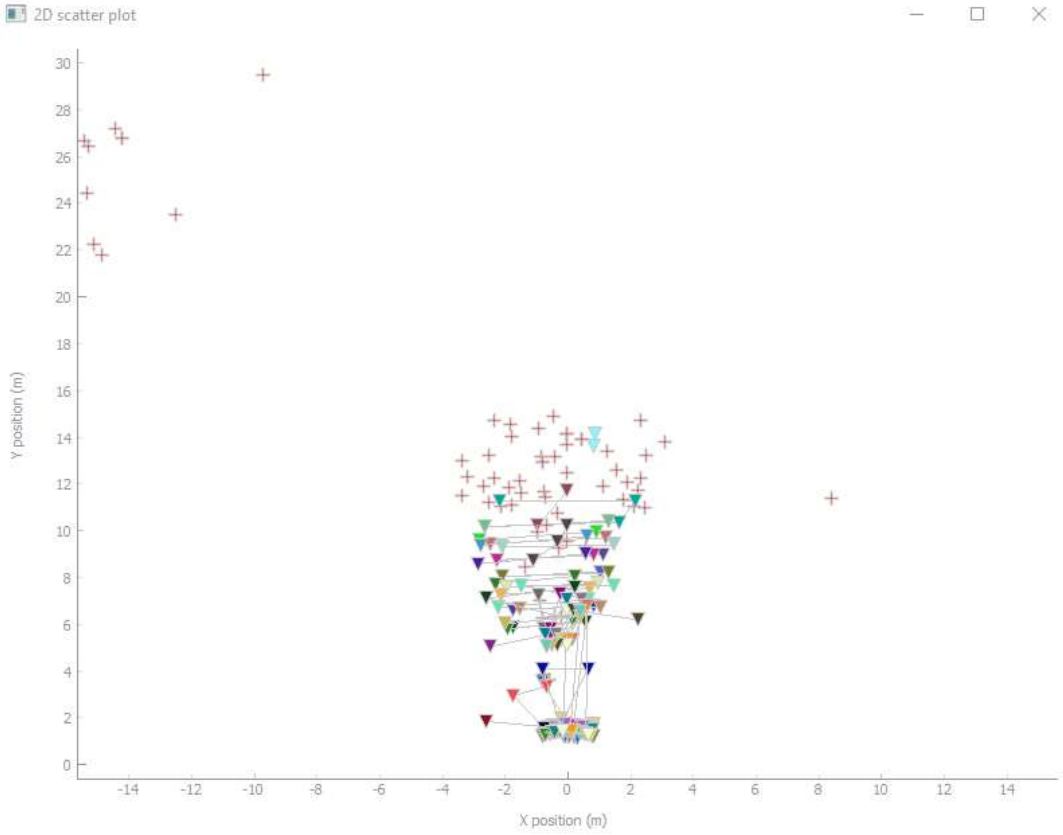


Figure 13. Frames-1012.csv where DBSCAN is applied, and plot is not cleared after new data came in. Noise is + symbol and clusters are different colours and lines are drawn between them.

Since the plot (Figure 14) in scenario number 2 (see Scenarios) when approaching the wall have data points more farther then the DBSCAN neighbour distance could not be smaller. Also, the maximum neighbour distance could not be much bigger because the field is only 50 m^2 and the aim is not to have one big cluster of all the data points.
Since the radar does not know how many clusters there will be, the DBSCAN method was better than K-Means clustering's. As it is hard to say how big the obstacles can be then the distances between the points as a parameter is not good to give as constants. This clustering method may be suitable in the future when better obstacle sizes can be determined, or more densely placed data points are placed on the plot. Right now, the grouping method does not satisfy the requirements.

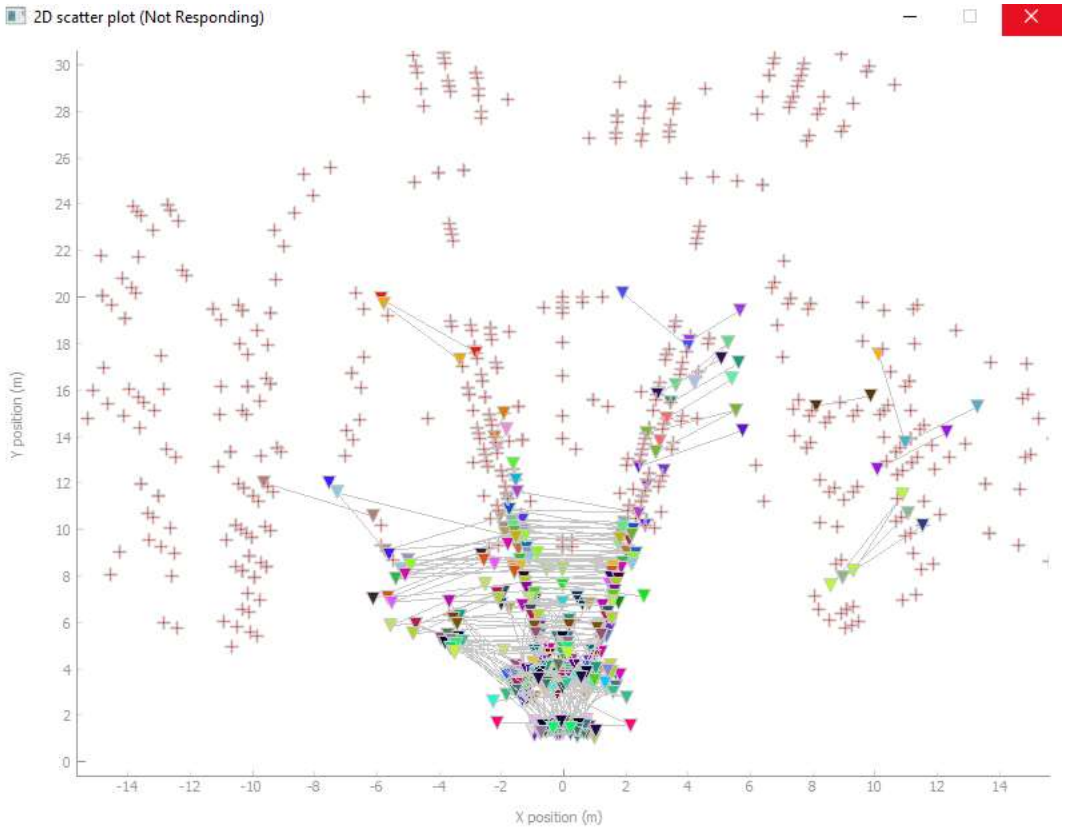


Figure 14. Frames-1103.csv of saved data points when walking towards the wall and applying DBSCAN algorithm. Noise is + symbol and clusters are different colours and lines are drawn between them.

3.4 OPTICS clustering

OPTICS is an acronym of Ordering Points To Identify the Clustering Structure as stated in [28]. The OPTICS clustering method is related to DBSCAN but unlike it, the OPTICS keeps cluster hierarchy for a variable neighbourhood radius. The clustering is better for using it on larger datasets than the DBSCAN clustering method. The implementation that sklearn offers is not original. It does perform k-nearest-neighborhood searches to identify core sizes and then computes distances only to the unprocessed points when putting together the clusters. The time complexity for the algorithm is $O(n^2)$.

When the OPTICS is applied to the same car scenario datasets (Figure 15) then a bit different results came out. With OPTICS, less noise was discovered, and car object was merged with noise which always occurs in front of the radar in 5 m radius.

On the other hand, when going through the same wall scenario as DBSCAN then results were better. Since the OPTICS calculates distances for cluster itself, it managed to detect wall as single obstacle and detect noise in front of the radar in 5 m radius as separate. Results are not perfect as plot (Figure 16) shows since it does not always cluster perfectly, and wall is often combined as one object with the noise.

Since the radar does not know how many clusters there will be, the OPTICS method parameters were better than K-Means clustering's. Since as the results show that noise is merged with real obstacle data points then this grouping method cannot be trusted with given parameters and will not fill the aim of this thesis.

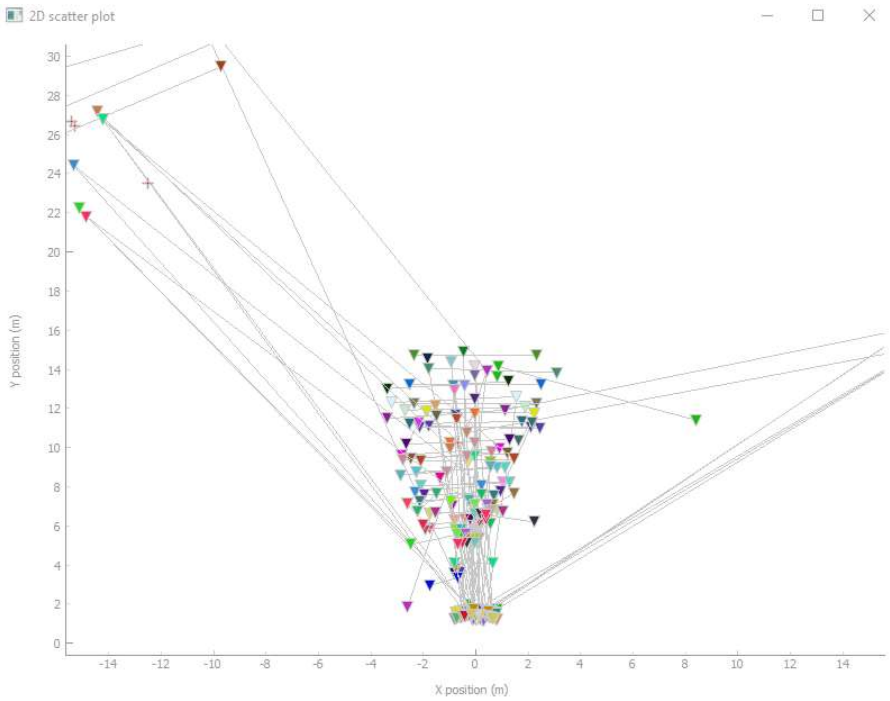


Figure 15. Frames-1012.csv where OPTICS is applied, and plot is not cleared after new data came in. Noise is + symbol and clusters are different colours and lines are drawn between them.

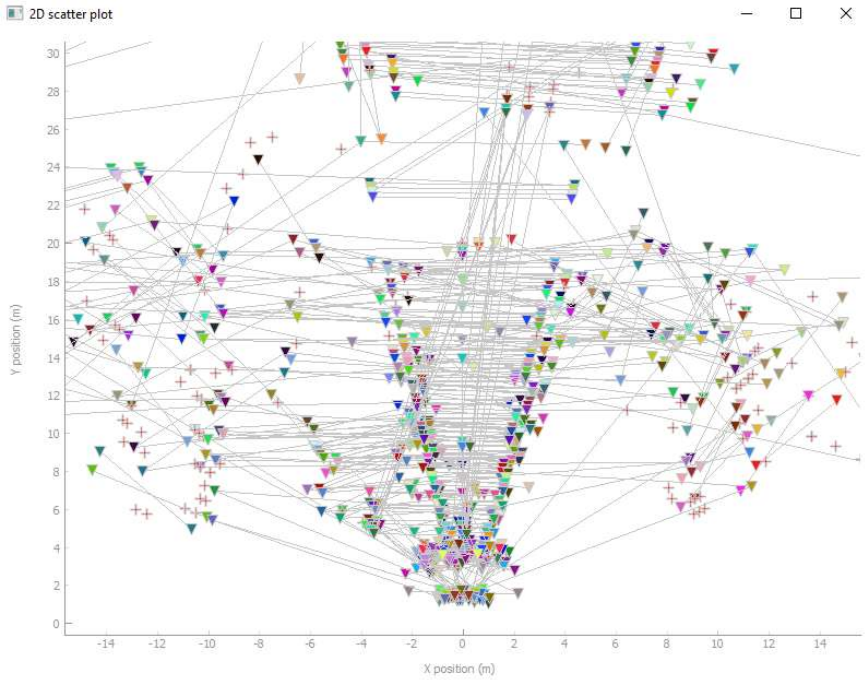


Figure 16. Frames-1103.csv of saved data points when walking towards the wall and applying OPTICS algorithm. Noise is + symbol and clusters are different colours and lines are drawn between them.

3.5 Custom boxing method

Custom boxing method is a grouping method which idea was generated by the robot ship team in a meeting. Currently tested radar viewing field size is N meters times N meters where N value is x-axis and y-axis length in meters. The algorithm (Figure 17) divides it to N pieces of 1 m^2. When data points are occurring in one of the squares then it starts to remember it for t seconds where t value is the parameter which says how long the occurrence should be remembered. When occurrence in 1 m^2 is bigger than c times then the information is passed to the ship where c is occurrence count in one square meter for t seconds.
The viewing sector is naively taken slightly wider than the actual viewing angles. The viewing sector in algorithm (Figure 18) is 120 degrees horizontally which is divided to 5 smaller sectors. An array of 5 values holds distances of closest squares. When the occurrence count is big enough then the square information is passed to a function which calculates its distance from radar and checks if it is the closest object in the sector, if yes then the array of sector's closest values is updated.

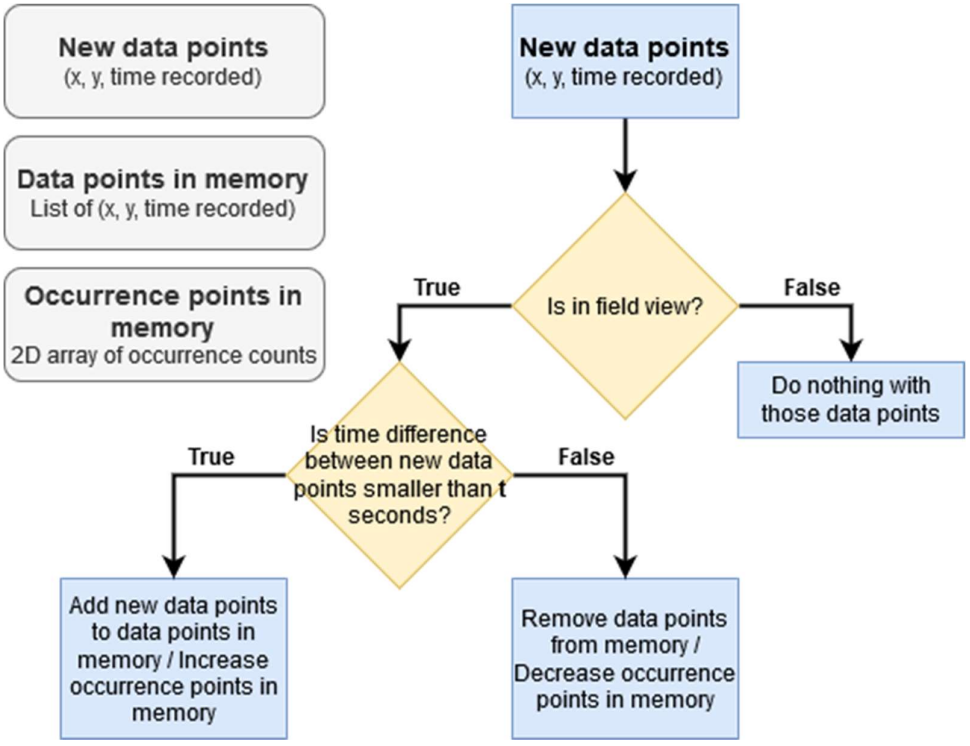


Figure 17. Saving new data points and updating data points and occurrence points in memory.

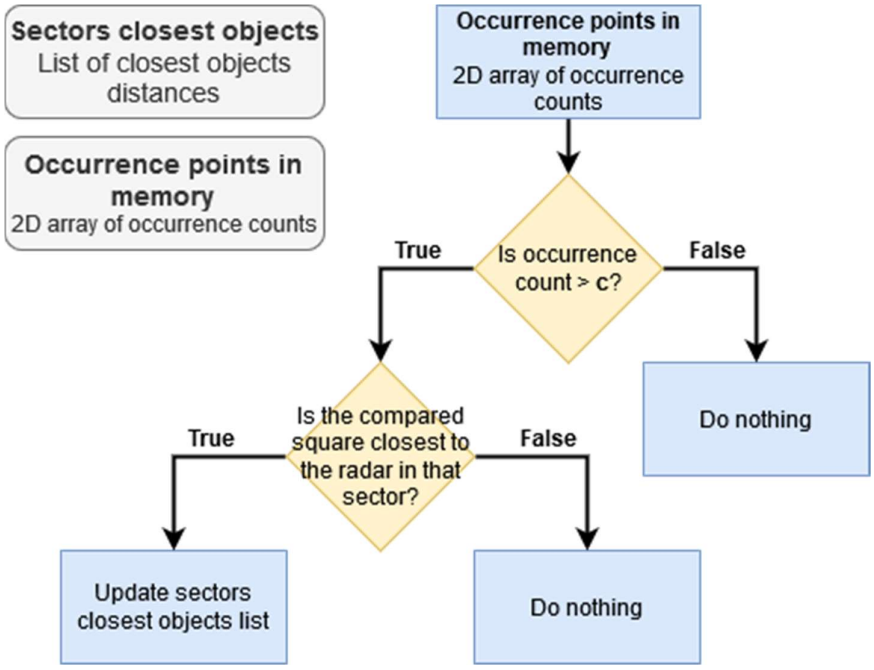
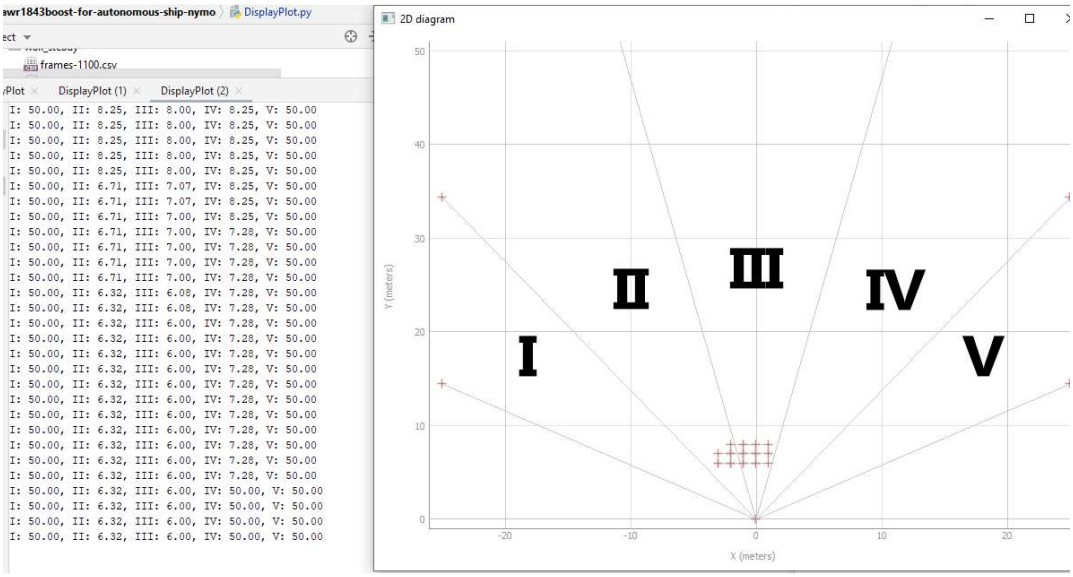


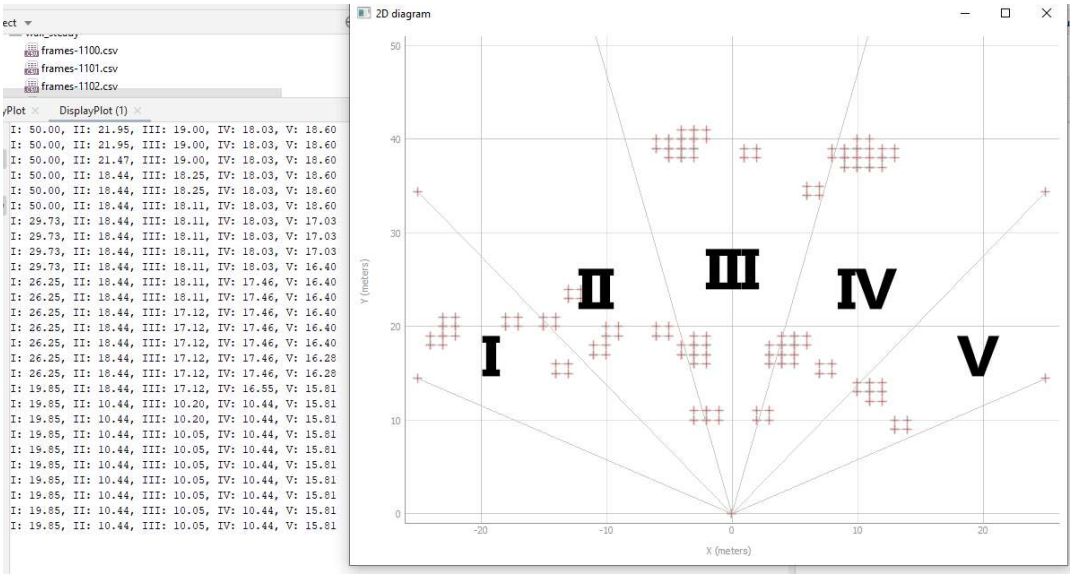
Figure 18. Updating information of sectors closest objects.

When the custom boxing method is applied on the same datasets as the previous algorithms then the results are the best. Since the radar is positioned in the end of the robot ship and few meters are not enough to save the ship from the collision then data points from first 5 m are not used. Also, instead of displaying data points, the 1 m times 1 m square boxes are displayed instead. When approaching the car (Figure 19) or the wall (Figure 20) the data groups are displayed. Each group of 1 m times 1 m is delayed, since the method used here is waiting for more points to appear in that box before it is displayed. As the robot ship team described, then the delay and showing distances +- 5 meters is enough to fill the aim and this grouping method is initially suitable.



```
I: 50.00, II: 8.25, III: 8.00, IV: 8.25, V: 50.00
I: 50.00, II: 8.25, III: 8.00, IV: 8.25, V: 50.00
I: 50.00, II: 8.25, III: 8.00, IV: 8.25, V: 50.00
I: 50.00, II: 8.25, III: 8.00, IV: 8.25, V: 50.00
I: 50.00, II: 8.25, III: 8.00, IV: 8.25, V: 50.00
I: 50.00, II: 6.71, III: 7.07, IV: 8.25, V: 50.00
I: 50.00, II: 6.71, III: 7.07, IV: 8.25, V: 50.00
I: 50.00, II: 6.71, III: 7.00, IV: 8.25, V: 50.00
I: 50.00, II: 6.71, III: 7.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.71, III: 7.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.71, III: 7.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.71, III: 7.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.08, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.08, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 7.28, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 50.00, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 50.00, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 50.00, V: 50.00
I: 50.00, II: 6.32, III: 6.00, IV: 50.00, V: 50.00
```

Figure 19. A single frame capture of plot where radar approaches the car. On the left the sectors closest object distance values are displayed where newer information is on the bottom. Numbers on plot are showing sector number. Red boxes are detected objects for the radar.



```
I: 50.00, II: 21.95, III: 19.00, IV: 18.03, V: 18.60
I: 50.00, II: 21.95, III: 19.00, IV: 18.03, V: 18.60
I: 50.00, II: 21.47, III: 19.00, IV: 18.03, V: 18.60
I: 50.00, II: 18.44, III: 18.25, IV: 18.03, V: 18.60
I: 50.00, II: 18.44, III: 18.25, IV: 18.03, V: 18.60
I: 50.00, II: 18.44, III: 18.11, IV: 18.03, V: 18.60
I: 29.73, II: 18.44, III: 18.11, IV: 18.03, V: 17.03
I: 29.73, II: 18.44, III: 18.11, IV: 18.03, V: 17.03
I: 29.73, II: 18.44, III: 18.11, IV: 18.03, V: 17.03
I: 29.73, II: 18.44, III: 18.11, IV: 18.03, V: 16.40
I: 26.25, II: 18.44, III: 18.11, IV: 17.46, V: 16.40
I: 26.25, II: 18.44, III: 18.11, IV: 17.46, V: 16.40
I: 26.25, II: 18.44, III: 17.12, IV: 17.46, V: 16.40
I: 26.25, II: 18.44, III: 17.12, IV: 17.46, V: 16.40
I: 26.25, II: 18.44, III: 17.12, IV: 17.46, V: 16.40
I: 26.25, II: 18.44, III: 17.12, IV: 17.46, V: 16.28
I: 26.25, II: 18.44, III: 17.12, IV: 17.46, V: 16.28
I: 19.85, II: 18.44, III: 17.12, IV: 16.55, V: 15.81
I: 19.85, II: 10.44, III: 10.20, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.20, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.05, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.05, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.05, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.05, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.05, IV: 10.44, V: 15.81
I: 19.85, II: 10.44, III: 10.05, IV: 10.44, V: 15.81
```

Figure 20. A single frame capture of plot where radar approaches the wall. On the left the sectors closest object distance values are displayed where newer information is on the bottom. Numbers on plot are showing sector number. Red boxes are detected objects for the radar.

4 Radar and controller software

4.1 Software for grouping data points and informing the robot ship

4.1.1 Software architecture

The final software (see Custom boxing method) is divided into three different parts - run, compute and parse. Activity diagram (Figure 21) describes the workflow of the final software. Run script is the controller which communicates with compute and parse part in order to get data ready for sending it to the robot ship. Parse is for configuring serial to read information from the radar and parsing configuration file which configures the millimetre radar according to the given config file. Also, parse reads live data from radar with 30 Hz frequency. Compute (see Custom boxing method) takes in new data with timestamp and computes new distance values for the sectors closest objects and returns them to run.

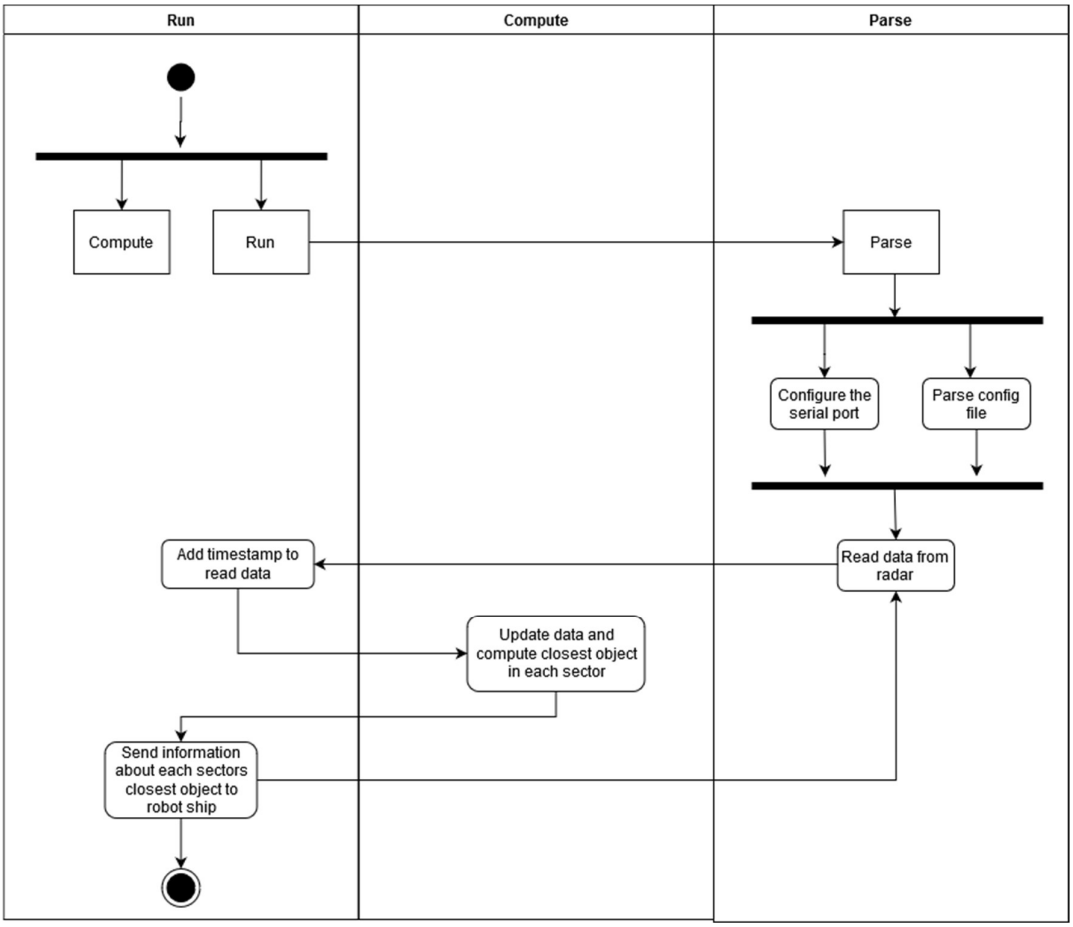


Figure 21. Activity diagram of final software where radar is configured, and data is computed for the robot ship.

4.1.2 Parsing data from the radar

As stated in [22], the program [29] starts with configuring serial ports and sends command-line interface commands from the configuration file (see

Appendix 3 – AWR1843config.cfg) to the radar. Antennas were hardcoded due to the configuration file having no information about them. Then, the data from the radar is parsed to 2D position and doppler velocity of reflected points.
The configuration in [22] had desirable configuration scene as Best Range Resolution and due to the aim the robot ship has, the configuration had to be replaced. The configuration file was generated from [30], where platform was set to xWR18xx, SDK version as 3.3 and desirable configuration scene as Best Range. After setting the parameters, the config file was downloaded with Save config to PC button and replaced with the current one.

4.1.3 Controlling parsing and computing data for the robot ship

The running script contains Run class which creates Parse object and configures its serials and sends configuration through command-line interface to be ready for detecting obstacles. Then, Compute object is created with parameter 50 which is the field size width and length.
From infinite while loop the created Run object calls Parse method which will start reading reflections data with 30 Hz frequency. Every dataset is validated and will get timestamp of the time when the reflections were captured. After that, the data set is passed to Compute object where the data is stored and closest object in each sector is computed. Closest object distances are stored in a list containing 5 float values where each value corresponds to sectors – first value is the distance from the closest object in first sector on the left, second value is about second sector from the left and so on.
In infinite while loop then starts to get reflections from the radar and checks if the data which came in is valid.

4.1.4 Informing the robot ship with current obstacles in each sector

According to the robot ship team, the radar with Raspberry Pi are independent from the main controller. Radar's controller should send sectors closest objects information over the network to the main controller. The reasonable approach for this could be over TCP. The robot ship controller would be server who listens requests and radar's controller would be a client who sends requests. The data comes initially in JSON format and without any headers.
The connection (Figure 22) is tested in a single computer and needs further development when radar is going to be applied on the ship. The connection over TCP is made using Pythons SocketServer library [31]. The server waits for client's requests to send over the data.

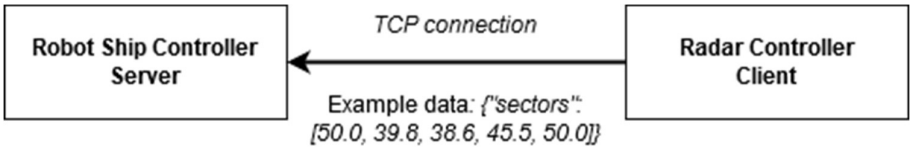


Figure 22. Block diagram of TCP connection of robot ship controller and radar controller.

4.2 Saving scenarios and replaying them through CSV

4.2.1 Scenarios

There were 8 scenarios that were captured each 10 +-5 times. Scenarios that were captured for applying clustering methods were talked through with robot ship team. The most important attributes for the scenarios were obstacle sizes, moving speed and distance from the object. Different scenarios have had different purpose on this work. Scenario number 1 with street mail and number 4 with a human are additional scenarios to get to know the radar better and how it acts. Scenario number 2 and number 3 are made to compare the affection of vertical movement since they are both about moving closer to the wall, but one has bigger vertical movement. Scenario number 7 is made to recognize if the glass of the box where the radar is in is generating the noise in 5-meter radius or not. Scenario number 6 and 7 are captured to see if the radar can detect the car and wall from 50-meters since the previous scenarios with these objects were from 20-meters or closer.
Captured scenario descriptions can be found below.

1. Radar faces street mail (Figure 23) with right side. Radar is turning right with a vertical movement +/- 0.05 m/s till it faces street mail with left side. Distance between radar and street mail is ~10 m.

Figure 23. Screenshot of street mail captured in video VID_frames-1303.mp4 on 2nd second.

2. Radar faces wall (Figure 24) with front view. Radar is moving with 5 km/h +/- 2 km/h speed towards wall with a vertical movement +/- 0.05 m/s. Distance between radar and wall is ~20 m.

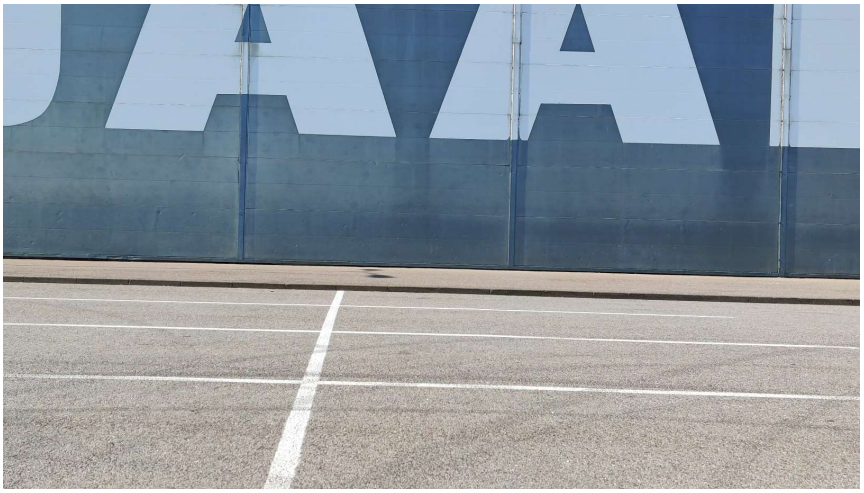


Figure 24. Screenshot of wall captured in video VID_frames-1103.mp4 on 1st second.

3. Radar faces wall (Figure 25) with front view. Radar is moving with 5 km/h +/- 2 km/h speed towards wall with vertical movement +/- 0.2 m/s. Distance between radar and wall is ~20 m.

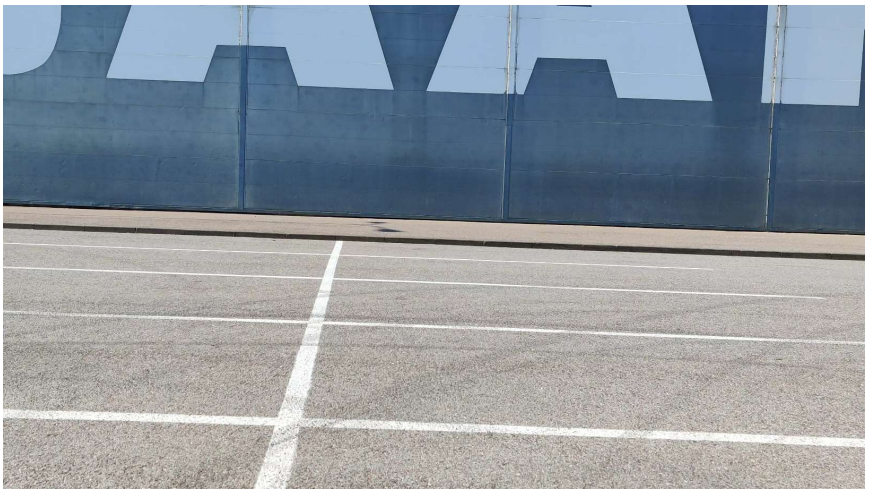


Figure 25. Screenshot of wall captured in video VID_frames-1208.mp4 on 1st second.

4. Radar has random facing direction. Radar is doing a 180 degree turn to right +/- 20 degrees with a vertical movement +/- 0.05 m/s. Human (Figure 26) is moving from right to left at random time from ~5 m.

Figure 26. Screenshot of human captured in VID_frames-1403.mp4 on 3rd second.

5. Radar faces a car (Figure 27) with front view. Radar is moving with 5 km/h +/- 2 km/h speed towards a car with vertical movement +/- 0.05 m/s. Distance between radar and a car is ~15 m.

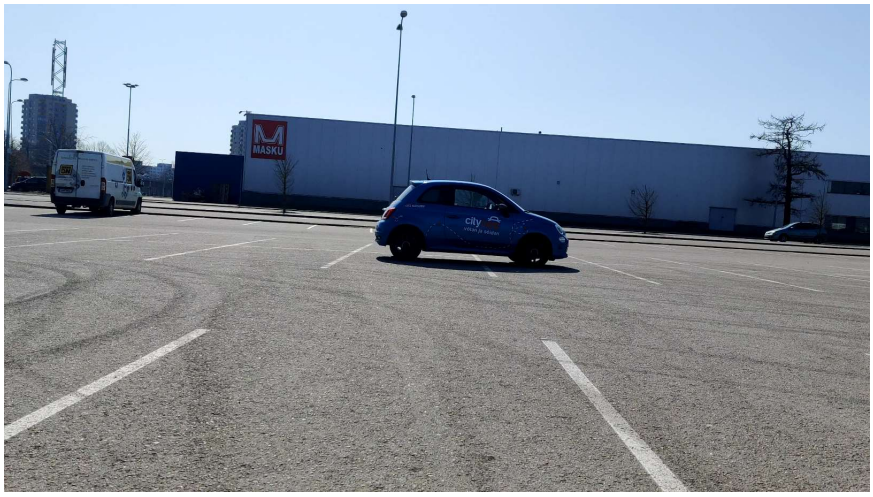


Figure 27. Screenshot of car captured in VID_frames-1017.mp4 on 1st second.

6. Radar faces a car (Figure 28) with front view. Radar is moving with 5 km/h +/- 2 km/h speed towards a car with vertical movement +/- 0.05 m/s. Distance between radar and a car is ~50 m.

Figure 28. Screenshot of car captured in VID_20200422_170149.mp4 on 1st second.

7. Radar faces a car (Figure 29) with front view. Radar is moving with 5 km/h +/- 2 km/h speed towards a car with vertical movement +/- 0.05 m/s. Distance between radar and a car is ~25 m. A glass is removed of the box where the radar is held at.

Figure 29. Screenshot of car captured in VID_20200422_173309.mp4 on 1st second.

8. Radar faces wall (Figure 30) with front view. Radar is moving with 5 km/h +/- 2 km/h speed towards wall with a vertical movement +/- 0.05 m/s. Distance between radar and wall is ~50m.

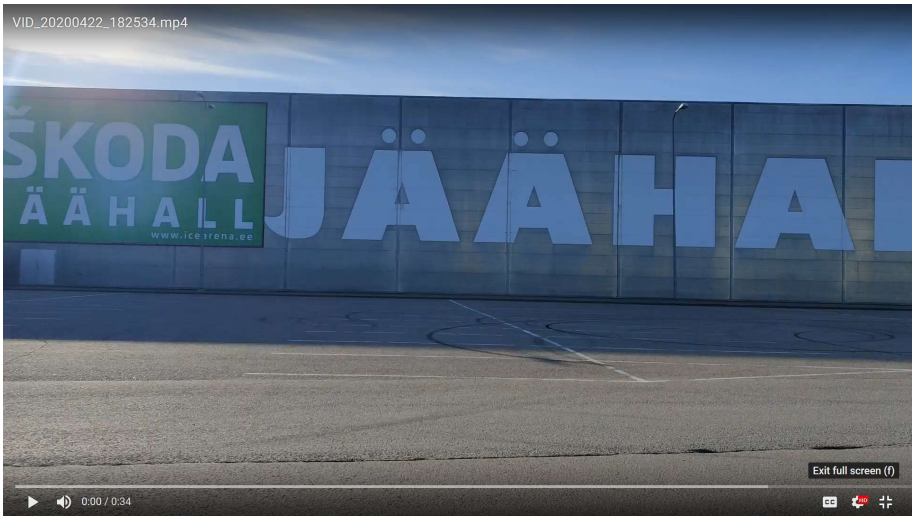


Figure 30. Screenshot of wall captured in VID_20200422_182534.mp4 on 1st second.

4.2.2 Detecting data points and saving them to a file

Parsing data points was done by using code from [29]. When different grouping methods needed to be tested then a good testing environment had to be created. Good testing environment is when different grouping algorithms can be applied on the same datasets which makes it faster and more reliable to compare the results.
Schema of the workflow (Figure 31) describes the modifications made for the parsing code [29]. Displaying the plot was removed from the script because radar's controller does not have GUI to display the plot and the plot has no purpose there. The script continuously parses reflections from the radar and valid data is appended on current csv file. Headers for the csv file are following: *frame_number* as number of dataset, *num_obj* as number of objects detected, *x* as x-axis values in meters, *y* as y-axis values in meters, *z* as z-axis values in meters, *velocity* as velocity of each datapoint towards the radar and *local_time* as timestamp when the data points were captured. Example of the file (see

Appendix 2 – frames-1013.csv) has first rows with recorded data where radar was facing car from 15 meters and started moving towards it in regular pedestrian speed.

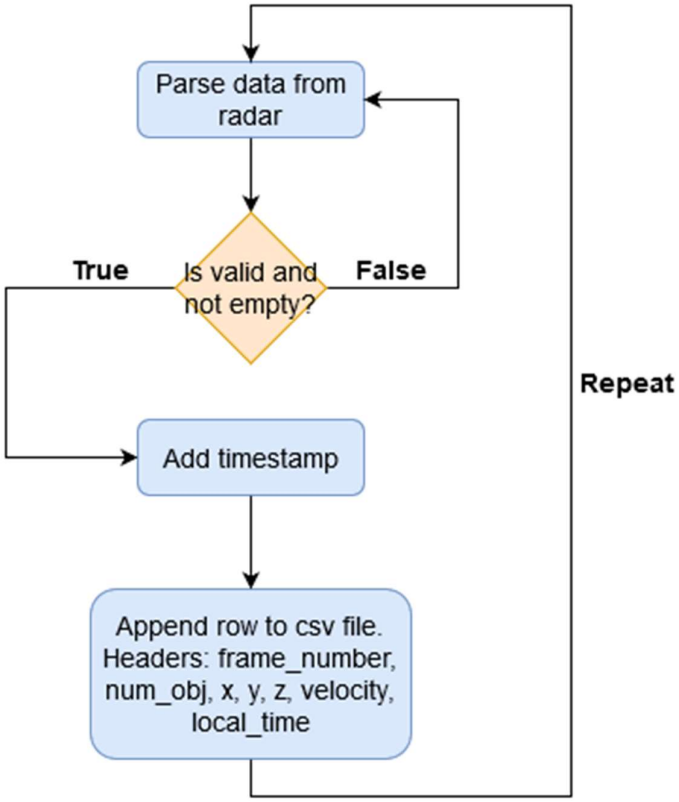


Figure 31. Workflow of data parsed from millimetre wave radar and saved in csv file.

4.2.3 Displaying data points from file on a 2D scatter plot

Displaying data points with a better illustration gives this work better result outputs. Differently grouping method data points were illustrated in each way. An example of one frame (Figure 12) and workflow (Figure 32) can be used to describe plots made for K-Means clustering, Mini Batch K-Means clustering, DBSCAN clustering and OPTICS clustering. Data points have triangles and relate to grey lines between the triangles. Also, + symbol is used to show noise in DBSCAN clustering and in OPTICS clustering. In the function, where the single dataset is asked if it has two or more data points in order to display them on the plot. Then, a noise which clustering value is -1, is displayed differently. When new cluster is detected then the colour is generated randomly. The loop goes through all points until all of them are displayed.

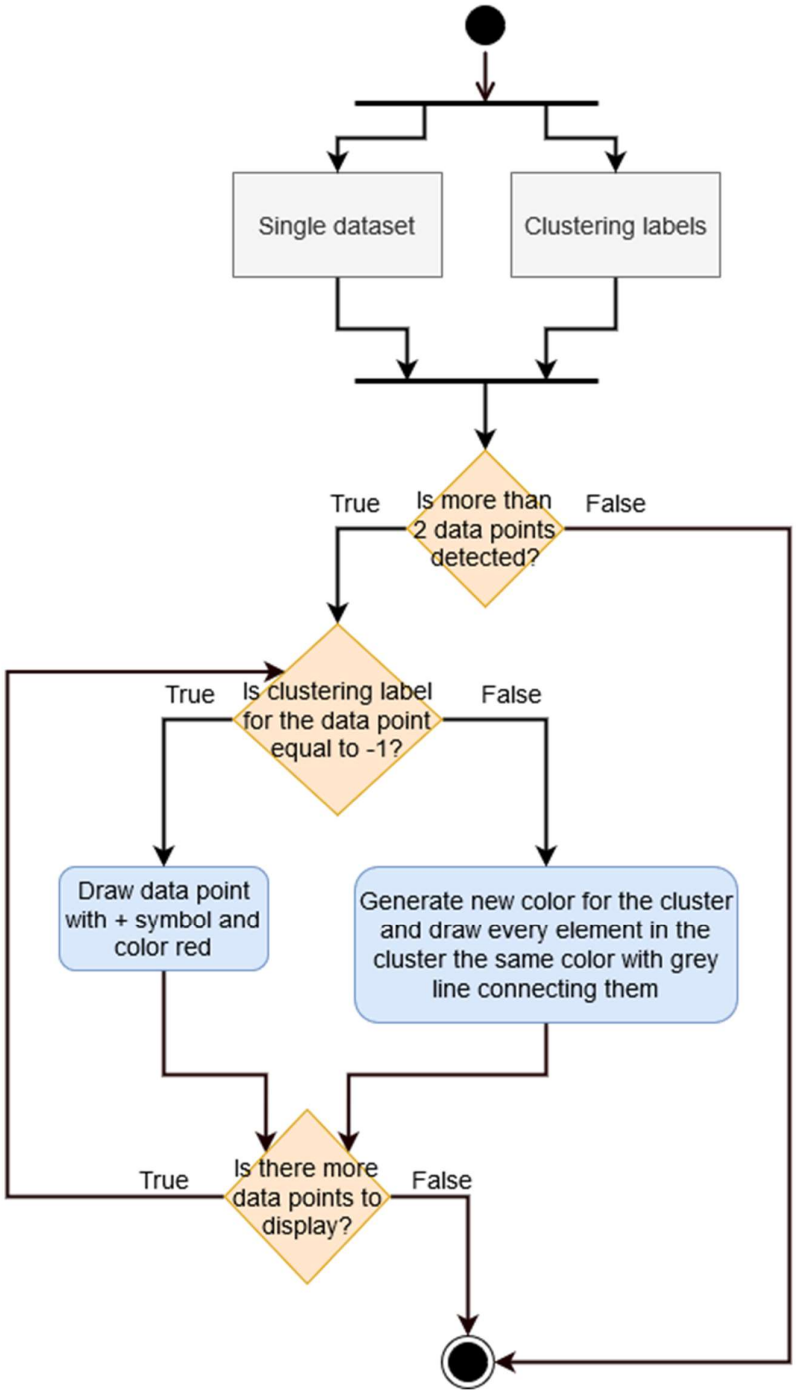


Figure 32. Workflow of displaying datapoints according to the cluster result.

Custom boxing method has unique displaying method, which was used for example in this figure (Figure 19). For custom boxing method the sectors are displayed along with data squares. Each sector border is drawn just for demonstration and is having start and end point with + symbol which relate with grey line. For drawing square boxes to display grouped objects a box is displayed where + symbols are in each corner and lines are drawn between them to draw a square.

5 Summary

The goal of this thesis was to assist finding suitable millimetre wave radar and write a software [20] which detects obstacles in 50 m radius for the robot ship NYMO. During the development, different software had to be implemented to reach the result. Eight different scenarios were played through to determine how the radar works and apply grouping methods for the real data. The built scripts can do the following functionality:

- Read data from AWR1843BOOST
- Save incoming data from the radar to files for re-living the captured moments
- Re-live captured moments with AWR1843BOOST
- Apply grouping methods like Mini Batch K-Means, K-Means, DBSCAN, OPTICS and custom boxing method for the same datasets
- Custom boxing method grouping method and calculation of closest object in a sector
- Display live clusters after grouping method is applied on 2D scatter plot
- Initial approach to send data from client to server

According to the robot ship team, the goal of this thesis is achieved. The value of this result is high as the other sensors on the robot ship NYMO could not detect objects as far as 50 meters. The software was made for working in real time which is crucial for the robot ship to operate quickly before driving into the obstacle. The robot ship NYMO will be more autonomous than before.

For the future work the radar should be mounted on the ship to work with its controller. Then, current software should be tested in real operating conditions for the better results about filling the aim. The radar can be integrated with camera and AIS system. The radar should also be tested for longer distances and integrate it into ROS system due to its ease of use and robot ship's hardware.

References

[1] B. Marr, 'The Incredible Autonomous Ships Of The Future: Run By Artificial Intelligence Rather Than A Crew', *Forbes*, 05-Jun-2019.

[2] 'Nymo avalik - Photo Station 6'. [Online]. Available: http://213.168.10.134/photo/share/PmKYvolY#!List/photo_4e796d6f2f4176616c696b_44534346313634312e4a5047. [Accessed: 08-May-2020].

[3] E. P. Engineering, 'Proof of Concept of Energy Efficient Universal Autonomous Surface Vessel', Tallinn, 2018.

[4] S. Paulus, 'NYMO shows the future of maritime industry', *Research In Estonia*, 07-Oct-2019.

[5] G. Brooker and S. J. Scheding, 'Millimetre waves for robotics Mining Radar View project Harmonic Tags View project', 2001.

[6] M. Schneider, 'Automotive Radar – Status and Trends', *Ger. Microw. Conf.*, pp. 3–6, 2006.

[7] S. Ogawa, T. Fukunaga, S. Yamagishi, M. Yamada, and T. Inaba, '76-GHz high-resolution radar for autonomous driving support', *SEI Tech. Rev.*, no. 86, pp. 12–17, 2018.

[8] C. Iovescu, R. A. Manager, and S. Rao, 'The fundamentals of millimeter wave sensors', *Texas Instruments*, pp. 1–8, 2017.

[9] K. Garcia, M. Yan, and A. Purkovic, 'Robust traffic and intersection monitoring using millimeter wave sensors', p. 13, 2018.

[10] J. Wenger, 'Automotive radar - Status and perspectives', *Tech. Dig. - IEEE Compd. Semicond. Integr. Circuit Symp. CSIC*, pp. 21–24, 2005, doi: 10.1109/CSICS.2005.1531741.

[11] M. Li, P. Schmalenberg, and J. S. Lee, 'Millimeter-wave tapered slot array for automotive radar applications', *ISAP 2016 - Int. Symp. Antennas Propag.*, pp. 702–703, 2017.

[12] C. Berkius *et al.*, 'Human Control of Mobile Robots Using Hand Gestures', 2018.

[13] D. Noetel, W. Johannes, M. Caris, A. Hommes, and S. Stanko, 'Detection of MAVs (Micro Aerial Vehicles) based on millimeter wave radar', *Millimetre Wave Terahertz Sensors Technol. IX*, vol. 9993, no. May, p. 999308, 2016, doi: 10.1117/12.2242020.

[14] S. Churkin and L. Anishchenko, 'Millimeter-wave radar for vital signs monitoring', *2015 IEEE Int. Conf. Microwaves, Commun. Antennas Electron. Syst. COMCAS 2015*, no. November, pp. 1–4, 2015, doi: 10.1109/COMCAS.2015.7360366.

[15] 'AWR1843 single-chip 76-GHz to 81-GHz automotive radar sensor evaluation module'. [Online]. Available: http://www.ti.com/tool/AWR1843BOOST. [Accessed: 04-Feb-2020].

[16] 'AWR1642BOOST AWR1642 single-chip 76-GHz to 81-GHz automotive radar sensor evaluation module | TI.com'. [Online]. Available: http://www.ti.com/tool/AWR1642BOOST#1. [Accessed: 12-Feb-2020].

[17] 'Download PuTTY - a free SSH and telnet client for Windows'. [Online].

Available: https://www.putty.org/. [Accessed: 09-May-2020].
[18] 'WinSCP :: Official Site :: Download'. [Online]. Available: https://winscp.net/eng/download.php. [Accessed: 09-May-2020].
[19] 'Getting started with Git - What is Git? | Raspberry Pi Projects'. [Online]. Available: https://projects.raspberrypi.org/en/projects/getting-started-with-git/2. [Accessed: 09-May-2020].
[20] 'Karl Matti / iaib · GitLab'. [Online]. Available: https://gitlab.cs.ttu.ee/kamatt/awr1843boost-for-autonomous-ship-nymo. [Accessed: 09-May-2020].
[21] 'MMWAVE-SDK mmWave software development kit (SDK) | TI.com'. [Online]. Available: https://www.ti.com/tool/MMWAVE-SDK. [Accessed: 13-May-2020].
[22] 'ibaiGorordo/AWR1843-Read-Data-Python-MMWAVE-SDK-3-: Python program to read and plot the data in real time from the AWR1843 mmWave radar board (MMWAVE SDK 3)'. [Online]. Available: https://github.com/ibaiGorordo/AWR1843-Read-Data-Python-MMWAVE-SDK-3-. [Accessed: 13-May-2020].
[23] 'mmWaveSensingEstimator'. [Online]. Available: https://dev.ti.com/gallery/view/1792614/mmWaveSensingEstimator/ver/1.3.0/. [Accessed: 04-May-2020].
[24] O. Schumann, M. Hahn, and J. Dickmann, 'Supervised Clustering for Radar Applications:On the Way to Radar Instance Segmentation', Dortmund, 2018.
[25] 'Comparing different clustering algorithms on toy datasets — scikit-learn 0.22.2 documentation'. [Online]. Available: https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html. [Accessed: 09-May-2020].
[26] 'sklearn.cluster.DBSCAN — scikit-learn 0.22.2 documentation'. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN. [Accessed: 10-May-2020].
[27] '2.3. Clustering — scikit-learn 0.22.2 documentation'. [Online]. Available: https://scikit-learn.org/stable/modules/clustering.html. [Accessed: 09-May-2020].
[28] 'sklearn.cluster.OPTICS — scikit-learn 0.22.2 documentation'. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.OPTICS.html#sklearn.cluster.OPTICS. [Accessed: 10-May-2020].
[29] 'AWR1843-Read-Data-Python-MMWAVE-SDK-3-/readData_AWR1843.py at master · ibaiGorordo/AWR1843-Read-Data-Python-MMWAVE-SDK-3-'. [Online]. Available: https://github.com/ibaiGorordo/AWR1843-Read-Data-Python-MMWAVE-SDK-3-/blob/master/readData_AWR1843.py. [Accessed: 15-May-2020].
[30] 'mmWave Demo Visualizer'. [Online]. Available: https://dev.ti.com/gallery/view/mmwave/mmWave_Demo_Visualizer/ver/3.4.0/. [Accessed: 15-May-2020].
[31] '20.17. SocketServer — A framework for network servers — Python 2.7.18 documentation'. [Online]. Available: https://docs.python.org/2/library/socketserver.html. [Accessed: 15-May-2020].
[32] 'xWR1843 Evaluation Module (xWR1843BOOST) Single-Chip mmWave Sensing Solution', 2019. [Online]. Available: http://www.ti.com/lit/ug/spruim4a/spruim4a.pdf. [Accessed: 05-Dec-2019].

[33] ‘Automotive TI mmWave sensors for mid-range radar’. [Online]. Available: https://www.youtube.com/watch?v=1PkcbE3zrYo. [Accessed: 04-Feb-2020].
[34] ‘MMWAVE-STUDIO mmWave studio | TI.com’. [Online]. Available: http://www.ti.com/tool/MMWAVE-STUDIO. [Accessed: 04-Feb-2020].
[35] ‘MMWAVE-SDK 03_03_00_03 - TI.com’. [Online]. Available: http://software-dl.ti.com/ra-processors/esd/MMWAVE-SDK/latest/index_FDS.html. [Accessed: 04-Feb-2020].
[36] ‘AWR1843BOOST | AWR1843 single-chip 76-GHz to 81-GHz automotive radar sensor evaluation module | TI store’. [Online]. Available: https://www.ti.com/store/ti/en/p/product/?p=AWR1843BOOST&utm_source=google&utm_medium=cpc&utm_campaign=epd-null-null-OPN_EN-cpc-store-google-wwe&utm_content=Device&ds_k=AWR1843BOOST&DCM=yes&gclid=EAIaIQobChMIt6bLqa-w5gIVQamaCh0hvAdvEAAYASAAEgKlX_D_BwE&gcl. [Accessed: 04-Feb-2020].
[37] ‘AWR1642 Single-Chip 77- and 79-GHz FMCW Radar Sensor’, 2018. [Online]. Available: https://www.ti.com/lit/ds/swrs203a/swrs203a.pdf. [Accessed: 12-Feb-2012].
[38] A. Nguyen, ‘mmWave Radar Sensors : Object Versus Range’, 2018.
[39] ‘AWR1642BOOST Texas Instruments | Mouser Estonia’. [Online]. Available: https://www.mouser.ee/ProductDetail/Texas-Instruments/AWR1642BOOST?qs=sGAEpiMZZMuC4zZxLL0ZTU4aJQ6YsVh%2F5fDtkF%2FguNKc3P9jL7onDQ==. [Accessed: 12-Feb-2020].
[40] ‘AWR1443BOOST AWR1443 single-chip 76-GHz to 81-GHz automotive radar sensor evaluation module | TI.com’. [Online]. Available: https://www.ti.com/tool/AWR1443BOOST?utm_source=google&utm_medium=cpc&utm_campaign=epd-null-null-GPN_EN_EVM-cpc-evm-google-wwe&utm_content=AWR1443BOOST&ds_k=AWR1443BOOST&DCM=yes&gclid=EAIaIQobChMI8Zjwrv3J6AIVxLUYCh14UA8tEAAYASAAEgLUwvD_BwE&gclsrc=aw.ds. [Accessed: 02-Apr-2020].
[41] ‘AWR1243BOOST | AWR1243 76-GHz to 81-GHz high-performance automotive MMIC evaluation module | TI store’. [Online]. Available: https://www.ti.com/store/ti/en/p/product/?p=AWR1243BOOST. [Accessed: 12-Feb-2020].
[42] ‘[Resolved] AWR1642BOOST: AWR1642 for long range(200 metres) - Sensors forum - Sensors - TI E2E support forums’, 2019. [Online]. Available: https://e2e.ti.com/support/sensors/f/1023/p/764851/2827223#2827223. [Accessed: 12-Feb-2020].
[43] ‘OPS243 Short-Range Radar’, San Jose.
[44] ‘OmniPreSense OPS243 Doppler Speed Radar Sensor - RobotShop’. [Online]. Available: https://www.robotshop.com/en/omnipresense-ops243-doppler-speed-radar-sensor.html. [Accessed: 02-Apr-2020].
[45] R. S. Types and T. Control, ‘AN-010 API’.
[46] ‘OPS243 Short-Range Radar Sensor - OmniPreSense | Mouser’. [Online]. Available: https://www.mouser.ee/new/omnipresense/omnipresense-ops243-radar-sensor/. [Accessed: 05-May-2020].
[47] ‘IWR6843ISK IWR6843 intelligent mmWave sensor standard antenna plug-in module | TI.com’. [Online]. Available: https://www.ti.com/tool/IWR6843ISK#2. [Accessed: 03-May-2020].
[48] ‘24GHz Millimeter-Wave Obstacle Avoidance Radar for drones to detect target

obstacles'. [Online]. Available: https://www.foxtechfpv.com/24ghz-millimeter-wave-obstacle-avoidance-radar.html#yt_tab_products1. [Accessed: 05-May-2020].
[49] '24G 雷达与 Pixhawk V3 飞控安装说明 2019.04', 2019.
[50] 'Spot: 250 Meters ARS408 21 Millimeter Wave Radar with Full Set of Agreement Documents, 1 Insurance|Arc Welders| - AliExpress'. [Online]. Available: https://www.aliexpress.com/item/4000109085932.html. [Accessed: 05-May-2020].
[51] [Revised July 2017] Continental Industrial Sensors, 'ARS 408-21 Premium Long Range Radar Sensor 77 GHz', 2015.
[52] I. C. Vision, 'Sensor Fusion Kit Integrated Camera Vision & mmWave RADAR', Bangalore, 1843.
[53] 'Sensor Fusion Kit - Mistral Solutions'. [Online]. Available: https://www.mistralsolutions.com/product/sensor-fusion-kit/. [Accessed: 05-May-2020].
[54] '60GHz AoPCB RADAR Module: Mistral - IWR6843 mmWave Sensor'. [Online]. Available: https://www.mistralsolutions.com/product-engineering-services/products/som-modules/60ghz-industrial-mmwave-radar-module/. [Accessed: 05-May-2020].

Appendix 1 – IWR6843.estimator.mmwave.json


```
{
    "configGenerator": {
        "createdBy": "mmWaveSensingEstimator",
        "createdOn": "2020-05-04T18:35:10.227Z",
        "isConfigIntermediate": 0
    },
    "currentVersion": {
        "jsonCfgVersion": {
            "major": 0,
            "minor": 4,
            "patch": 0
        },
        "DFPVersion": {
            "major": 1,
            "minor": 2,
            "patch": 0
        },
        "SDKVersion": {
            "major": 3,
            "minor": 0,
            "patch": 0
        },
        "mmwavelinkVersion": {
            "major": 1,
            "minor": 2,
            "patch": 0
        }
    },
    "lastBackwardCompatibleVersion": {
        "DFPVersion": {
            "major": 1,
            "minor": 1,
            "patch": 0
        },
        "SDKVersion": {
            "major": 2,
            "minor": 0,
            "patch": 0
        },
        "mmwavelinkVersion": {
            "major": 1,
            "minor": 1,
            "patch": 0
```

```
    }
},
"systemConfig": {
    "summary": "",
    "sceneParameters": {
        "ambientTemperature_degC": 20,
        "maxDetectableRange_m": 70,
        "rangeResolution_cm": 45,
        "maxVelocity_kmph": 65,
        "velocityResolution_kmph": 2,
        "measurementRate": 25,
        "typicalDetectedObjectRCS": 5
    }
},
"regulatoryRestrictions": {
    "frequencyRangeBegin_GHz": 77,
    "frequencyRangeEnd_GHz": 81,
    "maxBandwidthAllowed_MHz": 4000,
    "maxTransmitPowerAllowed_dBm": 12
},
"processingChainConfig": {
    "detectionChain": {
        "name": "",
        "detectionLoss": 1,
        "systemLoss": 1,
        "implementationMargin": 2,
        "detectionSNR": 12,
        "theoreticalRxAntennaGain": 9,
        "theoreticalTxAntennaGain": 9
    }
},
"mmWaveDevices": [
    {
        "mmWaveDeviceId": 0,
        "rfConfig": {
            "summary": "",
            "waveformType": "singleFrameChirp",
            "MIMOScheme": "TDM",
            "rlChanCfg_t": {
                "rxChannelEn": "0xF",
                "txChannelEn": "0x1"
            },
            "rlAdcOutCfg_t": {
                "fmt": {
                    "b2AdcBits": 2,
                    "b8FullScaleReducFctr": 0,
                    "b2AdcOutFmt": 2
                }
            },
            "rlLowPowerModeCfg_t": {},
            "rlProfiles": [
```

```
    {
        "rlProfileCfg_t": {
            "profileId": 0,
            "pfVcoSelect": "0x0",
            "pfCalLutUpdate": "0x0",
            "startFreqConst_GHz": 60,
            "idleTimeConst_usec": 2,
            "adcStartTimeConst_usec": 6.4,
            "rampEndTime_usec": 67.06666666666666,
            "txOutPowerBackoffCode": "0x0",
            "txPhaseShifter": "0x0",
            "freqSlopeConst_MHz_usec": 5.600452423095703,
            "txStartTime_usec": 1,
            "numAdcSamples": 179,
            "digOutSampleRate": 10000,
            "hpfCornerFreq1": 0,
            "hpfCornerFreq2": 0,
            "rxGain_dB": "0x1E"
        }
    }
],
"rlChirps": [
    {
        "rlChirpCfg_t": {
            "chirpStartIdx": 0,
            "chirpEndIdx": 0,
            "profileId": 0,
            "startFreqVar_MHz": 0,
            "freqSlopeVar_KHz_usec": 0,
            "idleTimeVar_usec": 0,
            "adcStartTimeVar_usec": 0,
            "txEnable": "0x1"
        }
    }
],
"rlRfCalMonTimeUntConf_t": {
    "calibMonTimeUnit": 1
},
"rlRfCalMonFreqLimitConf_t": {
    "freqLimitLow_GHz": 77,
    "freqLimitHigh_GHz": 81
},
"rlRfInitCalConf_t": {
    "calibEnMask": "0x1FF0"
},
"rlRunTimeCalibConf_t": {
    "oneTimeCalibEnMask": "0x0",
    "periodicCalibEnMask": "0x0",
    "calibPeriodicity": 10,
    "reportEn": 1,
    "txPowerCalMode": 0
```

```
                },
                "rlFrameCfg_t": {
                    "chirpEndIdx": 0,
                    "chirpStartIdx": 0,
                    "numLoops": 66,
                    "numFrames": 0,
                    "framePeriodicity_msec": 4.554
                }
            },
            "rawDataCaptureConfig": {
                "rlDevDataFmtCfg_t": {
                    "iqSwapSel": 0,
                    "chInterleave": 1
                }
            },
            "monitoringConfig": {}
        }
    ]
}
```

Appendix 2 – frames-1013.csv

```
frame_number,num_obj,x,y,z,velocity,local_time
1,2,[   0.          -13.190572],[15.144731  32.607967],[0.  0.],[-0.25026876  -
0.25026876],2020-04-07 10:50:15.172
2,2,[   0.          -13.190572],[15.144731  32.607967],[0.  0.],[-0.37540314  -
0.37540314],2020-04-07 10:50:15.282
3,2,[-0.9312788  17.770632  ],[14.87133    15.672223],[0.  0.],[-0.62567186  -
0.37540314],2020-04-07 10:50:15.395
4,6,[-0.9312788     0.3435045     0.                  0.58777434  -0.80151045
0.41220537],[14.87133        1.1720489    1.2213492    1.6056904    1.5103977
1.4064587],[0.  0.  0.  0.  0.  0.],[-0.7508063    -0.37540314  -0.37540314  -
0.37540314 -0.37540314 -0.25026876],2020-04-07 10:50:15.456
5,7,"[-0.45800596 -0.45800596  0.80151045 -1.3969183  -0.13740179  0.36640477
 -0.11450149]","[  2.8952353    3.6353097    3.5753078  14.834836     1.4591641
1.4190795
  1.2159702]",[0. 0. 0. 0. 0. 0. 0.],"[-0.7508063  -0.7508063  -0.7508063  -
0.7508063  -0.37540314 -0.37540314
 -0.25026876]",2020-04-07 10:50:15.569
```

Appendix 3 – AWR1843config.cfg

```
% ***************************************************************
% Created for SDK ver:03.03
% Created using Visualizer ver:3.3.0.0
% Frequency:77
% Platform:xWR18xx
% Scene Classifier:best_range
% Azimuth Resolution(deg):15
% Range Resolution(m):0.244
% Maximum unambiguous Range(m):50
% Maximum Radial Velocity(m/s):1
% Radial velocity resolution(m/s):0.13
% Frame Duration(msec):100
% Range Detection Threshold (dB):15
% Doppler Detection Threshold (dB):15
% Range Peak Grouping:enabled
% Doppler Peak Grouping:enabled
% Static clutter removal:disabled
% Angle of Arrival FoV: Full FoV
% Range FoV: Full FoV
% Doppler FoV: Full FoV
% ***************************************************************
sensorStop
flushCfg
dfeDataOutputMode 1
channelCfg 15 5 0
adcCfg 2 1
adcbufCfg -1 0 1 1 1
profileCfg 0 77 458 7 28.49 0 0 30 1 256 12499 0 0 30
chirpCfg 0 0 0 0 0 0 0 1
chirpCfg 1 1 0 0 0 0 0 4
frameCfg 0 1 16 0 100 1 0
lowPower 0 0
guiMonitor -1 1 1 0 0 0 1
cfarCfg -1 0 2 8 4 3 0 15 1
cfarCfg -1 1 0 4 2 3 1 15 1
multiObjBeamForming -1 1 0.5
clutterRemoval -1 0
calibDcRangeSig -1 0 -5 8 256
extendedMaxVelocity -1 0
lvdsStreamCfg -1 0 0 0
compRangeBiasAndRxChanPhase 0.0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
0
measureRangeBiasAndRxChanPhase 0 1.5 0.2
CQRxSatMonitor 0 3 4 63 0
CQSigImgMonitor 0 127 4
analogMonitor 0 0
aoaFovCfg -1 -90 90 -90 90
cfarFovCfg -1 0 0 49.99
```

```
cfarFovCfg -1 1 -1 1.00
sensorStart
```

Appendix 4 – Comparison of different radars

1. Millimetre wave sensor device AWR1843BOOST
   - AWR1843BOOST (Figure 33) is shown below from [15].
   - The [32] brings out its capability of operating in a 76 - 81 GHz frequency band.
   - As shown in [33], the radar is considered as a right solution for medium range radar applications due to its small form factor, low power and less than 4 cm high range resolution.
   - Like [33] also states, it has 150 meters of detection range.
   - According to [32], the beam width of the antenna is dependent on the radiation patterns. For example, at 78 GHz, the horizontal 3 dB-beamwidth is approximately ±28 degrees and elevation ±14 degrees. Similarly, the horizontal 6dB-beamwidth is approximately ±50 degrees and elevation ±20 degrees.
   - Official software's are available from [34]. Software mmWave Studio is a collection of tools which helps to experiment with the out-of-the-box demo. It contains mmWave Studio, mmWave Sensing Estimator and mmWave Demo Visualizer. The mmWave Demo Visualizer enables real-time plotting of point cloud from output of mmWare SDK which is fully configured through GUI interface. The mmWave Studio is free. As stated in [35], the software mmWave SDK is providing easy setup and fast out-of-the-box access to evaluation and development. SDK is available for Windows and Linux.
   - According to the official page at [36] the price for AWR1843BOOST is 299 USD.

Figure 33. AWR1843BOOST AWR1843 single-chip 76 - GHz to 81 - GHz automotive radar sensor evaluation board image.

2. Millimetre wave sensor device AWR1642BOOST
   - AWR1642BOOST's FMCW Transceiver is capable of operation in the 76 - to 81 - GHz band which is stated in [37].
   - According to [38] the sensor can detect cars from 199 m with long range chirp configuration from traffic monitoring. Without the configuration the sensor can see trucks up to 160 m range.
   - Price for the radar (Figure 24) is 299 USD on [39].

- The device can use mmWave studio software to process ADC data which was defined in [34]. There is a unified software platform mmWave SDK for TI radars which supports evaluation and development of the device as stated in [35].

Figure 34. AWR1642BOOST AWR1642 single-chip 76-GHz to 81-GHz automotive radar sensor integrating DSP and MCU evaluation board image (angled view)

3. Millimetre wave sensor device AWR1443BOOST
   - According to [40], AWR1443BOOST's (Figure 35) FMCW Transceiver is capable of operation in the 76- to 81-GHz band.
   - The sensor using high RCS chirp configuration from object versus range can see trucks up to 160 m and cars up to 120 m as stated in [38].
   - Price for the sensor is 299 USD on [40].
   - The device can use mmWave studio software to process ADC data which was defined in [34]. There is a unified software platform mmWave SDK for TI radars which supports evaluation and development of the device as stated in [35].

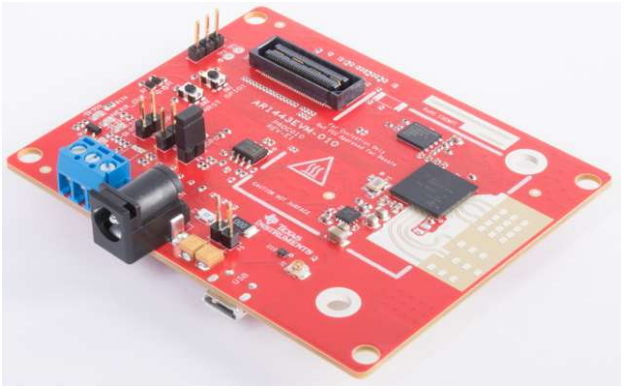


Figure 35. AWR1443BOOST AWR1443 single-chip 76-GHz to 81-GHz automotive radar sensor evaluation board image (angled view)

4. Millimetre wave sensor device AWR1243BOOST
   - According to [41] the AWR1243BOOST (Figure 36) is capable of operation in the 76- to 81-GHz band.
   - 1243 device was recommended to use for more than 160 m detection range for cars in [42].

- Price for the radar is 299 USD on [41].
- The device can use mmWave studio software to process ADC data which was defined in [34]. There is a unified software platform mmWave SDK for TI radars which supports evaluation and development of the device as stated in [35].

Figure 36. AWR1243BOOST AWR1243 76-GHz to 81-GHz high-performance automotive MMIC evaluation board image (angled view)

5. Millimetre wave sensor device OPS243
   - According to [43] the OPS243 (Figure 37) has detection range from 1 m to 100 m while reporting speed up to 560 km/h.
   - Also, it has a small width in relation to distance as 20 degrees.
   - The sensor costs 209 USD when bought from [44].
   - There is an AN-010-Q API interface which helps to control sensor output which is found in [45].
   - Four applications have been made for the sensor: Traffic Monitoring, Drone Collision Avoidance, Robotic Control and IoT Sensor as stated in [44].
   - The device has 24 GHz RF as shown in [46].

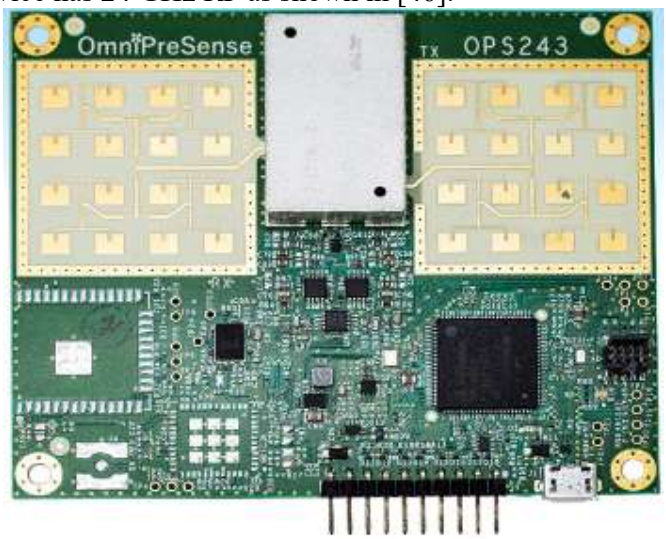


Figure 37. OPS243

6. Carrier platform MMWAVEICBOOST and plug-in millimetre wave sensor module IWR6843ISK

- IWR6843ISK (Figure 38) is a millimetre wave sensor with a long-range antenna enabling direct connectivity to its carrier MMWAVEICBOOST as defined in [47].
- The sensor enables point-cloud data through a USB and raw ADC data through a 60-pin connector as stated in [47].
- According to [47] the radar has 4 RX and 3 TX antenna with 108 degrees azimuth FoV and 44 degrees elevation FoV.
- As described in [47] it is a 60- to 64-GHz millimetre wave sensor.
- When using Texas Instruments official mmWave Sensing Estimator's, see [23] then with Long Range Default configuration for IWR6843 (see Appendix 1 – IWR6843.estimator.mmwave.json) the max range for detecting cars is 114.43 meters.
- The sensor with the carrier can use Texas Instruments mmWave Studio for characterizing and evaluating the device which also needs DCA1000 EVM as described in [35].
- Price in [47] for the sensor module is 125 dollars and for the carrier platform is 199 dollars which makes total of 324 dollars.
- There is a unified software platform mmWave SDK for TI radars which supports evaluation and development of the device as stated in [35].

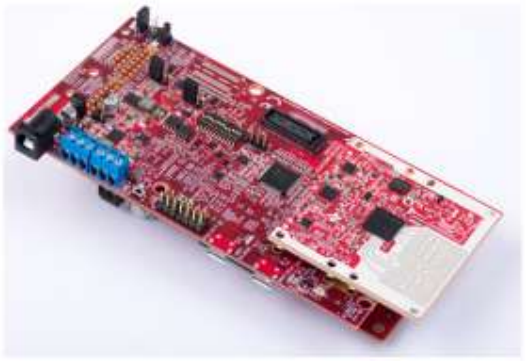


Figure 38. IWR6843ISK mmWave sensor on its carrier platform MMWAVEICBOOST

7. Millimetre wave sensor device PIXHAWK V3
   - The 24GHz radar (Figure 39Figure 2) has detection angles in horizontal as 28 degrees and pitch as 18 degrees as stated in [48].
   - There is a software built for the radar which will provide a square screen of current location and displays lines as objects along with their distance from radar which is described in [49].
   - Pixhawk V3 has detecting range maximum of 100 m as stated in [48].
   - Price for the sensor is 359.99 USD in [48].

Figure 39. 24GHz Millimeter-Wave Obstacle Avoidance Radar

8. Millimetre wave sensor device ARS408-21
   - ARS408 (Figure 40) is a long distance radar working on 77GHz frequency as stated in [50].
   - The sensor can see from 0.2 meters to 250 meters as described in [50].
   - The sensor comes along with a software which is built for cars detecting different objects like trucks, pedestrians and so on as defined in [50].
   - The sensor can see +-60 degrees in horizontal and +-20 degrees in elevation up to 70 meters. Also it sees +-9 degrees in horizontal and +-14 degrees in elevation in far range which is up to 250 meters as defined in [51].
   - Price for the radar is 844.11 dollars in [51].

Figure 40. ARS408-21 Millimetre Wave Radar

9. Millimetre wave radar integration Sensor Fusion Kit
   - Sensor fusion kit (Figure 41) is a camera and millimetre wave radar integration which uses TI single chip 77 GHz millimetre wave sensors AWR1443 and AWR1843 as described in [52].

- The radar is covering 76 – 81 GHz frequency Bandwidth as stated in [52].
- Starting software for the kit is offering classification and identification of captured objects and stream radars data as a video. Also different customization services are offered to tune the millimetre wave radar and camera's video as described in [52].
- Price for the kit is 1499 dollars in [53].

Figure 41. Sensor Fusion Kit

10. Millimetre wave sensor device Industrial AoPCB Module (Figure 42).
    - The module is based on IWR6843 millimetre wave sensor as defined in [54].
    - The radar module operates at 60 – 64 GHz frequency bands as stated in [54].
    - The cost for the module is 120 dollars in [54].
    - Datasheet for the module is not publicly available.

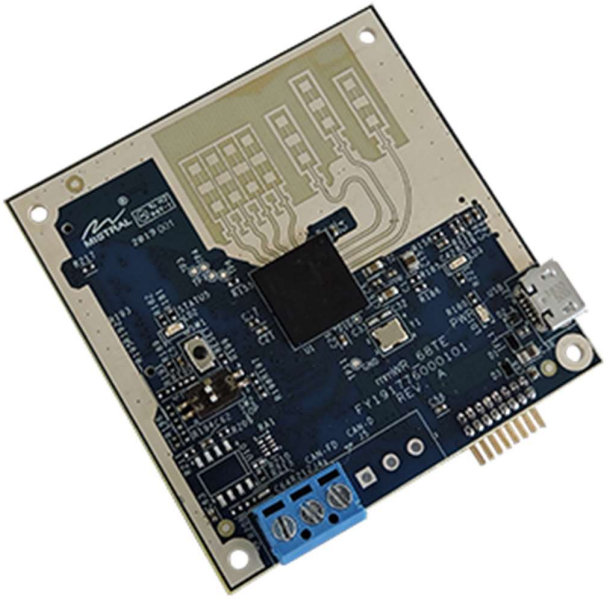


Figure 42. AoPCB Module