TALLINN UNIVERSITY OF TECHNOLOGY
SCHOOL OF ENGINEERING
Department of Electrical Power Engineering and Mechatronics

FURTHER DEVELOPMENT OF THE FLEXIBLE TRACK ROBOT

PAINDUVA ROOMIKROBOTI EDASIARENDUS

MASTER THESIS

Student: Daniil Valme

Student code: 204810MAHM

Supervisor: Anton Rassõlkin, professor

Tallinn 2022

AUTHOR’S DECLARATION

Hereby I declare, that I have written this thesis independently.
No academic degree has been applied for based on this material. All works, major viewpoints and data of the other authors used in this thesis have been referenced.

“18” May 2022

Author: ....
/signed digitally /

Thesis is in accordance with terms and requirements

“.......” .... 20….

Supervisor: ….........................
/signature/

Accepted for defence

“.......”....................20… .

Chairman of theses defence commission: ....
/name and signature/

Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I, Daniil Valme

1. grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis

"Further Development of the Flexible Track Robot",

supervised by Anton Rassõlkin

1.1 to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

1.2 to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2. I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.

3. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

[1] *The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.*

Department of Electrical Power Engineering and Mechatronics

THESIS TASK

Student: Daniil Valme, 204810MAHM
Study programme: MAHM, Mechatronics
Supervisor(s): Anton Rassõlkin, Professor

Thesis topic:
(in English) Further Development of the Flexible Track Robot
(in Estonian) Painduva roomikroboti edasiarendus

Thesis main objectives:

1. To optimize the mechanical design of the first porotype of the track robot
2. To modify the electrical design of the robot including motors
3. To develop the control system of the robot
4. To select sensor for the track module, design mountings and create preliminary algorithm
5. To build the second porotype

Thesis tasks and time schedule:

No	Task description	Deadline
1.	Literature review	24.12.21
2.	Mechanical design of the robot	28.01.22
3.	Selection of the motors, updating the electrical circuit	11.02.22
4.	Design of the control system	04.03.22
5.	Selection of the sensors, mounting and algorithm development	18.04.22
6.	Building the prototype	08.04.22
7.	Testing the prototype, optimization	15.04.22
8.	Summarising results and writing report	13.05.22

Language: English **Deadline for submission of thesis:** "18" May 2022

Student: …………………….. ……………….. "……"………………………………20….a
/signature/

Supervisor: ………………… …………………….. "……"…………………20….a

/signature/

Consultant: ………………… ……………………… “…….”…………………20….a

/signature/

Head of study programme: …………… ………………… “…….”…………………20…..a

/signature/

Terms of thesis closed defence and/or restricted access conditions to be formulated on the reverse side

CONTENTS

PREFACE

This thesis describes the further development of the reconfigurable continuous track robot previously developed by author in cooperation with the student of Tallinn University of Technology Diana Belolipetskaja.

While writing the thesis the author learnt much about the design of the mobile robot and modern approaches used in their design.

The author would like to express gratitude to the supervisor professor Anton Rassõlkin for the mentoring.

List of abbreviations and symbols

3D	3 Dimensional
2D	2 Dimensional
ABS	Acrylonitrile Butadiene Styrene
CoM	Centre of Mass
DC	Direct Current
DoF	Degree of Freedom
FOV	Field of View
GNSS	Global Navigation Satellite Systems
IR	Infrared
LiDAR	Light Detection and Ranging
Li-ion	Lithium-Ion
Li-Po	Lithium Polymer
MPT	Motion Prediction Technique
PID	Proportional Integral Deferential
PLA	Polylactic Acid
PWM	Pulse-width Modulation
Radar	Radio Detection and Ranging
RCTR	Reconfigurable Continuous Track Robot
RGB	Red Green Blue
ROS	Robot Operating System
rpm	Revolutions per Minute
SLAM	Simultaneous Localization and Mapping
Sonar	Sound Navigation and Ranging
TOF	Time of Flight
TPE	Styrenic block copolymer
UGV	Unmanned Ground Vehicle

1 INTRODUCTION

In recent decades, land transport vehicles and robots have achieved significant technology development that has led to miniaturization and improvement of computers, sensors, and networking performance, which are the cornerstone of mobile robotics [1]. Mobile ground robots find their implementation in different applications due to the ability to adopt robots for specific tasks. Various solutions are proposed for transportation, cleaning, agriculture, space exploration, and service.

Several mobile robots have been developed to work in extreme environmental conditions, which can cause potential risks to human health and safety. These robots are widely used for inspection [2], mine mapping [3], nuclear operation [4], bomb squad, planetary explorations [5], and search-rescue operations [6]. Those robots can provide the operator with information from non-accessible or hazardous areas such as gas-filled tunnels, unstable ground, damaged structures, large cervices, and contaminated locations [7]. Robots can detect leakages, hot spots, gas leaks, and deteriorating machines before a significant problem occurs due to their ability to perform high accuracy measurements [2].

One of such robot types is the reconfigurable continuous track robot (RCTR). The main idea behind the RTCR concept is to enhance the tracked locomotion system advantages by adding reconfigurable robot properties. In the traditional approach, the track is considered as one element of a system, the configuration of which is controlled by external elements such as pulleys. In contrast, in the RTCR concept, the track is considered as the mechanism, where each element is presented as a link that can be controlled separately. The reconfigurable structure of the robot can enable it to crawl over obstacles and adapt its external shape according to the environment without using external moving parts or additional modules. RCTR robots have equal pressure dissipation, the ability to move in uneven terrain, high traction potential, and the ability to climb on high obstacles. Compared to other reconfigurable or modular track robots, the RCTR design has fewer actuators. It increases reliability, simplifies control, and decreases the weight and size [8].

The prototype of the RCTR robot controlled by Arduino Uno Rev3 microcontroller board [9] was developed from scratch by the students from Tallinn University of Technology [10], [11], and was inspired by the design and concepts presented by T. Kislassi and D. Zarrouk [8]. The developed platform uses two 12 V DC motors to generate propulsion and one servo drive to control the inclination angle of the track links. The robot can move in two directions and has a length of 542 mm, a width of 139 mm, and a height of 134 mm. Links of the track have a design that allows selecting the angle between two adjacent links using a rotating pin mechanism. Each track link can have three states: locked at 0 degrees, locked at 15 degrees, and unlocked. The developed prototype is controlled via a Bluetooth module, or it can be preprogrammed for a specific task. The final goal of the project is to create a tank-line chassis, which can pass complex obstacles by adopting its shape.

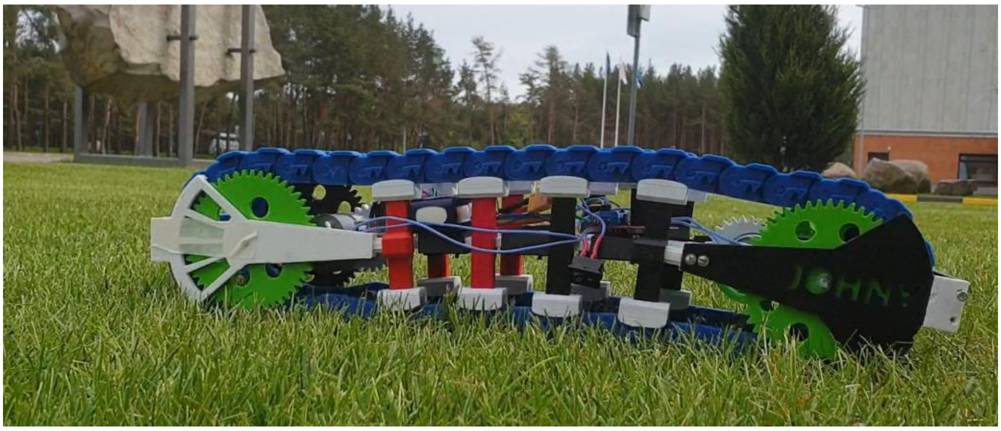

Figure 1.1 First prototype of the RCTR [11]

The goal of this thesis is to further develop the first prototype of the RCTR based on the test results and approaches most used commonly in mobile robotics:

- Optimize the mechanical design of the robot;
- Upgrade thrust and electrical systems of the robot;
- Further develop control system of the robot;
- Prepare the platform for the implementation of sensors.

2 LITERATURE REVIEW

2.1 Mobile robots for difficult terrain

Moving in difficult terrain conditions has always been one of the critical challenges for mobile robotics [1]. In specific applications, mobile ground vehicles are also called unmanned ground vehicles (UGV) [12]. Difficult terrain for mobile robots can be described as an environment with a large number of unique undefined obstacles. Robots must have good passability to pass such obstacles, which is often achieved by implementing complex kinematic structures with a high number of Degrees of Freedom (DoF) [5]. Control of such mechanisms requires precise information regarding the environment using sensors. An example of such a robot is MARC modular autonomous adaptable robot presented in Figure 2.2.

According to the movement principle, several robotic concepts commonly used for harsh terrain conditions can be classified: wheeled robots [13], legged robots [14], [15], [16]; track robots; reconfigurable robots [17]; and hybrid robots [18], [6], [19], [20], [21]. Tracked and wheeled robots show good performance in environments with even terrain; however, they are not capable of passing high obstacles.

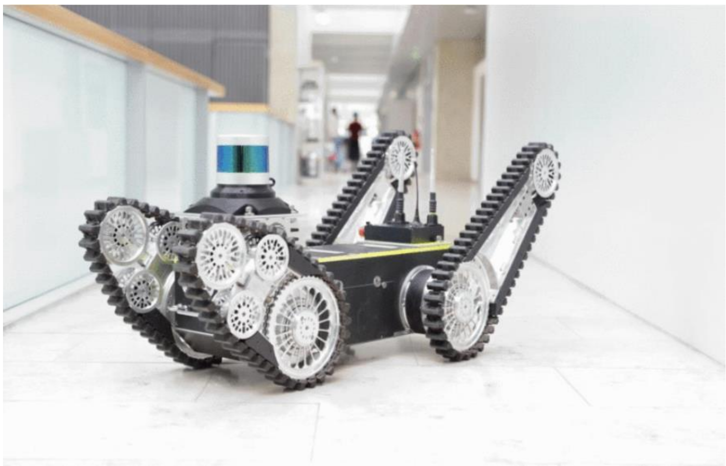

Figure 2.1 MARC - Modular Autonomous Adaptable Robot [20]

In the past, the research community has put a lot of effort into increasing the level of autonomy of such robots [14]. The critical reasons for the surge in the efforts are communication issues [14] or the desire to automate specific processes and decrease the level of human involvement [4], [18]. The problem of interrupting signals is especially crucial in the areas without communication networks, such as the underground environments. In such conditions, the operator cannot control the robot remotely, so the robot must be able to navigate autonomously. This problem is being addressed in research works, and many solutions were proposed during DARPA Subterranean Challenge [22]. DARPA competition aims to propose new technologies that augment operations underground [14]. Increasing the level of autonomy of the robot is very often related to the reduction of operating costs. An example of such an approach is the Starship last-mile delivery robots in urban environments [18]. In this fully developed version, the remote-control supervisor in the command center has only to be involved a small part of the time in complex situations [18]. The next example of automation in mobile robotics is ANYmal four-legged robot. The main task of ANYmal is autonomous surveillance tours of specific locations of the offshore wind plant. Similar to the Starship robot, the ANYmal robot is also not fully autonomous and requires the attention of the operator to analyze complex tasks.

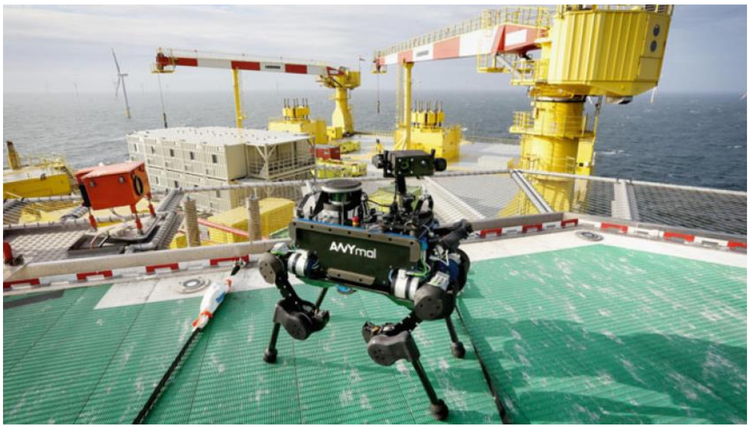

Figure 2.2 ANYmal legged inspection robot [2]

Autonomous obstacle avoidance is crucial for robot locomotion in complex environments. Using information obtained from the sensors, the obstacles can be defined by three parameters: their geometry, state, material, and structure. Based on the shape of the object, it is possible to distinguish two types of obstacles based on their profile: positive and negative. In the real world, the positive obstacles can be presented as steps, stones, boxes, or hills, and the negative can be presented as gaps, potholes, ditches, and crevices [1]. To estimate the traversability of both, the robot should be able to measure their sizes and compare them with the threshold values. Those values are defined by the technical parameters of a particular platform.

Based on the state of the obstacle, it is possible to distinguish two types: dynamic and static [23]. In the case of static obstacles, the position of the obstacle does not change. In the case of dynamic obstacles, the position of the object may change, what makes obstacle avoidance more complex and requires advanced motion prediction techniques

(MPT) to be applied. In certain conditions, it becomes essential to estimate not only the state and shape of the object but its material and structure as well. It becomes crucial, especially in the case of off-road terrain, where it is crucial to understand whether the structure of the terrain is traversable [16], [24].

2.2 Environment perception in mobile robotics

Designed obstacle avoidance system must consider the kinematic properties of the RTCR robot type described in [25], the characteristics of the propulsion system [10], and the mechanical design of the robot [11]. The developed solution will be verified on the test tracks, where both types of obstacles are presented.

Autonomous navigation of the robot requires precise information regarding the surrounding environment, which is processed and used for motion planning and navigation [26]. The autonomous vehicle observes information regarding the environment using sensors. Sensors must be able to create both a perceptive and locational view of the environment so that the vehicle can make decisions in real-time. Quick updating and accurate information regarding the current location of the vehicle must be fetched so that the necessary information can be sent to the control algorithms. Navigation in an environment requires detailed information such as distance to the closest objects, type of the object, speed, and color, for which multiple sensors are used [26]. Different sensor types are used in parallel To minimize the probability of errors for specific environment parameters. By the type of parameters being observed, the sensors can also be defined as exteroceptive and proprioceptive [27].

Exteroceptive sensors are used for perceiving the environment, for calculating the distance to an object, and defining its dimensions or material. Examples of exteroceptive sensors are LiDAR, sonar, radar, visible light (VIS) camera, infrared light camera (IR), depth cameras, and hyperspectral cameras [24], [26]. A comparison of exteroceptive sensors used for different mobile platforms is presented in Table 2.1.

The sonar sensors use low-cost and user-friendly technology for the operation [28]. For robots, ultrasonic sensors are used primarily for measuring short distances at low speeds as proximity sensors [26]. The significant issues incurred while using sonars are caused by the width of the sonar beam, as it leads to poor directional resolution.

Ultrasonic sensors also show low performance in the case of smooth surfaces as they do not produce detectable echoes [29].

RGB cameras are very often used in mobile robotics nowadays due to their low-cost, high-quality color information and high resolution [14], [2], [30]. They are used chiefly for environment capturing, object detection, and recognition. Infrared cameras work in infrared (IR) wavelengths, and they are used to detect the temperature of the objects or in situations where peaks of illumination may occur [26]. In addition to detecting and recognizing obstacles, such sensors for robots are used mainly for inspection purposes [2].

LiDAR and depth camera sensing technologies provide depth information about the surrounding environment. Compared to RGB cameras, they are more reliable as they do not depend on lighting conditions. Various methods are put forward on point cloud processing to deal with multifarious tasks, especially on object detecting and classifying [31]. In existing mobile robotics solutions for harsh environmental conditions, various combinations of 3D rotating LiDAR, 2D rotating LiDAR, and RGD-D cameras are usually used. In [2], [14], [30] 3D rotating LiDARs placed on the top of the robots have a more extensive range. Those LIDARs are responsible for the localization of the robot and mapping. Depth cameras are commonly used to provide depth information around the robots at shorter distances [14], [32]. Sometimes, robots are equipped with a single beam LiDAR. In such cases, the distance to a single point on the surface can be estimated.

Radars show good performance in bad weather conditions like snow and rain. However, radars cannot define the shape of the object due to low resolution and may struggle with a particular material [26]. For that reason, they are not often used for estimation of the dimensions of obstacles, however, specific solutions are still present [30].

Table 2.1. Comparison table of exteroceptive sensors used in different solutions

Type of the sensor	ANYmal [2]	Hexapod [14]	Absolem [14]	RoboSimian [5]	Hulk [30]
Ultrasonic sensor	-	-	-	-	-
Lidar	+	-	+	-	+
VIS camera	+	+	+	+	+
Depth camera	+	+	+	-	-
Radar	-	-	-	-	+

Proprioceptive sensors are used to measure values from within the system. They can give information regarding the position of different degrees of freedom, acceleration, temperature, the voltage level of critical components, and force applied to an effector or motor current. Proprioceptive sensors are usually passive sensors [26], [27].

Examples of proprioceptive sensors are: Inertial Measurement Unit (IMU), tactile sensors, encoders, Global Navigation Satellite Systems (GNSS), force sensors, and torque sensors. A comparison of proprioceptive sensors used in five different mobile platforms is presented in Table 2.2.

Usually, most robots are equipped with IMU sensors that provide accurate angular rate and acceleration data [2]. In the case of legged and wheel-on-limb robots, each joint is equipped with encoders and joint-torque sensors [5], [33]. This is done to get instant feedback regarding the position and state of each joint that can be used for further motion planning [16], [5]. In the case of ANYmal robot, 3D force sensors integrated into rubber feet provide haptic information regarding contact with the ground. GNSS provides accurate 3D object positions by a global satellite system and the receiver (i.e., GPS, Galileo, and GLONASS). GPS keeps the vehicle on its intended route with an accuracy of 30 centimeters [26].

Table 2.2.Comparison table of proprioceptive sensors used in different solutions

Type of the sensor	ANYmal [2]	Hexapod [14]	Absolem [14]	RoboSimian [5]	Hulk [30]
IMU	+	+	+	+	+
GNSS	+	-	-	-	+
Encoder	+	-	+	+	+
Force	+	+	-	+	-
Torque sensor	+	+	-	+	-

2.3 Data processing and traversability estimation

After the information regarding the surrounding environment is observed by sensors, it must be fused. Sensory fusion or data fusion aims to improve the measurement of two or more sources of data from sensors beyond the individual measurement of each of them. Sensory fusion is especially required in situations when large amounts of disparate sensor data are produced. Sensorial fusion applied to the measurement of redundant data reduces the uncertainty of the measurements, increases the accuracy, and improves the integrity of the system [26].

Fusion methods mostly applied in robotics are very often based on probabilistic methods, which are indeed now considered the standard approach to data fusion in all robotics applications. Probabilistic data fusion methods are generally based on Bayes' rule for combining prior and observation information. Practically, this may be implemented in several ways: through the use of the Kalman and extended Kalman filters, through sequential Monte Carlo methods, or using functional density estimates [34]. The fused data is used as the input for the process called Simultaneous Localization and Mapping (SLAM). SLAM algorithms compute the likelihood of both the pose the and the environment using onboard sensors.

Data fuse techniques are used for generating traversability maps. In the case of wheeled and hexapod walking robots described in [14], all points from incoming point clouds that do not belong to the top modelled surface of the terrain are filtered out. The filtered point cloud is then fused with the elevation map using one dimensional Kalman filter. A similar approach is used for the legged robot described in [35] and the track robot with two active sub-tracks [6]. In the case of ANYmal robot [2], 3D point cloud data from 3D LiDAR is fused with depth information from depth cameras. To obtain information regarding object sizes, other techniques can be used as well. Some of them use only camera images to extract the actual distance to the object, width, and height based on any given bounding box from the captured frame [36].

Traversability of the environment can be estimated through analysis of data obtained from the environment, one of the approaches is proposed in [35]. In this method, the data obtained from the sensors is used to create a robot-centric traversability elevation map of the terrain. The traversability map locally describes three different terrain characteristics: local terrain slope (s), local terrain roughness (r), and local step height (h). Such an approach to defining the environment is reasonable in the case of complex environments like steps, gaps, or corridors. In Equation 2.1, W_1, W_2, and W_3 are the weights that sum up to one. The critical values s_{crit}, r_{crit}, and h_{crit} are the maximum allowed values. In case one of the terrain characteristics exceeds its critical value, the traversability is set to 0 [35]. Figure 2.3 is a visualization of a traversability map.

$$t = 1 - W_1 \frac{s}{s_{crit}} - W_2 \frac{r}{r_{ctri}} W_3 \frac{h}{h_{crit}} \tag{2.1}$$

where t – traversability index

W_1 – weight factor for local terrain slope,

W_2 – weight factor for terrain roughness,

W_3 – weight factor for local step height,

s – local terrain slope,

r – terrain roughness,

h – local step height

s_{crit} – local terrain slope,

r_{crit} – terrain roughness,

h_{crit} – local step height.

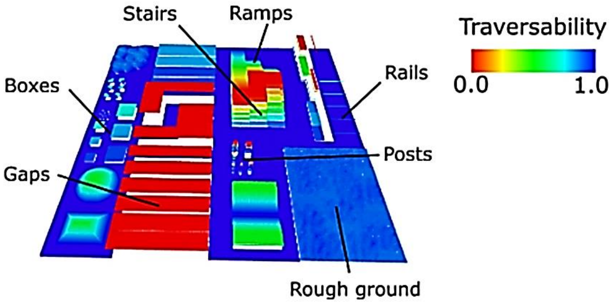

Figure 2.3. Traversability map shows the different obstacles, the cells are colored corresponding to their elevation values [35]

2.4 Motion planning

After the traversability estimation is done, the path should be generated. The task of path generation is to find the shortest path from point A to point B [2], [14]. There can be two types of path planning algorithms defined: offline path planning (algorithms are generated prior to execution) and online (the algorithms work online) [37]. The next process performed after path planning is motion planning. Motion planning is primary and one of the most complex robotic tasks that aims to provide collision-free motions for complex bodies along the path. In most forms of motion planning, the robot is not allowed to touch obstacles [37].

There are three aspects of the robot that must be considered during motion planning: the shape of obstacles, kinematics, and dynamics of the particular robot. Kinematics is a geometric problem that involves the representation of the vehicle configurations in a collision, given the admissible trajectories. Dynamics involves the accelerations and temporal considerations. It is derived from two aspects. The first is choosing a control reachable in a short period of time given the current velocity and the maximum accelerations. The second - is taking the braking distance into account so that after a control execution, the vehicle can always stop before collision by applying the maximum deceleration [25], [38].

RoboSimian [5] is a hybrid robot with actively articulated suspension presented in Figure 2.4. For motion planning of flexible limbs, it requires specific motion planning approaches. The controller of the platform uses a pre-computed lookup table to calculate its inverse kinematics. The inverse kinematics solver is used to ensure computationally efficient generation of safe paths through the 32 DoF joint space of the robot. The computational complexity of the proposed architecture is minimal and can be generalized. The main task of the controller is to maintain the roll and pitch of the chassis relative to the robot-centric gravity-aligned. The output of the controller is a required trajectory for the endpoint of each limb [5].

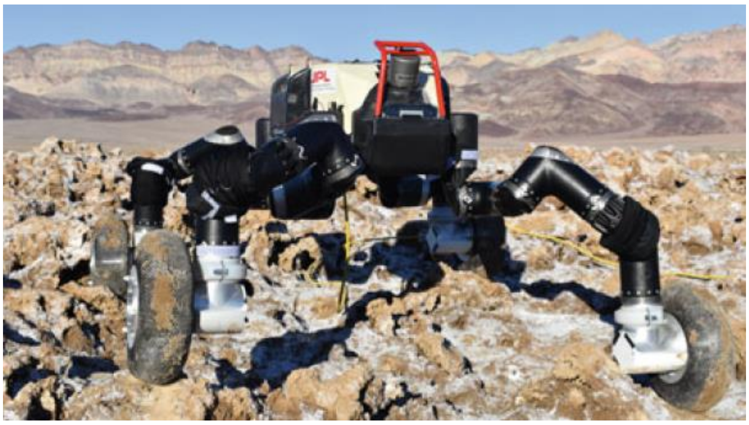

Figure 2.4. RoboSimian robot with actively articulated suspension [5]

To overcome unknown obstacles, the ANYmal robot relies on its perception capabilities to safely plan its footholds and adapt its body pose. If obstacles are known as steps, special maneuvers can also be preprogrammed. To inspect a checkpoint, the robot walks to a defined viewpoint in the 3D map and adapts its posture if required. The robot uses different gaits for locomotion depending on the terrain and speed. The robot employs a trotting gait on smooth and flat terrain for fast walking up to 1.0 m/s [33].

In the case of the RCTR robots, the robot can change its shape by changing the inclination angle between links. This feature is used to increase maneuverability. The maximum height of the obstacle h_{max} that the robot can climb is a function of the length of the robot, its density ρ, its maximum pitch angle α_{max}, mass of the front (F_f), and rear (F_R) mechanisms. The maximum height can be reached when the center of mass (COM) is behind the last joint contact s_0 with the horizontal surface. In this case, the sum torques around point s_0 is zero [25]. Figure 2.3 represents the parameters required for the estimation of the maximum height of the obstacle.

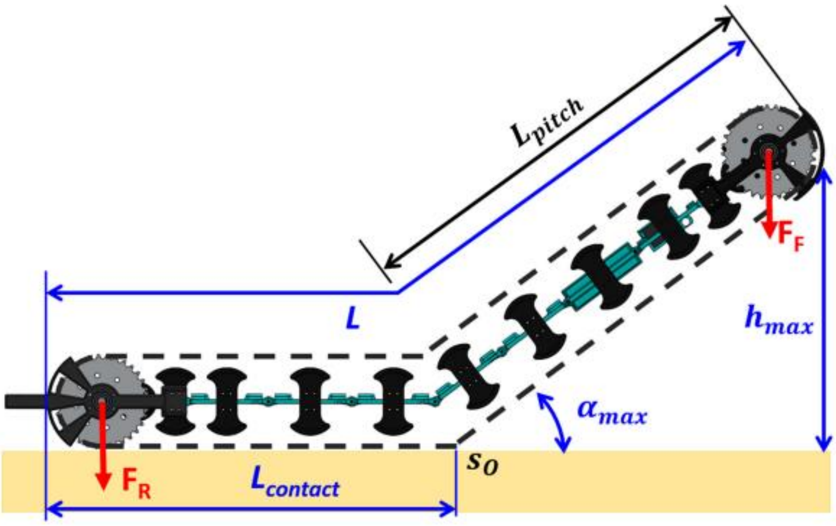

Figure 2.5 Parameters required for estimation of the maximum height that the RCTR can climb: *L* – total length of the RCTR, $L_{contact}$ – length of the robot contacting the ground, $L_{contact}$ – length of the inclined part, F_R – the mass of rear mechanisms, F_F – the mass of front mechanisms, s_0 – last contact, a_{max} – maximum inclination angle [25].

2.5 Hardware architecture

Computations for SLAM, motion planning, and path planning as well as for image processing and fuse, very often require a high amount of computational power. For that reason, multiple computers for processing are used. In the case of the ANYmal inspection robot, there are three PCs: one for navigation and motion planning, one for locomotion control, and a third one dedicated to the specific application. Such structure forms the control system of ANYmal. The data obtained from the sensors is transferred via the network by the Robot Operating System (ROS) running on a low-latency patched Ubuntu [39]. Figure 2.6 represents the architecture of the control system of the ANYmal robot. A similar approach is introduced in the RoboSimian robot [5], in which the control system of the robot is composed of three separate components: a force controller, an orientation controller, and a leveling controller.

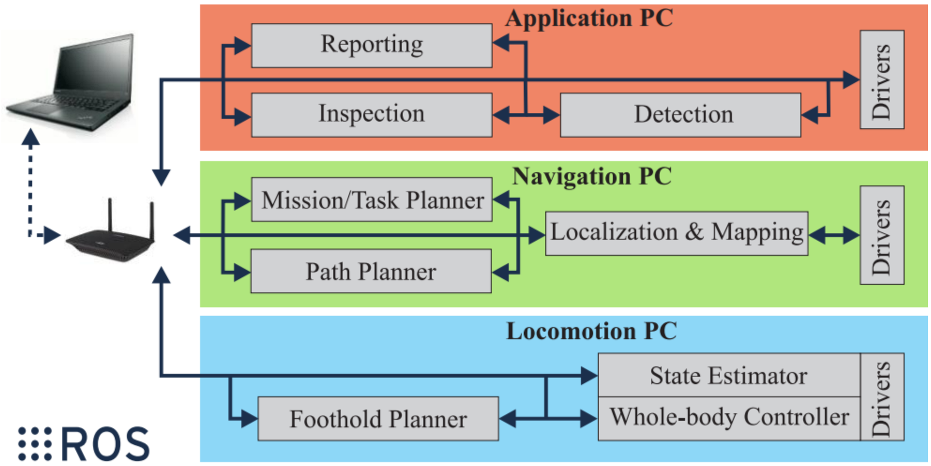

Figure 2.6. Control system architecture of ANYmal robot [39]

In publication [14] is described the hardware architecture for three types of mobile robots used in harsh terrain: wheeled, tracked, and legged. Tracked robot Absolem and hexapod robot are presented in Figure 2.7. For object detection, the Husky wheeled robot is used the NVidia Jetson TX2 board that achieves detection rates of around two frames per second. A NUC-i5 computer is responsible for localization calculations, control, and mapping. In the case of the Absolem tracked robot presented in the same paper, object detection is done by NVidia Jetson TX2. For 3D point cloud processing, navigation, mapping, localization, and exploration algorithms, execution is used Intel Core i7-based PC. In the case of the hexapod robot, map building, exploration, and object detection are performed by the Nvidia Jetson TX2 board. For object detection, three solutions use a neural network with a modified version of the YOLOv3. YOLOv3 detects the bounding box for the objects, which then is projected on the 3D map. Then the position of the object is defined by applying Kalman filtering over the temporally consistent detections.

In the case of the robot presented in paper [32] for SLAM, a CUDA-enabled single-board computer Nvidia Jetson Nano was used. Most of the image processing and data manipulation is achieved via the utilization of the NVidia Jetson Nano single-board computer. Robot Operating System (ROS) is responsible for setting up a common communication platform with a desktop PC for control, data collection, and visualization.

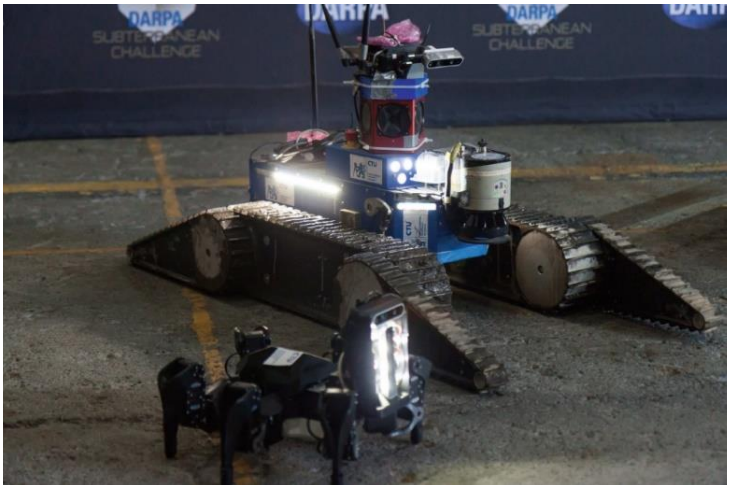

Figure 2.7. Absolem (back) and hexapod (front) platforms after a run in the mine [14]

2.6 Weight optimization

Weight optimization is typically used to decrease the weight of the detail and meet detail performance requirements. The goal of weight optimization is to safely distribute loads inside the body [40]. Reducing the weight leads to more effective use of material used in detail production and costs in additive manufacturing, also known as 3D printing. Using the additive manufacturing process to produce highly complicated structures is a novel and rapidly developing prototyping trend [41].

In the case of 3D printing technology, the smaller weight helps decrease printing time, accelerating the whole prototyping/product development process and reducing the quantity of material. In automotive design, weight reduction is one of the main ways to decrease CO2 emissions. Therefore, plenty of weight reduction methods and design guidelines were developed and defined with standards [42].

2.7 Literature review outcomes

There are several existing solutions in mobile robotics that are used in harsh or difficult environmental conditions. They are mainly used for inspection, rescue operations, planetary explorations, and mapping. The increasing autonomy level of mobile robots is primarily required for two purposes: automation of the process that is related to the reduction of operating fees and the inability to control mobile robots remotely due to communication issues in complex environments. There are current research works and competitions that aim to increase the level of autonomy of the robot. These solutions cannot be called fully autonomous, as they require the attention of the operator in case of more complex tasks.

Motion planning in the case of autonomous mobile robots is based on the information observed from the environment. This information helps to estimate the traversability of the environment and generate a path to overcome obstacles. In the case of online path planning, the robot can change its path in case of changing conditions, whereas offline path planning algorithms are generated before the execution. Motion planning of the specific robot is based on its kinematic and dynamic characteristics.

Traversability of the obstacles is defined by their physical dimensions and geometry type, dynamical properties, and the structure of their material. Existing methods estimate the traversability of the environment using information extracted from the 3D point cloud. This approach leads to some similar patterns in sensor selection in mobile robotics. 3D rotating LiDARs are usually used in mobile robotics applications as they provide precise information regarding surrounding space. However, such LIDARs require a more complex hardware architecture, including multiple both onboard and external processing units. This increases the complexity of robot architecture and requires advanced data fusion techniques due to the large amount of information obtained from the sensors. An alternative for them is depth cameras and 2D-LiDAR – the processing of the data, in this case, can be done using single board computers like NVidia Jetson TX2 or NVidia Jetson Nano. For range finding in more straightforward tasks, single beam LiDAR can be used. Despite the low accuracy, ultrasonic sensors can be used as proximity sensors for specific conditions.

In the case of mobile robots, it is essential to control the limbs of a robot with multiple DoFs precisely. For that reason, information regarding each limb is monitored using a set of sensors. A similar approach can be implemented on the RCTR robot by adding IMU sensors on the track modules.

Based on the literature review, the existing solutions can be divided into two categories based on the used sensor type. The first group relates to more advanced solutions and uses 3D-LiDARs, and requires more complex hardware. The second group of robots uses sensors producing less data as 2D LiDAR, single beam LiDAR, depth, or visual camera.

Considering the above-mentioned information, the plan for the further development was created. The final goal of RCTR robot development is to create a platform that is able to autonomously navigate in harsh environmental conditions. The chassis of the robot must consist of two flexible tracks attached to the carrier body. To achieve the decided result, the development progress is split into several phases. Overall, the steps of the development can be described as follows:

- Stage 1. This stage involves the development of a mechanical, thrust, and control system for a single track. The outputs of this stage and their quality define the quality of the next stages. The design of created track at some level must consider the requirements of the next steps. This approach helps to avoid excessive changes in future design.
- Stage 2. The second stage is mainly dedicated to the development of the perception system of the flexible track. The goal is to study the potential of integration of sensors inside the flexible track to provide the main control system with precise information.
- Stage 3. In the third stage, the main attention must be put on the development of the chassis consisting of two tracks.
- Stage 5. The fourth stage covers the selection of the hardware required for the onboard computations as well as providing the robot with sensors that can be mounted on the main body.
- Stage 4. The fourth part of the development is mainly dedicated to making the robot autonomous. This part has to cover sensor data fusion, localization, and overall optimization of the created robot.

2.8 Aim of the work

The goal of this work is to further develop the first protype of the RCTR created previously. The focus is set mainly on developing the key element of the robot – its flexible track.

Mechanical design. There were found several disadvantages in the mechanical design of the RCTR robot. Based on them the list of goals was created:

- Update the design of the track links;
- The design of the locker was not reliable enough to last long;
- Create a new design for the inclination limiting elements;
- To modify the design of the connections inside frame elements;
- To increase the overall modularity and serviceability of the platform;
- Optimize the weight of the robot.

Thrust system. During the tests, it became obvious that the thrust systems required modifications. The speed of the robot was too high, so it made the control of the robot complicated.

- To select new motors;
- To select a new driver;
- Provide the motion control unit with the feedback regrading position of the motor shaft;

Control system. The new control system must control the position of the robot based on the encoder information. The created system must present the unit that can be easily implemented during the next phases of the robot development. The task can be split into the following goals:

- Select the controller;
- Create the algorithm that considers the parameters of the selected motors;
- Design the electrical circuit.

Perception system.

- Make preliminary selection of sensors;
- Create the mountings for the sensors considering their specific features;
- Propose a preliminary algorithm for the implementation of the sensors.

The second prototype of the platform must be tested. The technical requirements for the second version of the platform are described in Table 2.3.

Table 2.3 Technical requirements for the platform

Parameter	**Required value range**
Length, mm	500 – 1000
Width, mm	80 – 150
Height of the robot, mm	100 – 160 mm
Average speed of the robot, m/s	0,08 – 0,1
Angular velocity of sprocket wheel rotation, rad/s	1,9 – 2,3
Rotational speed of sprocket wheel, RPM	18 – 22
Mass, kg	2 – 5
Acceleration, m/s^2	0,2

3 MECHANICAL DESIGN

3.1 Task description

As the first prototype was created mainly to get proof of concepts, in the design of the second prototype, the priority was switched to increasing the reliability of the robot and changing the dimensions of the robot to test the performance of the system with more free space inside the frame. This space is used to place new components inside the frame and protect them in situations when the robot bends.

During tests of the mechanical design of the robot, several disadvantages of the created prototype were found. The following elements required modifications:

- Reconfigurable track;
- The frame of the robot, placement of the structural elements;
- Shape keeping elements;
- Fastening of the parts.

3.2 Reconfigurable track

The main element of the robot of the mechanical design of the robot that had to be modified is the track link. In the first prototype of the robot, the links were connected using the spikes that were placed on the sides of the side plates, whereas the lockers were attached to the link using the metal rod. After tests, the plastic spikes on the side plates broke under pressure. During the development of the first prototype of the track, the main focus was set to make it easy to assemble and modify. So, the parts of the link were put together using a friction connection. This connection worked quite well with the small-scale robot and during not very long exploitation period. However, the connection did not last too long during the continuous load. Considering those facts, the following changes were introduced:

- The links are attached using the metal rod where the lockers are placed;
- The screws were used to connect the side plates to the middle plate;
- The design of the spikes got modified.

It was tested to 3D print the link as the solid part, such an approach did not require any connection between the parts, but this approach had several disadvantages. Due to the complex geometry of the part, the printing time of one link was about 12 hours.

Considering the fact that the track itself consists of more than 40 links, the total printing time of links only without breaks would take roughly 20 days. For that reason, it was decided to move further with modular design and to use screws. Another advantage of the implementation of screws is the ability to easily modify the width of the track by placing a new middle plate. The side plates of links were changed as well, the spikes placed on the sides of the plates were reinforced to better resist the pressure, similarly to [8].

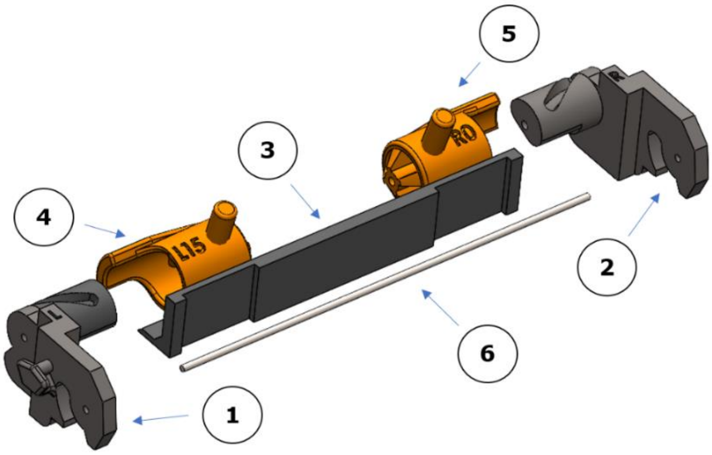

Figure 3.1 Modified track link design:
1 – left link plate; 2 – right link plate; 3 – link middle plate; 4 - 15º left locking pin; 5 - 0º right locking pin; 6 – link shaft.

The design of lockers also had to be modified. The main disadvantage of it was the size of the pin moving inside the link slot. Due to its small thickness, it was not reliable and got damaged during the tests. The design of the second prototype was created using the same approaches as in [8] and taking into account the new design of links.

3.3 Frame design

One of the main disadvantages of the frame of the first prototype was a combination of the elements in the front and the rear parts of the robot. In the first design, the support legs were placed on the first front plate, not on the front plate. It made the front part of the robot unstable while climbing the obstacles. For that reason, it was decided to put support legs on the front plate as well. The connection between the support and body plate was modified, as well new elements were added. The front side changes are introduced the Figure 3.2.

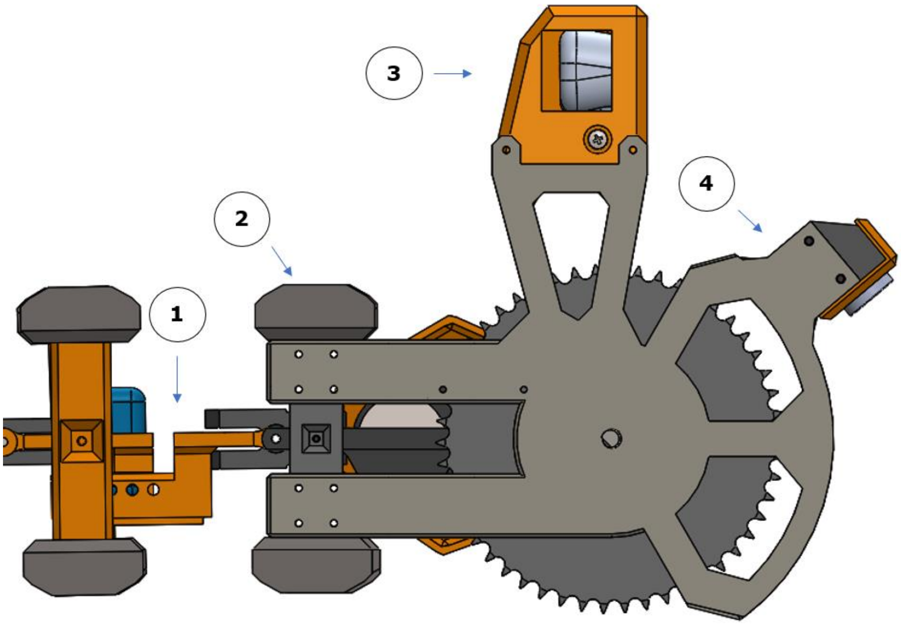

Figure 3.2 New annotation of the components in the front part:
1 – tensioning mechanism; 2 – support integrated to front plate; 3 – camera sensor mounting; 4 – front-looking sensor mounting.

The order of components inside the frame was defined by physical dimensions, their weight, and cable management. The components responsible for the control were placed close to each other to simplify the wiring inside the robot frame. Two batteries were placed close to the front and the rear part of the robot to equally distribute the weight and facilitate the climbing on the obstacles.

The plates of the body frame are attached using the hinge-like connection. These connections had to be reinforced as they got broken after long tests. The dimensions of the plates were adjusted for the new components described in Chapter 4. One of the plates of the frame was provided with the tensioning system. By changing the distance between two halves of the plate, it is possible to adjust the tension of the chain. The flexible elements that limited the movement of the robot were modified as well. In the previous version of the robot, the elements were attached to the sides of the plates. In the second version of the robot, the ribs were moved to the middle of the frame plate, by doing this, the number of ribs was reduced, and the usage of internal robot space was optimized.

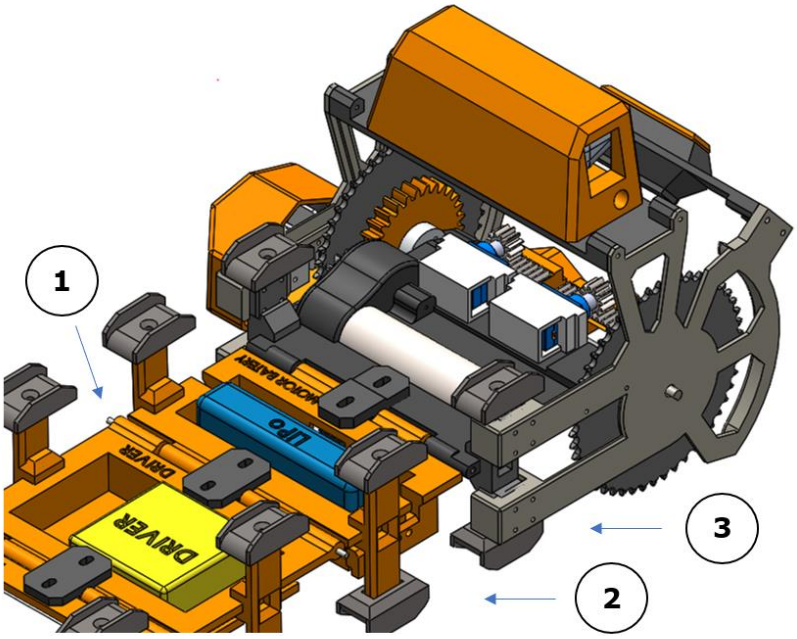

Figure 3.3 Updated design of frame components:
1 – hinge connection with flexible elements; 2 – updated support with the screw connection; 3 – updated guiding plates

All the slot fastenings were replaced with screws as the slots were impractical and required more space on the body of the robot. Changing the parts in the case of slot fasteners was complex and required more time. Also, the slot fasteners got easily damaged during the replacement of the details. The screws were used for attachment for side plates and support legs. The changes of design are introduced in the Figure 3.3.

3.4 Transmission

Several changes were introduced in transmission as well. The rotational speed of the motor selected in Chapter 4 was too high. To decrease the speed, the transmission was equipped with a custom gearbox. The transmission is made of 2 parts: a reducer gearbox and external gears. The reducer gearbox consists of 6 elements presented in Figure 3.4. The gear ratio for the gearbox is 2,5 and 2 for the external gears. The total gear ratio of the updated transmission equals 5. Such a design helped increase the motor output torque and reduced the speed of the motor.

As the gearbox had to fit the space inside the front and read plates, it needed to have small dimensions, so the module of 1,25 was used for gears. The gears are designed to be mounted on D-shafts. To fasten the driven gear on the shaft, the 4 mm aluminum fixator is used. The gearbox has mounting holes that are used to join it with both motor and frame of the robot. The external gears were designed using 1,75 module.

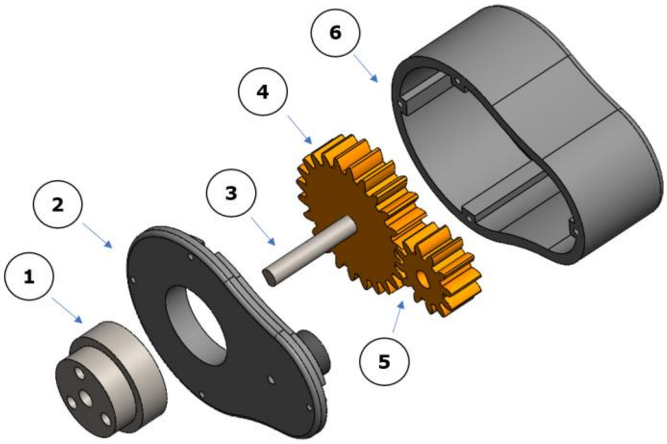

Figure 3.4 Gearbox design:
1 – gear fixator; 2 – gearbox cover; 3 – gearbox D-shaft; 4 – driven gear; 5 – driver gear; 6 – gearbox body.

The sprocket wheel and the external gears were printed as the solid part in the first prototype. Moreover, they were mounted on the main D-shaft using only friction. During the tests, it turned out that it was impossible to disassemble the shaft and integrated sprocket wheel without damaging them. For that reason, the gear and sprocket wheel were separated from each other to increase modularity, and for mounting on the shaft, special 5 mm aluminum fixators were used. The solution presented in Figure 3.5 helped improve the overall modularity of the transmission, allowing to easily adjust it for the new requirements.

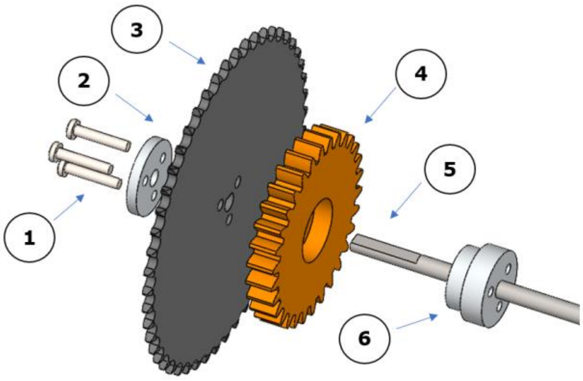

Figure 3.5 Updated sprocket wheel, gear, and D-shaft connection:
1 – fastening screws; 2 – fixator cap; 3 – sprocket wheel; 4 – shaft gear; 5 – main shaft; 6 – fixator.

3.5 Generative design

To start weight optimization of the RCTR platform, it was decided to choose three details of the robot: chain support, middle plate, and sprocket wheel. Firstly, the 3D models of the details were created. Then these parts were divided into sub-parts taking into consideration the structural loads and constraints. Figure 3.6 represents an example of input geometry definition and stress simulation view. To define input structures for generative design, there were three possible geometry types used:

- Preserve geometry is represented as green areas. To select the detailed geometry that can be changed during the generative design process, functional blocks of each part were defined. The functional block is a part of the detailed design that cannot be changed due to its functional task.
- Starting shape geometry is yellow-coloured and can be changed during the generative design creation process.
- Obstacle geometry marked with red color creates the space in which the new geometry of the details can be generated. In the case of RCTR parts, the obstacle geometry of the parts was similar to the original shape of the part due to the interaction features of the component.

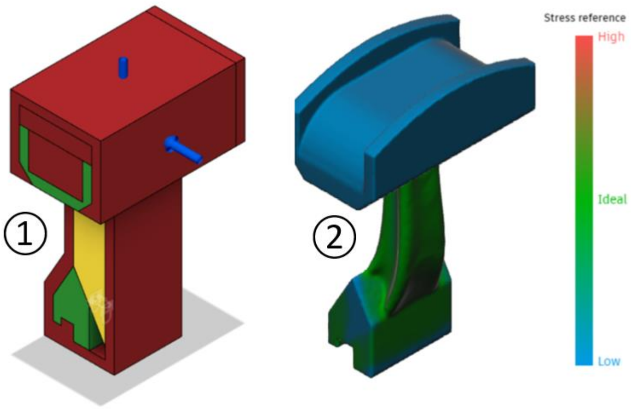

Figure 3.6 Generative design environment:
1 - example of defining input geometry; 2 - example output of stress analysis.

Each optimized detail had an individual set of loads that were calculated according to its operating conditions and functional task. Then, it was supposed to combine loads into load cases, simulating different circumstances. For generative design, acrylonitrile butadiene styrene (ABS) was selected as the material for all details. In the sprocket wheel, there were three most essential load types presented: electrical motor torque, robot weight, and sideload. To estimate the impact of the torque *M* of the DC motor on the sprocket wheel, the circumferential force was calculated, taking into consideration sprocket pitch diameter.

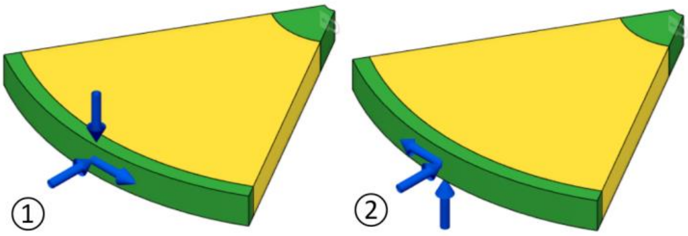

Figure 3.7 Sprocket wheels load cases:
1 – load case for the right side; 2 – load case for the left side.

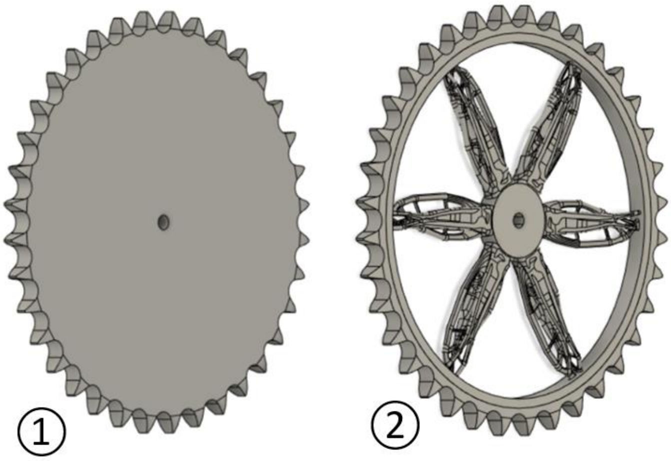

Figure 3.8 Sprocket wheel:
1 – input geometry; 2 – design using generative optimization.

The value of the gravitational force equals 5,5 N and is represented as the weight of the RCTR platform equally distributed between four sprocket wheels. Side load was added as an external load applied to the side surface of the wheel. The approximate value of the side load was calculated as half of weight 11 N. To create the design of the sprocket, there were two load cases considered, as shown in Figure 3.7. In both cases, the direction of sprocket circumferential force and side load vectors had the opposite direction to simulate different load conditions. The optimization result of the sprocket wheel is shown in Figure 3.8.

The frame plate had two loads applied: compression loads and stretching loads. As shown in Figure 3.9, load cases 4 and 6 represent the stretching load when the force of 22 N is applied to the inner surface of plate holes. Load cases 1 and 3 illustrate a similar situation but for compression loads. Load cases 2 and 5 are represented with side loads that imitate the impact of external forces affecting the side surface of the plates. The optimization result of the frame plate is shown in Figure 3.10.

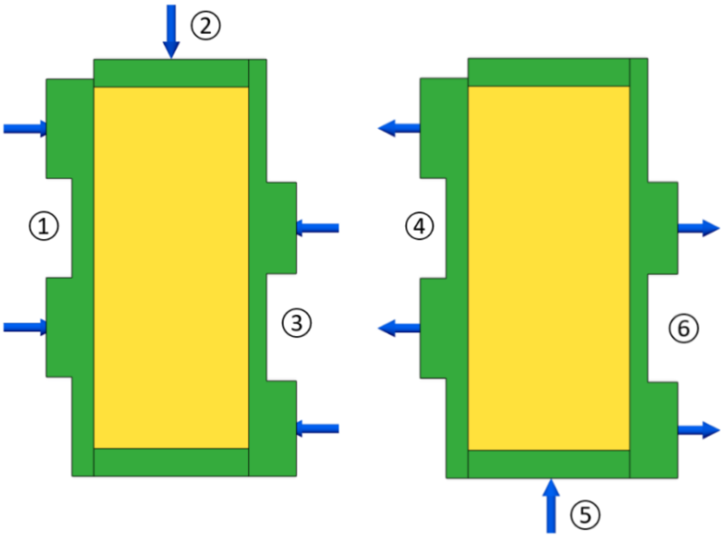

Figure 3.9 Frame plate load cases:
1 and 3 - compression loads; 2 and 5 - side loads; 4 and 6 - stretching loads.

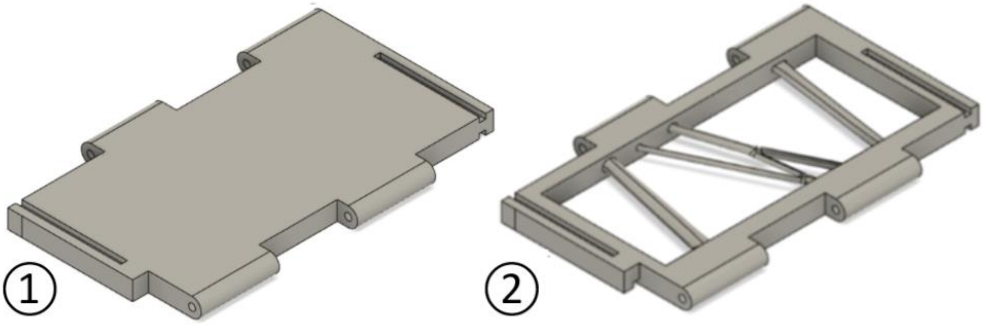

Figure 3.10 Middle plate:
1 – input geometry; 2 – design using generative optimization.

For the support chain, there were two loads taken into consideration. Weight of the RCTR platform 22 N applied on the upper surface of the detail. Also, there was a side load of 10 N applied on the side surface. Two load cases had the same structure, but the side load vectors were used on opposite side surfaces, as shown in Figure 3.11. The optimization result of the chain support is offered in Figure 3.12.

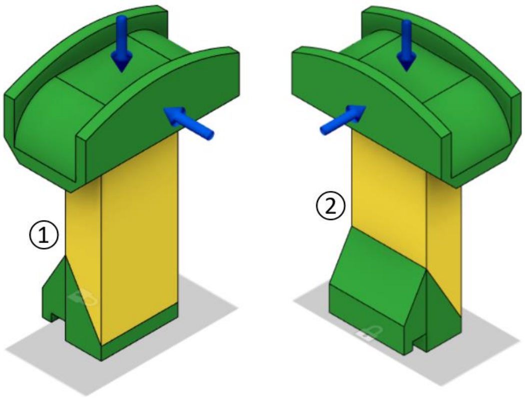

Figure 3.11 Chain support load cases:
1 – load case for the right side; 2 – load case for the left side.

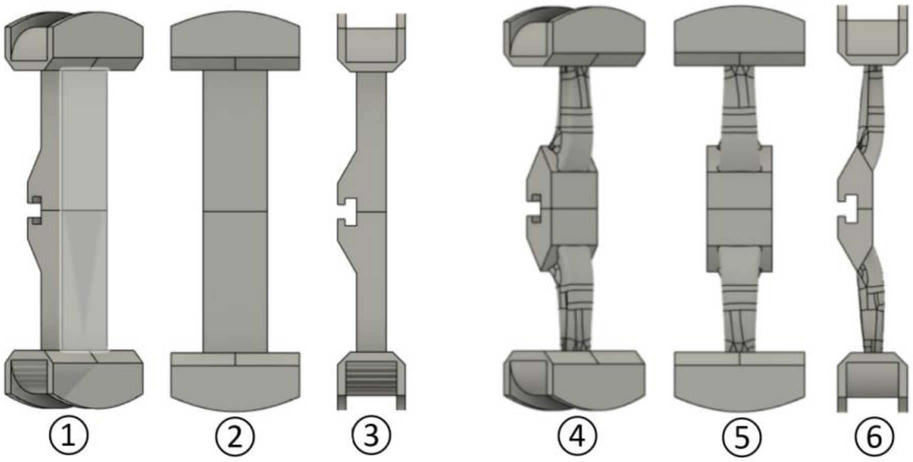

Figure 3.12 Chain support geometry:
1-3 – input geometry; 4-6 – design using generative optimization.

Two key parameters were selected to evaluate the impact of generative design implementation, such as printing time and material consumption. For calculating the printing time, specific parameters were used as input material was selected ABS, with the printing speed of 50 mm/s. For the infill rate, there were two options: 90% or 100%.

Table 3.1 Printing time

Parameter	Sprocket wheel	Body plate	Support leg
Infill rate before, %	90	90	90
Time before, h	10 h 12′	7 h 34′	4 h 9′
Infill rate after, %	100	90	90
Time after, h	9 h 15′	3 h 26′	3 h 40′
Time change, h	57′	4 h 8′	29′
Time change. %	9,3%	54,6%	11,6%

The layer height was changed from 0,2 mm to 0,12 mm as the part geometry required a more precise printing resolution to print the sprocket wheel with a modified design. Also, fewer supports were used, and the infill rate was increased from 90% to 100%. From the calculation results presented in Table 3.1, it is possible to conclude that generative design has a different influence on print time duration. Generally, the time required for printing is smaller than the original details. However, in the case of a more complex design, the time change was insignificant. Time optimization results are shown in Table 3.2.

Table 3.2 Material consumption

Parameter	Sprocket wheel	Body plate	Support leg
Infill rate before, %	90	90	90
Mass before, g	55	41	24
Infill rate after, %	100	90	90
Mass after, g	23	18	17
Mass change, g	32	23	7
Mass change, %	58,2%	56,1%	29,2%

Using of generative design had a significant influence on the masses of RCTR parts. The decrease in mass varied from 29,2% to 58,2%, accordingly. Weight optimization results are shown in Table 3.3. Applying the generative design on the selected parts led to the potential decrease of the weight of the robot in 234 grams.

Table 3.3 Assembly mass change

Nr.	Detail Parameters			
	Part name	Number of parts in the assembly	Change of the mass, g	Mass change in assembly, g
1	Sprocket wheel	4	32	132
2	Frame plate	2	23	46
3	Chain support	8	7	56
			Total:	234

4 ELECTRICAL DESIGN

4.1 Thrust system and actuators

4.1.1 Required modifications

The first prototype was equipped with Pololu 1102 motor. Initially, it was planned to control the robot without any feedback regarding the main shafts of the robot by limiting the actuation time. The idea was to find out during the tests the relation between PWM and time values in relation to travelled distance. During the tests, it became obvious that considering motor control as an open loop system was not the best approach for this platform. The main reason for that was the too high speed of the motor. It was hardly possible to make any precise conclusions based on the tests. Also, the open loop method could be used only for conditions where the external forces affecting the robot are changing over time. To implement more precise control, the motor has to be provided with feedback regarding the position of the shaft. For that purpose, rotary encoders are most commonly used.

The most unpredictable behavior of the robot was observed in the case of starting on inclined surfaces. In that case, the robot was often not able to climb the obstacle. There were several reasons affecting this. First of all, the L298N [43] motor driver is able to deliver only 2 A current per channel, while the stall current of the motor equals 3,2 A. So, the motors were not able to provide enough torque to run the robot. Secondly, the change of forces affecting the robot made the selected parameters for motor control useless.

Considering the above-mentioned facts and the changes in the mechanical design of the second prototype introduced in the previous chapters, the flowing changes in the platform had to be introduced:

- The thrust system of the robot must deliver the required performance in all conditions;
- The travel speed of the robot must be reduced;
- The motor driver must be changed;
- The motors must be provided with the feedback;
- The components of the thrust system should fit the physical dimensions of the robot.

4.1.2 Motor selection

To specify the technical requirements for the motors, the new calculations had to be done. The main requirement for the motors based on conducted tests is the ability to deliver high torque at a low speed. The additional requirement is the ability to provide feedback regarding the position of the robot, as the control system of the robot should get precise information regarding the position of the shaft and movement. This information will be used to control the state of the reconfigurable track and odometry. The methodology introduced in was used for the calculation of the required parameters of the motors. The first parameter to be calculated is the required *M* torque, it is done using Equation 4.1.

$$M = F \cdot D \cdot sin\alpha \tag{4.1}$$

where *M* – toque of the motor, Nm;

F – force, N;

D – lever arm length, m;

a – the angle between force and lever arm, degrees.

In the case of a rotational movement, the angle between the force and the force arm is 90°. The length of the power arm equals the radius of the robot sprocket wheel, which is 55 mm. To calculate the required force, it is necessary to know the weight of the robot, the maximum speed at which the robot will move, the acceleration, and the maximum angle of climbing. Those parameters are defined in Table 2.1. It is also necessary to consider the frictional force F_F that affects the movement of the crawler. As the operating environment of the crawler robot may vary, the coefficient of friction *μ* between concrete and rubber is considered, as it has the highest coefficient of friction and can consider all other materials. The coefficient of friction between concrete and rubber is 0,85.

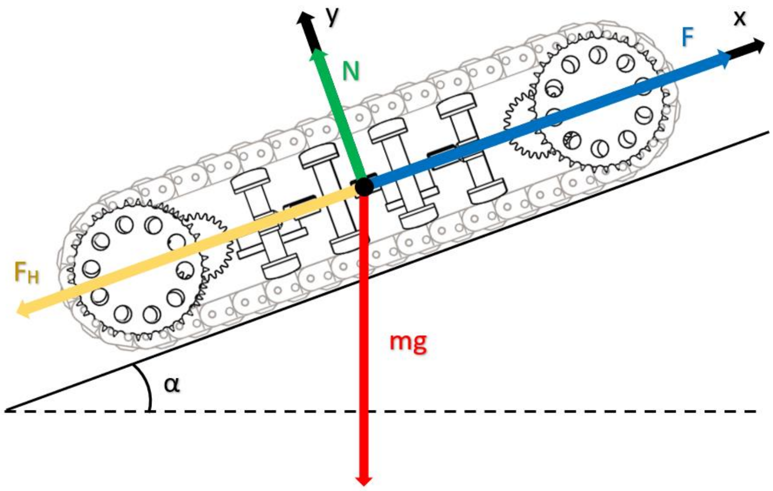

Figure 4.1 Forces diagram of the RCTR [44]:
mg (red) – force of gravity; N (green) - normal force; F (blue) - driving force acting parallel to the surface; F_F (yellow) - friction force.

To calculate the force providing the movement, all the forces acting on the body during the movement shown in Figure 4.1 shall be considered. The vector sum of all forces is equal to the product of body mass and acceleration, based on Newton's second law, Equation 4.2.

$$\vec{F_s} = m \cdot \vec{a} \tag{4.2}$$

where $\vec{F_S}$ – vector sum of applied forces, N,

m – mass of the body, kg,

D – lever arm length, m,

$\vec{\alpha}$ – acceleration of the body, m/s^2.

Another force affecting the robot is the friction force that is calculated using Equation 4.3.

$$F_F = \mu \cdot F_N \tag{4.3}$$

where F_F – friction force, N,

μ – friction coefficient,

F_N – normal force, N.

The final Equation 4.4 for the total force considering the force of gravity, normal force, friction force F_F, and driving force F_D, is introduced in [10], for the calculations was also used gravitational acceleration constant with the value 9,8 m/s^2.

$$F = m \cdot (a + \mu \cdot g \cdot cos\alpha + g \cdot sin\alpha) \tag{4.4}$$

$$F = 3{,}5 \cdot (0{,}02 + 0{,}85 \cdot 9{,}8 \cdot cos45° + 9{,}8 \cdot sin45° = 45{,}6\ N$$

where g – gravitational acceleration constant, m/s^2.

Based on the calculated values, the total torque delivered by two motors is:

$$M = F \cdot D = 45{,}6\ N \cdot 0{,}055\ m = 2{,}5\ Nm$$

So, each motor should be able to deliver 1,25 Nm of Torque. Selection of the motor also requires the calculation of the motor power. The angular speed of the sprocket wheel, accordingly to the Table 2.3 should be equal 2 rad/s. The power value was calculated using Equation 4.5.

$$P = M \cdot \omega = 1{,}25\ \cdot\ 2 = 2{,}5\ W \tag{4.5}$$

where P – the power of the single motor, W,

ω – angular speed, rad/s.

Table 4.1 Requirements for single motor

Motor parameters	Required values
Power, W	2,5
Minimal Torque / motor, Nm	1,25
Motor length, mm	100
Motor diameter, mm	50
Rotational speed, rpm	18 – 22
Supply voltage, V	12

It was decided to use Pololu 4847 high-power DC motor with a metal gearbox integrated [45]. The motor is equipped with 48 counts per revolution quadrature magnetic encoder on the motor shaft. The total length of the motor is 85 mm, and the diameter of the body is 24,4 mm. The motor has a D-shaped shaft with the diameter of 4 mm in diameter and the length of 12,5 mm from the face plate of the gearbox. Based on the recommendation displayed in the motor datasheet, the motor operating current should be equal to up to 25% of the stall current value. To guarantee safe performance, for the calculations, it was decided to use values of parameters of the motor performing at maximal efficiency. Table 4.2 Pololu 4847 motor describes parameters.

Table 4.3 Required parameters

Motor parameters	Max efficiency values	Upper limit values for maximal efficiency range
Torque, Nm	0,45	0,85
Power, W	4	6,2
Rotational speed, rpm	87	74
Current, A	0,94	1,25
Stall current, A	5	5

The maximal speed of the motor is 100 RPM. To decrease the speed of rotation and increase the torque, it was decided to implement the gearbox in transmission with the ratio of 5. To do that, Equation 4.6 was used.

$$n_T = \frac{\omega_{in}}{\omega_{out}} = \frac{M_{out}}{M_{in}} = \frac{RPM_{in}}{RPM_{out}} \tag{4.6}$$

where n_T – total gear ratio,

ω_{in} – input angular velocity, rad/s,

ω_{out} – output angular velocity, rad/s,

M_{in} – input toque, Nm,

M_{out} – output toque, Nm.

So, based on the Equation 4.6, the output rotational speed can be calculated.

$$RPM_{out} = \frac{RPM_{in}}{n_T} = \frac{100}{5} = 20$$

Equation 4.6 was used for the calculation of output torque as well.

$$M_{out} = M_{in} \cdot n = 0{,}45 \cdot 5 = 2{,}25\ Nm$$

To control the motors, the DRI0041 driver was selected. It can be supplied with the DC voltage within the range of 7 to 24 V and provide each motor with 7 A of continuous current. For control, the driver requires a PWM signal, the minimum valid pulse width is 5 us. The length and width of the driver are 55 mm, whereas the height is 25 mm. The driver requires a voltage reference of 3,3 V or 5 V from a microcontroller. To insulate the board, the driver was covered with a layer of caption tape.

To control the locking mechanism, it was decided to use two servos instead of one. The main reason for that is the increased width of the robot, so one servo is not able to quickly lock pins in both directions

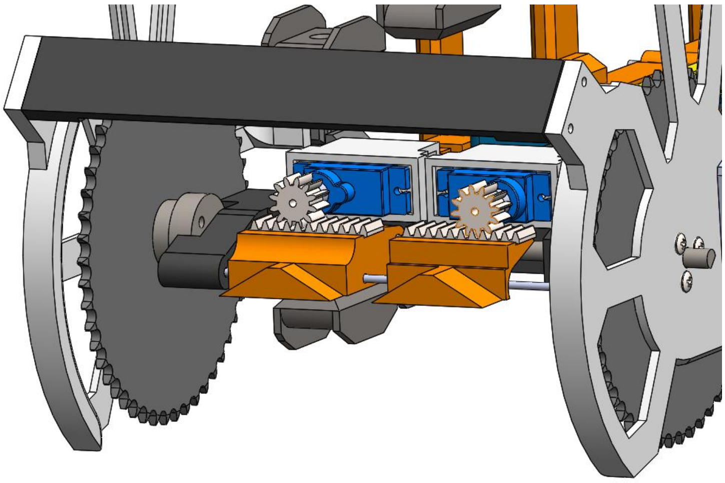

Figure 4.2 Updated design of the locking mechanism

4.2 Electrical system

4.2.1 Power system

Based on the results of the tests, it was decided to extend battery capacity. The main reasons for that were the implementation of new components, change in the robot design, and the desire to increase the autonomy of the robot. As the dimensions of the robot were changed, it gave the opportunity to use different types of batteries.

Similar to the first porotype, preference was given to the lithium batteries. In addition to the previously selected lithium polymer (Li-Po) battery [46], a block of two series-connected Li-ion 18650 P1834J batteries was added [47]. The batteries produced by KeepPower, have a typical capacity of 3400 mAh, a normal voltage of 3,7 V, and a maximum discharge current value of 6,8 A. The elements are equipped with in-built safety circuity that has protection against overdischarge, overcharge, overcurrent, and short-circuit. The main advantage of the battery element with an in-built protection circuit is the ability to save space inside the robot. To mount the batteries inside the frame, a special holder was used. As the Li-Po battery is able to deliver a higher discharge current, what is crucial, especially in case of high stall currents. It was decided to use Li-Po to power DC motors, whereas 18650 batteries are used to power microcontrollers, servos, and sensors.

Considering the above-mentioned changes, the new calculations were done. The separate working time was calculated for both the main propulsion power system and power systems of the control elements. The current consumption of the components is described in Table 4.4.

Table 4.4 Current consumption of the components

nr	Component	Average current consumption	Peak current consumption	Qty
1	Pololu 4847 DC motor [45]	940	5000	2
2	Arduino UNO REV3, mA [9]	800	800	1
3	Micro Servo SG90, mA [48]	100	360	2
4	DC motor encoder, mA [45]	50	50	2
5	HC-06 Bluetooth module, mA [49]	8	40	1
6	DFR0205 DC-DC power module, mA [50]	5	5	1
7	DRI0041 motor driver, mA [51]	22	22	1

To calculate the working time of the systems, Equation 4.7 was used.

$$t = \frac{C}{I_k} \tag{4.7}$$

where t – the working time of the system,

C – battery capacity, mAh,

I_k – the total peak current of the system, mA.

$$t_{PS} = \frac{1300}{2 \cdot 940} = 0{,}69\ h$$

where t_{PS} – the working time of the propulsion system, h.

$$t_{CS} = \frac{C}{I_k} = \frac{3250}{800 + (360 \cdot 2) + (50 \cdot 2) + 40 + 5 + 22} = 1{,}93\ h$$

where t_{CS} – the working time of the control system, h.

Accordingly, to the calculations, the Li-Po battery is able to provide the main propulsion system with power for about 40 minutes which is quite an appropriate time, whereas the 18650 Li-ion batteries can deliver power to the control system of the robot for about 115 minutes. It means that this capacity difference can be used to implement sensors in the future.

4.2.2 Connection diagram design

In the design of the new circuit, there were considered the drawbacks of the first design. One of the most crucial disadvantages was the inability to monitor the discharge level of Li-Po. In the second prototype, the robot is provided with a separate digital voltmeter module that indicates the battery level on the screen. In some cases, during the tests, the robot behaved in an unpredicted way. As the main control switch was placed inside the frame of the robot, it was not able to turn the robot off. For that reason, the new prototype is equipped with two switches – one for the propulsion control system and one for the control system. Also, special attention was put to wiring and alignment of the electronic components inside the frame, as during the tests of the first prototype, the wires got stuck between the track links.

The coral element of the circuit is the DFR0205 power step-down converter module that is used as a power distribution module [50]. The module is able to convert DC voltage in the range of 3,6 - 25V to a selectable level in the range of 3,3 - 25V. The module has two options for the output voltage level controlled by the onboard switch. The first one is 5 V and the second is adjustable. The board has three output interfaces that make the connection of the wires simple: 5.08 mm terminal interface, JST2.54 interface, and two 2P pin headers. 5.08 mm terminal interface is used to power servos, whereas the pin header interfaces are used to power encoders. The fourth interface can be used to get the original voltage output, so the Arduino Uno was powered with it, as it requires a higher voltage level in the range of 6 - 20 V. The module is able to deliver up to 5A of constant output current. Another advantage of the board compared to others is its small dimensions - 46x50x20mm, which means that module will fit inside the robot frame. The board is provided with the ON/OFF button, but it controls only 3 of 4 outputs, so it leaves one of the outputs always powered.

The driver is powered directly from the Lithium Polymer (Li-Po) battery using the pluggable connectors. There were two options to connect the driver with the motors. The first was to solder them directly to the power cable, and another one was to use pin-headers. The second option was selected as it made the system modular and simplified the potential modifications. The motor driver has in total six pins to control the two motors. ENA and ENB are enable pins that are used to send the PWM signal from the microcontroller board to the driver. OUT1 and OUT2 pins relate to the first motor, whereas OUT3 and OUT4 relate to the second. Those pins define the direction of motor rotation. Important to mention that two pins of the same motor can never be in a high state, as it leads to short circuits and destruction of components. The driver is also provided with the changeable 12 A fuse placed inside the slot. Arduino 5 V pin was used to power the HC-06 Bluetooth module, whereas the 3,3 V pins were used to provide voltage reference for the motor driver. The breadboard connection diagram is shown in Figure 4.3.

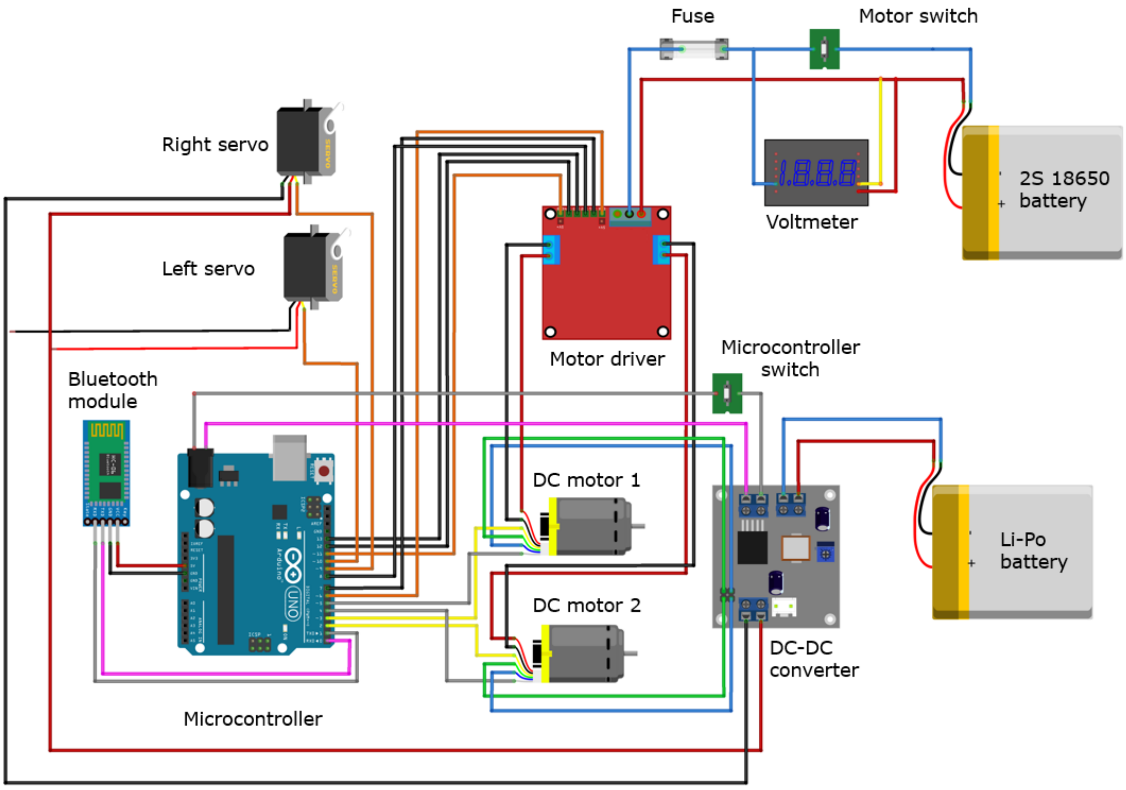

Figure 4.3 Breadboard connection diagram

5 CONTROL SYSTEM AND ENVIRONMENT PERCEPTION

5.1 Hardware architecture of the robot

The main goal of the thesis is to develop a highly modular platform that consists of two flexible track modules and the main body. Based on the literature review, similar platforms usually consist of multiple computers with particular tasks path planning, image processing, motion planning, etc. The main reason for that is the limited computation power of small-sized onboard computers. The same approach was selected for the development of the RCTR. Based on the previously gained experience, the development work was done considering the following stages of the robot development and potential changes that may appear due to this. For that reason, certain assumptions were made. It was decided to split the robot control units of the robot into 3:

- General main and image processing unit;
- Motion planning unit;
- Perceptions unit.

The most significant factor affecting the selection of the components was the mechanical design limitations. The selected components must fit inside the track frame elements. The maximum dimensions of the. In case the dimensions of the integrated components considering the connection interfaces exceed those values, the length of the frame plate elements must be modified. It means that the robot will lose its flexibility, and the concept of the robot will be inapplicable. For example, one of the most widely used single-board computers used for image processing and SLAM is the Jetson Nano, which has a length of 80 mm, a width of 100 mm, and a height of 30 mm. Those dimensions do not consider the wiring and sizes of the peripheral devise connectors. In case the USB 3.0 connector is attached to the board, the length increases to about 105 mm, what automatically changes the length of the plate to 125 mm, which is not acceptable.

Considering the above-mentioned facts, it was decided to focus this work on the optimization of the motion control unit. The main control and image processing unit and perception units should be selected in phases of the robot development as they will cover the design of the main body of the robot. This work also considers the selection of the sensors that can be implemented on the flexible track part.

5.2 Motion control unit

Each flexible track must be controlled with a separate motion control unit. The units must provide control for two DC motors and two servos. Control of the DC motors must be done using the feedback from the encoders.

The motion control unit must have enough pins for the connection. The board must have four PWM pins in total for control of two servos and two pins for the DC motors. Also, four digital pins must be used to control the direction of rotation of the motors, and the board should have at least two interrupt pins to read shaft position using the encoder. As the direction of the rotation is defined by the code, there is no particular need to use both encoder outputs to read the position values. Accordingly, to the motor driver datasheet, it must be provided with a 3,3 or 5 V voltage reference for the control. As the platform has limited external space, the motion control limit should fit inside the frame. An additional requirement was the ability of the HC-06 Bluetooth module to provide remote control during the tests.

It was decided to go further with the Arduino Uno board. Compared to the other boards, it has smaller dimensions, all the required interfaces, and has a lower price. The board has in total of 14 digital pins, which include 6 PWM pins and two interrupt pins. It is essential to mention that pin number 2 is both interrupt and PWM pin. Both 5 V and 3,3 V power pins are presented on the board, so the Bluetooth module and driver can be powered.

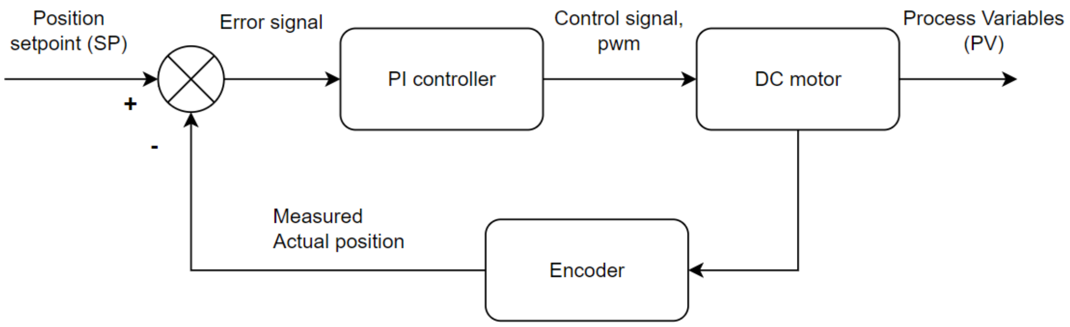

Figure 5.1 DC motor shaft position closed loop control system

To control the motors, it was decided to implement PI control due to its simplicity, the parameters were selected empirically. The description of control system is presented in Figure 5.1. However, the code includes all the parameters required for the PID control. The control signal of the control is the PWM value that is applied to the motors. Important to mention that the maximal and the minimal control signal values are limited based on the efficiency map of the Polulu 4847 motor. It was made to make the motor work in the most efficient speed range. The maximal applied PWM value is 255, and the minimal is 180 PWM.

During one full rotation of the motor shaft, the encoder of both A and B generates 48 rising and falling edges., while reading only a single edge of one channel results in 12 counts per revolution of the shaft [45]. The motor is provided with two gearboxes, the gearbox of the motor with a ratio of 98,7779:1 and the external gearbox with a ratio of 5:1. It means that using only one channel will generate 1185 counts per revolution for the motor gearbox shaft and 5927 counts per revolution for the main shaft of the robot. Each link of the robot corresponds to 36° of the main sprocket wheel diameter. It means that moving the one link will result 593 encoder counts and the moving of the sprocket to 1° results 16,5 counts. So, the designed system can deliver exact information regarding the position of the sprocket wheel.

To test the performance of the control unit, it was implemented on the flexible track. As the input was used, the code was implemented on the first prototype, but it was modified to meet the new needs. While developing the code, the special attention was to make it modular in applicable for future needs. Similar to the first prototype, it was decided to use the HC-06 Bluetooth module to control the robot using the smartphone application. The application was updated using MIT App Inventor visual programming environment. The code is created using blocks, so the functionalities of the app can be quickly adopted for new needs. The environment has the tools to visualize received data, what is crucial for debugging the code and optimization of the system. Several changes were introduced to the user interface design to make it more user-friendly. Developed user interface is shown in Figure 5.2.

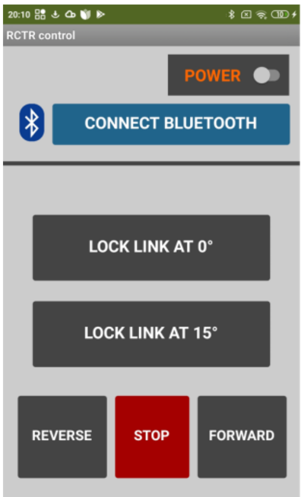

Figure 5.2 Updated user interface of the smartphone control application

The idea behind the code controlling the motion unit is simple. The code is constantly running inside the infinite loop, reading the values obtained from the Bluetooth module using serial communication at a baudrate of 9600. The commands are generated by the smartphone application. Each command corresponds to the particular action performed by the controller unit. While creating the code, the limitations and specific features of the hardware were considered. The speed of the motor rotation is limited accordingly to the limit values. Also, the direction of the motor shaft rotation cannot be changed instantly to prevent the motor from overheating and undesirable consequences. The set of preprogrammed actions enables testing the critical functionalities of the control unit:

- Moving the robot and forward and reverse directions;
- Locking links under the specific angle;
- Stopping the robot.

The simplified algorithm for the developed code is presented in Figure 5.3, the code with the comments can be found in Appendices.

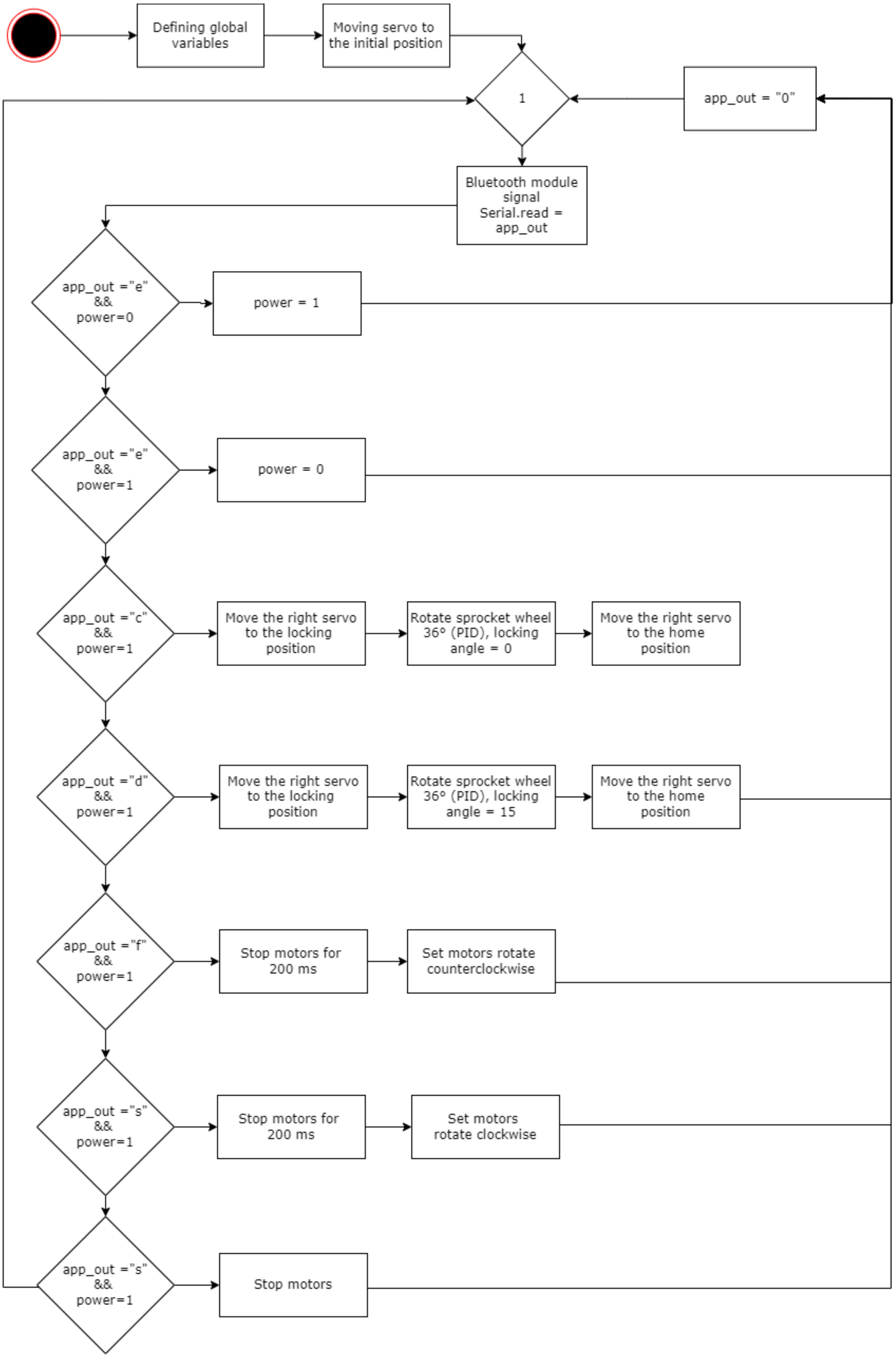
Defining global variables
Moving servo to the initial position
1
app_out = "0"
Bluetooth module signal Serial.read = app_out
app_out ="e" && power=0
power = 1
app_out ="e" && power=1
power = 0
app_out ="c" && power=1
Move the right servo to the locking position
Rotate sprocket wheel 36° (PID), locking angle = 0
Move the right servo to the home position
app_out ="d" && power=1
Move the right servo to the locking position
Rotate sprocket wheel 36° (PID), locking angle = 15
Move the right servo to the home position
app_out ="f" && power=1
Stop motors for 200 ms
Set motors rotate counterclockwise
app_out ="s" && power=1
Stop motors for 200 ms
Set motors rotate clockwise
app_out ="s" && power=1
Stop motors

Figure 5.3 Main control algorithm

5.3 Environment perception

5.3.1 Requirements

Based on the literature review, such types of robots are usually provided with the flowing types of exteroceptive sensors: depth camera, VIS camera, and LiDAR. As this work covers the first and the second stages of the robot development, it was decided to focus mainly on those types of sensors that can be implemented on the single flexible track module. The output of this chapter is assumed to be used in the next stages of robot development.

The RCTR needs to have the capability to detect and overcome the static obstacle in front of the robot autonomously by changing the configuration on track links. It is possible to define two types of static obstacles, which are concave and convex. To concave obstacles, potholes, ditches, and clefts can be related. Convex obstacles can be represented by stairs, fallen trees, the rubble of collapsed buildings, rocks, and many other urban landscape forms. So, both types of obstacles should be detected.

The aims of this sub-chapter are:

- to select the sensors that can be implemented on the flexible track module of the robot; create the preliminary mechanical design for mounting of those sensors;
- propose the preliminary algorithm for their implementation of the robot.

5.3.2 Selection of sensors

TF-Luna is a single-point ranging LiDAR that utilizes the time of flight (ToF) principle [52]. The module has small dimensions of 21,25x35x19 mm and has a detection range up to 8 m. The distance resolution of the sensor is 10 mm, and the blind zone of 200 mm. The FOV of the sensor is 2°. At the distance of one meter, the module is able to detect the object with a minimum edge length of 35 mm in the case of the distance of 8 meters, the edge length is 280 mm. The module requires a 5 V power supply voltage. The peak current consumption of the sensor is 150 mA, whereas the average value is about 70 mA.

Figure 5.4 TF-Luna Micro single-point ranging LiDAR [52]

Similar to the robots described in the literature, it was decided to use the Intel RealSense D435i depth camera that can be used in both indoor and outdoor conditions. One of the key features is an inbuilt IMU for tracking robots in 6 FOFs. The module is widely used for SLAM in robotics and has detailed documentation. It is provided with both active infrared stereo and RGB modules. The framerate of the depth module is 90 fps with a resolution up to 1280 x 720 FOV for the depth module is 87° × 58°. The resolution of the RGB sensor is 1920 x 1080 with a framerate of 30 fps. The minimum depth distance of the D435i is about 28 cm. The point clouds data, as well as RGB image output, can be processed using machine vision and machine learning techniques. Compared to TF-Luna D435i has a higher current consumption of 700 mA and gets power via a USB type A connector.

Figure 5.5 Intel RealSense D435i depth camera module [53]

For the pothole edge detection was selected the HC-SR04 ultrasonic sensor. It has the detection range of 20 – 4000 mm and a measuring angle of 15°, and a resolution of 3 mm. The small blind zone means that the module can easily detect a concave obstacle. The module is relatively small, 45x20x15mm, what means that it can be easily mounted in the front of the robot.

Figure 5.6 HC-SR04 ultrasonic distance sensor module [54]

5.3.3 Design of sensor mountings

The design of the mountings has to consider the technical parameters of the selected sensors. The sensor mountings must protect the sensors from external damages and provide the ability to adjust the configuration in case that is needed. The sensors must be placed in the front part of the robot.

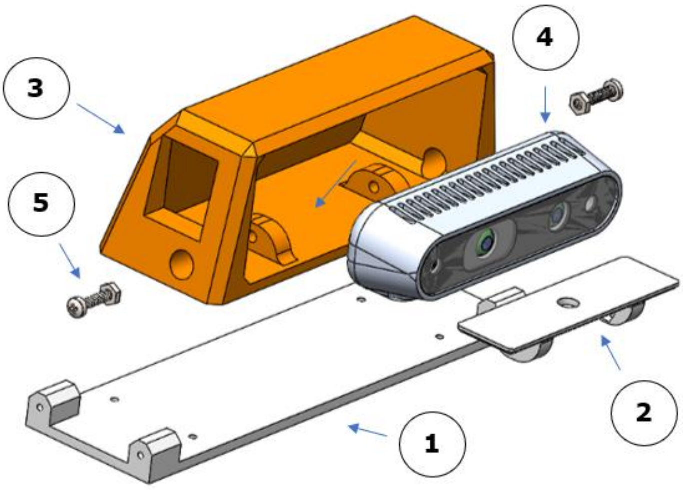

Figure 5.7 Design of the depth camera mounting:
1 – upper plate; 2 – camera plate; 3 – camera housing; 4 – D435i camera; 5 – mounting screws.

It was decided to place the RealSense camera on top of the front part of the robot. The camera is moved slightly backward for several reasons. The first reason is the camera protection in case of robot falls, and another reason is the minimization of the blind zone of the camera. There are no robot parts in the field of camera view that can interfere in the data collection. The sensor mounting consists of 2 parts which are the main body on the mounting plate. The parts are connected together using M3 bolts, the angle of inclination can be adjusted in case that is needed. For that purpose, special service holes were added to the body. Figure 5.8 illustrates the blind zone of the camera based on the vertical field of view, and it can be seen that the potholes can be detected with the depth camera at a distance of 305 mm, which is very close to the value of the minimum depth distance. The design of the mounting is presented in Figure 5.7.

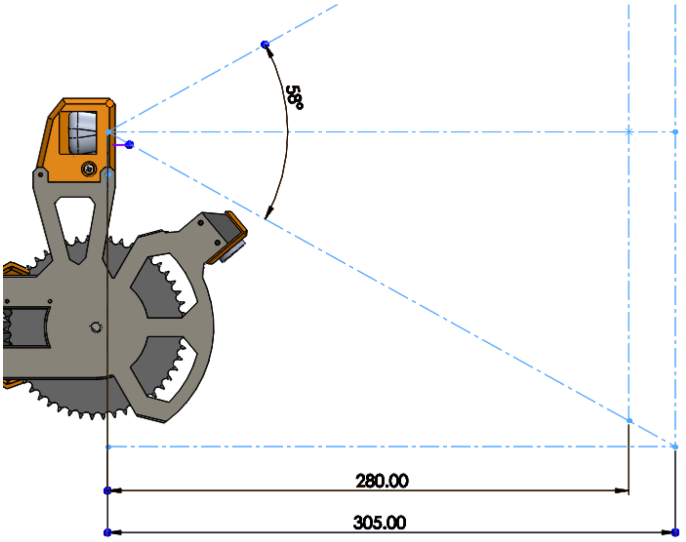

Figure 5.8 FOV and depth zone of the D435i camera on RCTR

For the TF-Luna sensor, the mounting that enables profile scanning was designed. For that purpose, the typical approach for hobby robotics was used. The sensor is attached to the plate, which is rotated by the servo motor. For that purpose, a 180° servo can be used, enabling profile scanning both up and down. To protect the module from external impact, it is protected with a cover. The designed module can be used for both measuring the distance to the object in front of the object and defining its height. The height of the object can be calculated using three parameters. The first is the initial distance to the object detected in front of the robot. The next parameters are the angle that corresponds to the maximum height of the obstacle and the distance to the highest point of the obstacle. Those values can be derived from the sensor output values by calculating the deviation and servo position. The design of the TF-Luna sensor mounting is shown in Figure 5.9, whereas the algorithm is described in Figure 5.10.

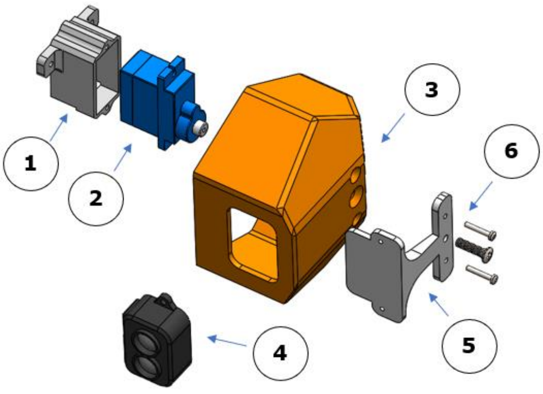

Figure 5.9 Design of the TF-Luna sensor mounting:
1 – servo housing; 2 – servo; 3 – module cover; 4 – TF-Luna sensor; 5 – sensor mounting; 6 – fastening screws.

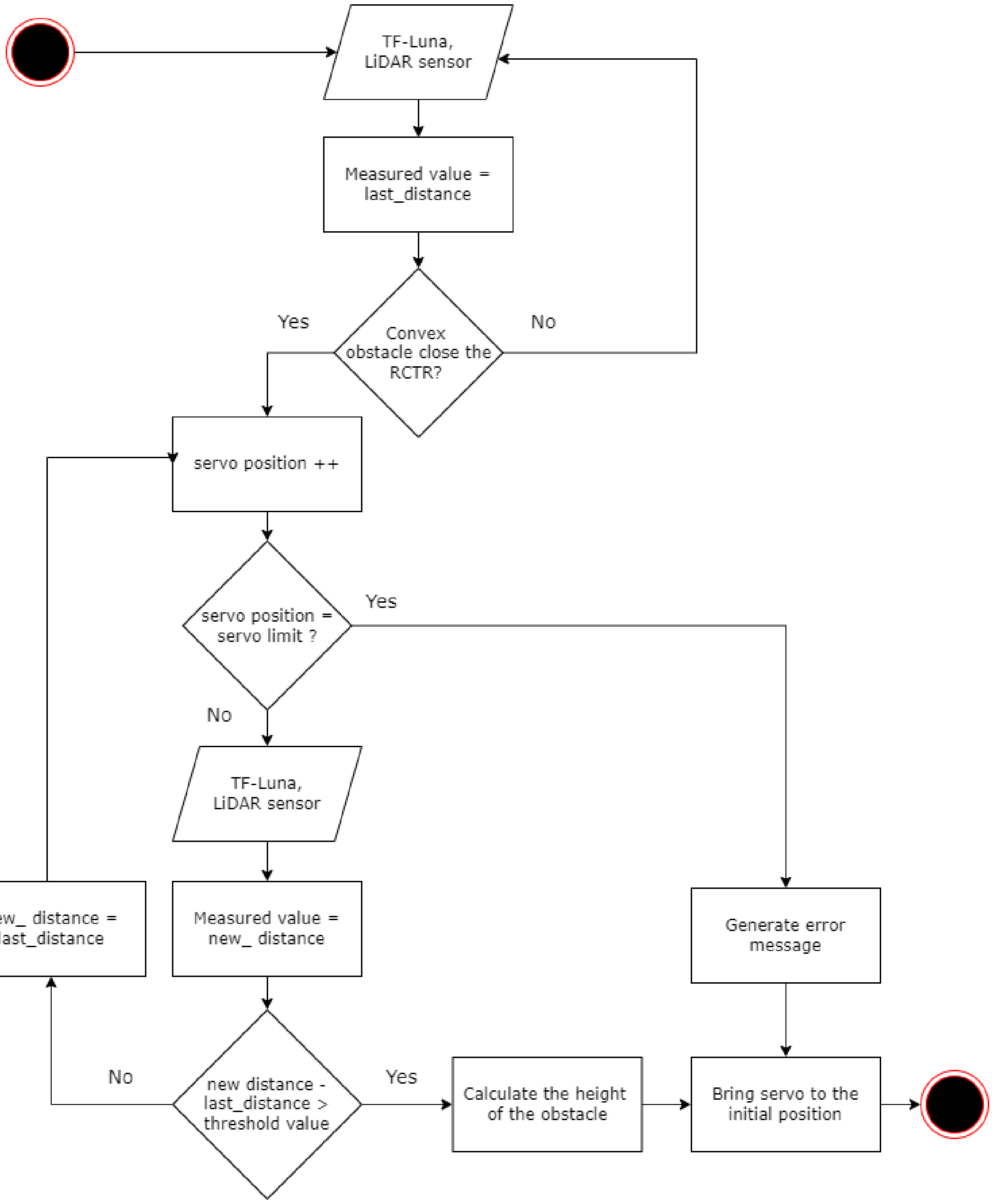
TF-Luna, LiDAR sensor
Measured value = last_distance
Convex obstacle close the RCTR?
Yes
No
servo position ++
servo position = servo limit ?
Yes
No
TF-Luna, LiDAR sensor
new_ distance = last_distance
Measured value = new_ distance
Generate error message
No
new distance - last_distance > threshold value
Yes
Calculate the height of the obstacle
Bring servo to the initial position

Figure 5.10 Preliminary algorithm for TF-Luna sensor module

The design of ultrasonic sensor mounting is shown in Figure 5.11. The sensor is placed on the plate connecting the front chain guiding plates. The position of the mounting considers the vertical FOV of the Intel RealSense camera, so the ultrasonic sensor does not interfere with the depth information reading. The module is inclined to capture the distance to the ground closer to the robot.

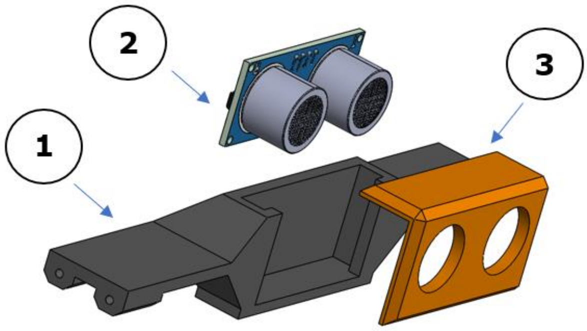

Figure 5.11 Ultrasonic sensor mounting:
1 – front plate mounting; 2 – HC-SR04 ultrasonic sensor; 3 – module cover.

5.3.4 Environment perception algorithm

In order to tackle obstacles, a preliminary algorithm was developed. The robot must be able to detect both concave and convex obstacles, measure key parameters, and estimate the traversability of obstacles. The information obtained from the sensor should create a traversability map that considers the specific features of the track robot. There are two main possible obstacle types: concave obstacle and convex obstacle. The obstacles must be measured using different sensor types to ensure the results are correct.

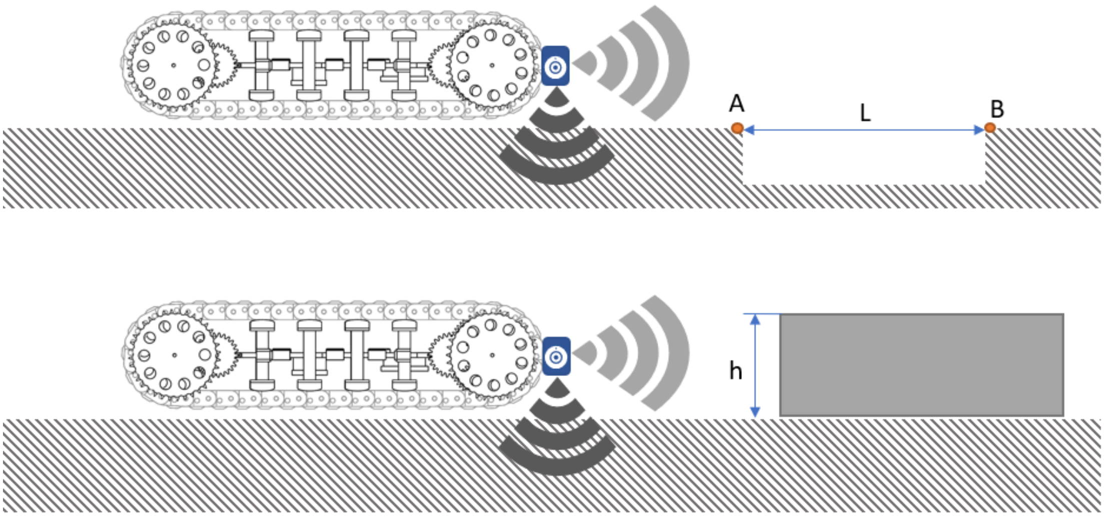

Figure 5.12 Types of obstacles and their parameters:
A – starting edge of the negative/concave obstacle, *B* – ending edge of the negative/concave obstacle, *L* – length on the negative obstacle, *h* – height of the positive obstacle.

It is possible to define two sets of sensors accordingly to their primary task. The task of the down-looking sensors is to detect the edge of the concave obstacles to prevent the robot from falling inside them. The task of the forward-looking sensors of the robot is to continuously monitor the area in front of the robot. The dimensions of the obstacles can be calculated using the point cloud generated by LiDAR and depth camera and by extracting the required parameters from the RGB image. The precise position of the edge can be detected using sonar.

The preliminary robot control algorithm is introduced in Figure 5.13. The idea behind it is quite simple: after initializing the components, the robot calculates the path and starts moving. In the case of the concave objects, the robot first detects the edge of the pothole and then changes the position to take the measurement. The parameters of the platform define the maximal traversable length for the pothole. After comparing the maximum traversable pothole length to measured values, the algorithm must decide if the robot can overcome the obstacle. If the pothole is too large, the robot must search for a new path. Otherwise, it overcomes the obstacle and continues navigation.

In the case of convex surfaces, the algorithm measures the height of the object and decides if it is possible to climb on it. If the object height is within the range of maximal traversable height, the algorithm generates the track configuration and executes overcoming. A new path must be generated if it is impossible to overcome the obstacle.

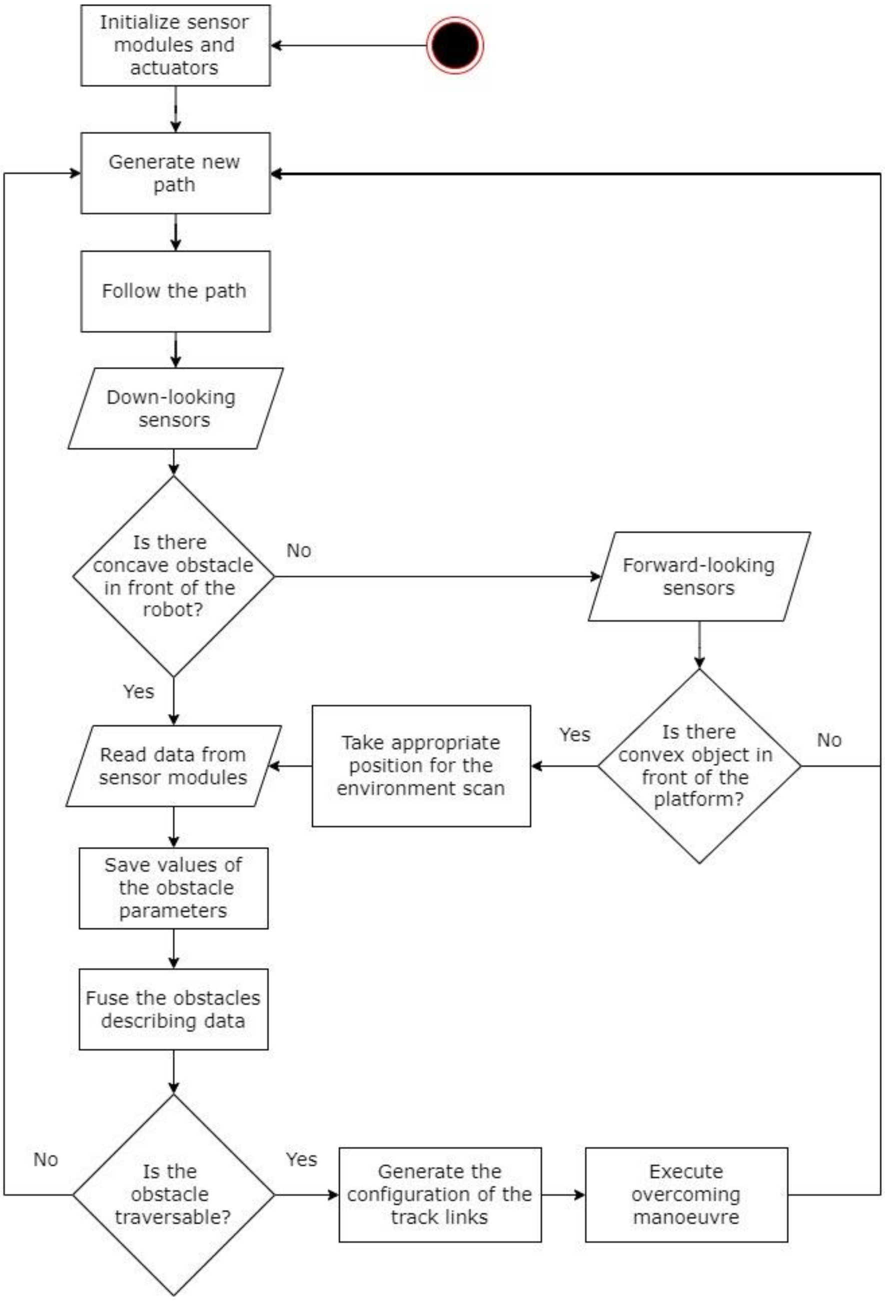
Initialize sensor modules and actuators
Generate new path
Follow the path
Down-looking sensors
Is there concave obstacle in front of the robot?
No
Forward-looking sensors
Yes
Is there convex object in front of the platform?
Yes
No
Take appropriate position for the environment scan
Read data from sensor modules
Save values of the obstacle parameters
Fuse the obstacles describing data
No
Is the obstacle traversable?
Yes
Generate the configuration of the track links
Execute overcoming manoeuvre

Figure 5.13 Preliminary robot main algorithm

6 BUILDING AND TESTING THE PROTOTYPE

6.1 3D printing of the components and assembling

Most of the parts used in the track robot design were created using plastic 3D printing. The main advantage of 3D printing is the ability to produce parts with complex shapes in a short time. Most of the parts were produced using PLA plastic, as it has a low price and good mechanical properties. For printing the flexible elements of the frame, TPE plastic was used. Compared to the first prototype, different approaches were used to optimize the 3D printing and increase the output quality.

First of all, different infill rates were used for the different parts. The lower infill rates were used for non-critical parts of the robot. For example, the front, rear, and chain guiding plates of the robot were printed at the infill rate of 50%. It helped to decrease the printing time of the part and the material consumption. Compared to the generative design, the main disadvantage of a such weight reduction approach is not considering the nature of the loads affecting the parts. However, it is not critical in the case of passive elements. The infill of 100% was implemented mainly on the parts related to the transmission.

Also, the different printing resolution was implemented for different parts mainly due to tolerance requirements of the details. The resolution of 0,1 mm was primarily applied to transmission parts like gears and sprocket wheels. It took a while to figure out the best settings for printing gears for the 3 mm D-shaft before the connection became solid. Increasing the printing quality led to a longer printing time, in some cases, this difference was up to 80% of the initial time.

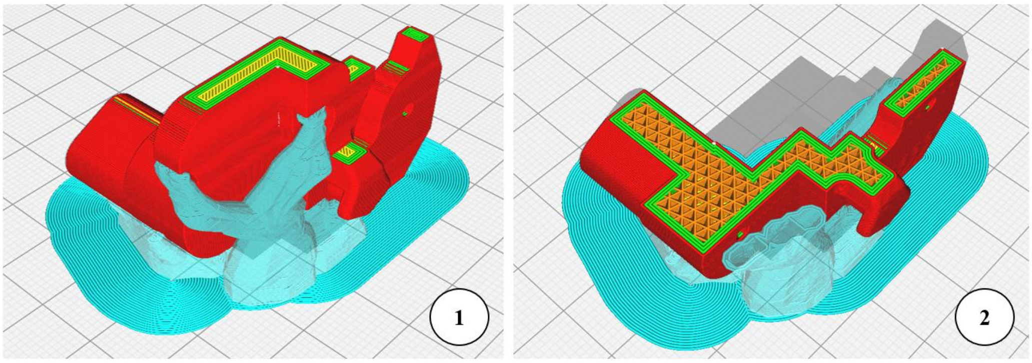

Figure 6.1 Tree support structure used for link plate:
1 – external view of the detail, 2 – internal view of the detail and support structure

Based on the experience gained while creating the first prototype, the track links were printed vertically. In the case of horizontal annotation of the part, the connection of the layers became the weak point of the detail. For example, in the case of the side plates of the track links, vertical annotation guaranteed better performance of the parts. Also, the track links had small elements that had to be printed very precisely. The tree type of the support structure presented in Figure 6.1 was used for that purpose.

The prints were printer using Anycubic Vyper and Creality Ender-3 S1 printers. The total printing time of the robot was about 300 hours without considering that prints had defects and some parts were modified and re-printed on the go. Before assembling, the details required some processing, which included removing the supports, drilling the holes, and grinding the surfaces in case of defects. Nearly half of the printing time was spent on printing the track elements.

For cable management simplification, the cables were joined using the cable strips. The fame parts were connected using the 12 mm 2.2M screws. The same approach was used for some of the electronics components. During the assembling of the track links, the mounting holes were drilled. However, during the tests, it became evident that such an assembling technique was not reasonable, as in some cases, the shaft was not well fixed inside the link. There was an option to use super or hot glues, but it significantly reduced the modularity. The best way to assemble the track links was to use the rod as the drill bit and insert the rotating link shaft inside the unmachined mounting holes. The rotation of the shaft slightly melts the plastic creating a glue-like stable connection. The essential advantage of such an approach is the ability to disassemble the parts if needed.

6.2 Tests of the platform

After the assembling of the robot was done, the new prototype was tested. Overall it is possible to conclude that the updated design of the robot showed good performance. The speed of the robot has significantly reduced, which simplified the tests as it helped to observe better the behavior of the robot. During the first tests, several design flaws were found. In the 3D model, the height of the heads of the screws was not considered. For that reason, the position of certain transmission parts had to be modified. Other parts that had to be changed were the guiding plate plates. In the updated design, plates were provided with reinforcing ribs to make them more stable. However, it turned out during the tests that due to deformation of the details, the ribs rested against the side plates of the track blocking the whole chain. It also led to the damage to the front plate as the motors were trying to spin the gears. The result is shown in Figure 6.2. The design of the gearbox mounting was modified as well as 3D printing settings.

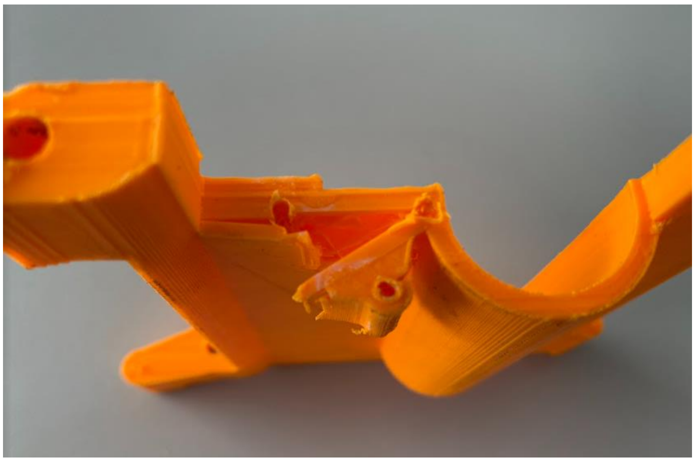

Figure 6.2 Front plate damaged during the tests

Also, the gap between the link spikes and the guiding elements of the plates appeared to be minor, so it has to be modified. It was hard to predict which gap would guarantee the smooth moving of the chain and not cause the excessively loose tensioning. After the changes in the design of guiding plates were introduced, the system started working as it was expected to.

The control system overall has shown good performance. Certain parts of the motion control unit code were modified and tuned during the tests. Improper connection of the wires caused communication issues, so the robot could not be controlled via the Bluetooth module.

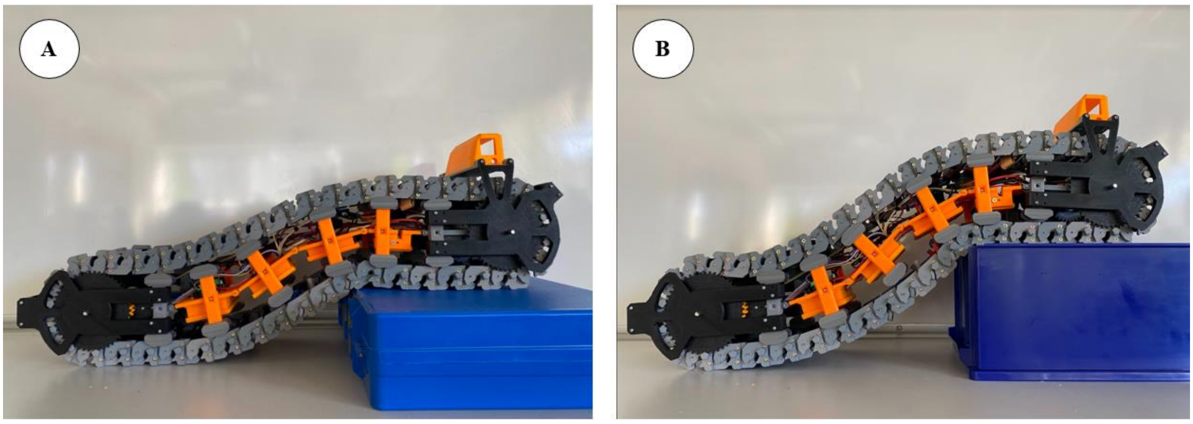

Figure 6.3 Tests of the developed prototype:
A – robot climbing the 90 mm obstacle; B – robot climbing the 150 mm obstacle.

A significant advantage of the platform is the balanced flexibility of the frame, although the length of each plate was extended, and there were components placed inside each of them. The tensioning mechanism has shown promising performance and made tuning of the platform very simple. Figure 6.3 shows that the robot has the potential to climb over obstacles of different heights.

7 FUTURE PLANS

Considering the general plan of the track robot development, the main focus in the future must be set on creating the platform consisting of two separate tracks. Significant attention can be put on investigating the behavior of such mobile robots in rugged terrain. These tests can help evaluate the overall potential of the platform and its reliability.

During the design of the robot, several aspects were identified that could be addressed in the next phases of the platform development. The active tensioning system can be introduced for the control of the track. During different working modes, the chain of the robot may require different tensioning. Implementation of the active tensioning system will increase the overall reliability of the robot.

During the subsequent phases of the robot development, significant attention must be put into creating the framework for sensor data fusion. In this topic can be considered the generation of the traversability map using only limited onboard computational power. The outputs of such research may become helpful for the solutions used in areas with limited communication options.

Another topic that may be considered in the future is the track link design changes that will help to improve traction. Design development may include research of the materials used for the surfaces of the link which contacts the ground.

8 CONCLUSION

During the last years the field of the mobile robotics was quickly developing. Several approaches by different researches were introduces for moving in difficult terrain environment. Based on the concept the robots can be divided into legged, tracked, reconfigurable, wheeled and hybrid. Those platforms are used for different task including inspection, service and delivery.

One of the recently developed concepts is the reconfigurable continuous track robot that combines the advantages of different approaches. In 2020 the RCTR robot was created by students of Tallinn University of Technology. The platform showed good performance, however, come parts of the robot required further development. Based on the test results and literature review, there was created the strategy for the further development of the robot that consists of 4 stages. In work covered is covered the first and partly the second stages. During the design of all components, the special attention was set on increasing the overall modularity of the system. The table 8.1 represents the final parameters of the developed platform.

Table 8.1 Parameters of the second prototype

Parameter	**Achieved value**
Length, mm	700
Width, mm	165
Height, mm	200
Top speed of the robot, m/s	0,11
Mass, kg	3,45
Acceleration, m/s^2	0,2

The new mechanical design solves the main drawbacks of the first prototype. The structural elements of the frame were modified in order to meet the new requirements defined by set of electrical components. The custom gearbox was developed for the new transmission. Also, the design of the track links was modified. The generative design techniques were tested as the tool to optimize the weight of the robot. The results of those test showed that the implementation of the generative design helps to significantly reduce the weight of detail as well as decrease the printing time of the parts.

The changes were also introduced to the thrust system and control systems of the robot. There were selected the new motors provided with inbuilt encoders. The encoders were used for the development of the robot closed loop control system. The developed motion control unit is the module that can be easily implemented in bigger system. To control the performance of the robot, the smartphone application was updated using the MIT App inventor.

Considering the direct relation between the mechanical design of the flexible track and the perception system of the robot, the preliminary selection of the sensors was done. The work describes the concepts used for the selection of the sensors and design of the mountings that considers specifications of sensors, application and their task within the algorithm.

9 KOKKUVÕTE

Viimastel aastatel on mobiilse robootika valdkond kiiresti arenenud. Keerulisemaastiku tingimustes liikumiseks on uuringustes tutvustatud mitmeid erinevaid lähenemisviise. Kontseptsiooni järgi saab robotid jagada jalgadega, roomik-, ümberkonfigureeritavateks, ratastega ja hübriidrobotiteks. Neid platvorme kasutatakse erinevate ülesannete jaoks, sealhulgas inspekteerimine, teenindus ja pakivedu.

Üks hiljuti välja töötatud kontseptsioone on ümberkonfigureeritav roomikrobot, mis ühendab erinevate lähenemisviiside eelised. Sarnane lahendus on loodud 2020. aastal ka Tallinna Tehnikaülikooli tudengite poolt. Platvorm näitas head jõudlust, kuid roboti osad nõudsid edasiarendamist. Katsetulemuste ja kirjanduse ülevaate põhjal koostati roboti edasiarendamise strateegia, mis koosneb neljast etapist. Antud töös käsitletakse esimest ja osaliselt teist etappi. Kõikide komponentide projekteerimisel pöörati erilist tähelepanu süsteemi üldise modulaarsuse suurendamisele. Tabelis 9.1 on esitatud arendatud platvormi lõplikud parameetrid.

Table 9.1 Teise prototüübi parameetrid

Parameeter	Saavutatud väärtus
Pikkus, mm	700
Laius, mm	165
Kõrgus, mm	200
Roboti maksimaalne kiirus, m/s	0,11
Mass, kg	3,45
Kiirendus, m/s^2	0,2

Uus mehaaniline konstruktsioon lahendab esimese prototüübi peamised puudused. Raami konstruktsioonielemente muudeti, et need vastaksid uutele elektrilistele komponentidele. Uue ülekande süsteemi jaoks töötati välja kohandatud käigukast . Samuti muudeti roomiku linkide disaini. Generatiivseid disainitehnikaid testiti roboti kaalu optimeerimiseks. Nende katsete tulemused näitasid, et generatiivse disaini rakendamine aitab oluliselt vähendada detailide kaalu ja lühendada detailide trükkimise aega.

Muudatused viidi sisse ka roboti propulsiooni- ja juhtimissüsteemides. Valiti uued mootorid, mis on varustatud sisseehitatud enkoodritega. Enkoodreid kasutati roboti suletud ahela juhtimissüsteemi väljatöötamiseks. Väljatöötatud liikumisjuhtimisseade on moodul, mida saab rakendada suurema süsteemi osana. Roboti jõudluse juhtimiseks uuendati nutitelefoni rakendust MIT App Inventor keskkonna abil.

Arvestades otsest seost painduva raja mehaanilise konstruktsiooni ja roboti tajusüsteemi vahel, tehti andurite eelvalik. Töös kirjeldatakse andurite valikul kasutatavaid kontseptsioone ja kinnituste kujundust, võttes arvesse andurite spetsifikatsioone, rakendust ja nende ülesannet algoritmis.

LIST OF REFERENCES

[1] M. B. Alatise and G. P. Hancke, "A Review on Challenges of Autonomous Mobile Robot and Sensor Fusion Methods," *IEEE Access*, vol. 8, pp. 39830–39846, 2020, doi: 10.1109/ACCESS.2020.2975643.

[2] C. Gehring *et al.*, "ANYmal in the Field: Solving Industrial Inspection of an Offshore HVDC Platform with a Quadrupedal Robot," *Springer Proceedings in Advanced Robotics*, vol. 16, pp. 247–260, 2021, doi: 10.1007/978-981-15-9460-1_18.

[3] R. Losch, S. Grehl, M. Donner, C. Buhl, and B. Jung, "Design of an Autonomous Robot for Mapping, Navigation, and Manipulation in Underground Mines," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1407–1412, Dec. 2018, doi: 10.1109/IROS.2018.8594190.

[4] M. di Castro, M. Ferre, and A. Masi, "CERNTAURO: A modular architecture for robotic inspection and telemanipulation in harsh and Semi-Structured environments," *IEEE Access*, vol. 6, pp. 37506–37522, Jun. 2018, doi: 10.1109/ACCESS.2018.2849572.

[5] W. Reid *et al.*, "Actively Articulated Wheel-on-Limb Mobility for Traversing Europa Analogue Terrain," *Springer Proceedings in Advanced Robotics*, vol. 16, pp. 337–351, 2021, doi: 10.1007/978-981-15-9460-1_24.

[6] M. Gianni *et al.*, "MIOM:A MIxed-Initiative Operational Model in Urban Search and Rescue," 2016. [Online]. Available: https://www.researchgate.net/publication/303313600

[7] J. Zhao, J. Gao, F. Zhao, and Y. Liu, "A search-and-rescue robot system for remotely sensing the underground coal mine environment," *Sensors (Switzerland)*, Oct. 2017, doi: 10.3390/S17102426.

[8] T. Kislassi and D. Zarrouk, "A Minimally Actuated Reconfigurable Continuous Track Robot," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 652–659, Apr. 2020, doi: 10.1109/LRA.2019.2959237.

[9] Arduino, "Arduino Uno Rev3," 2021. https://store.arduino.cc/products/arduino-uno-rev3/ (accessed Dec. 05, 2021).

[10] Diana Belolipetskaja, "Design of mechanical and electrical part for body shape changing track robot," Bachelor thesis, Tallinn University of Technology, Tallinn, 2020.

[11] Daniil Valme, "Design of mechanical and control system for flexible track robot," Bachelor thesis, Tallinn University of Technology, Tallinn , 2020.

[12] T. Miki, P. Khrapchenkov, and K. Hori, "UAV/UGV autonomous cooperation: UAV assists UGV to climb a cliff by attaching a tether," *Proceedings - IEEE*

International Conference on Robotics and Automation, vol. 2019-May, pp. 8041–8047, May 2019, doi: 10.1109/ICRA.2019.8794265.

[13] A. Al-Shanoon, H. A. Tan, H. Lang, and Y. Wang, "Mobile robot regulation with position based visual servoing," *CIVEMSA 2018 - 2018 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications, Proceedings*, Aug. 2018, doi: 10.1109/CIVEMSA.2018.8439978.

[14] T. Rouček *et al.*, "DARPA Subterranean Challenge: Multi-robotic Exploration of Underground Environments," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11995 LNCS, pp. 274–290, 2020, doi: 10.1007/978-3-030-43890-6_22.

[15] E. Guizzo, "By leaps and bounds: An exclusive look at how Boston dynamics is redefining robot agility," *IEEE Spectrum*, vol. 56, no. 12, pp. 34–39, Dec. 2019, doi: 10.1109/MSPEC.2019.8913831.

[16] E. Tennakoon, T. Peynot, J. Roberts, and N. Kottege, "Probe-before-step walking strategy for multi-legged robots on terrain with risk of collapse," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 5530–5536, May 2020, doi: 10.1109/ICRA40945.2020.9197154.

[17] E. Kelasidi, M. Jesmani, K. Y. Pettersen, and J. T. Gravdahl, "Locomotion efficiency optimization of biologically inspired snake robots," *Applied Sciences (Switzerland)*, vol. 8, no. 1, Jan. 2018, doi: 10.3390/APP8010080.

[18] T. Hoffmann and G. Prause, "On the regulatory framework for last-mile delivery robots," *Machines*, vol. 6, no. 3, Sep. 2018, doi: 10.3390/machines6030033.

[19] M. Gianni, F. Ferri, M. Menna, and F. Pirri, "Adaptive Robust Three-dimensional Trajectory Tracking for Actively Articulated Tracked Vehicles*," *Journal of Field Robotics*, vol. 33, no. 7, pp. 901–930, Oct. 2016, doi: 10.1002/ROB.21584.

[20] R. Edlinger and A. Nuechter, "MARC-Modular Autonomous Adaptable Robot Concept," *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2019*, pp. 43–49, Sep. 2019, doi: 10.1109/SSRR.2019.8848934.

[21] M. Arai, Y. Tanaka, S. Hirose, H. Kuwahara, and S. Tsukui, "Development of 'Souryu-IV' and 'Souryu-V:' Serially connected crawler vehicles for in-rubble searching operations," *Journal of Field Robotics*, vol. 25, no. 1–2, pp. 31–65, Jan. 2008, doi: 10.1002/rob.20229.

[22] "DARPA Subterranean (SubT) Challenge." https://www.darpa.mil/program/darpa-subterranean-challenge (accessed Nov. 28, 2021).

[23] C. Hu, L. Zhao, L. Cao, P. Tjan, and N. Wang, "Steering control based on model predictive control for obstacle avoidance of unmanned ground vehicle," *Measurement and Control (United Kingdom)*, vol. 53, no. 3–4, pp. 501–518, Mar. 2020, doi: 10.1177/0020294019878871.

[24] D. C. Liyanage, R. Hudjakov, and M. Tamre, "Hyperspectral imaging methods improve RGB image semantic segmentation of unstructured terrains," *15th International Conference Mechatronic Systems and Materials, MSM 2020*, 2020, doi: 10.1109/MSM49833.2020.9201738.

[25] T. Kislassi and D. Zarrouk, "A Minimally Actuated Reconfigurable Continuous Track Robot," in *IEEE Robotics and Automation Letters*, Apr. 2019, vol. 5, no. 2. doi: 10.1109/LRA.2019.2959237.

[26] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A Systematic Review of Perception System and Simulators for Autonomous Vehicles Research," *Sensors 2019, Vol. 19, Page 648*, vol. 19, no. 3, p. 648, Feb. 2019, doi: 10.3390/S19030648.

[27] H. I. Christensen and G. D. Hager, "Sensing and Estimation," in *Springer Handbook of Robotics*, Springer, Cham, 2016, pp. 91–112. doi: 10.1007/978-3-319-32552-1_5.

[28] W. Yuan, J. Li, M. Bhatta, Y. Shi, P. S. Baenziger, and Y. Ge, "Wheat height estimation using LiDAR in comparison to ultrasonic sensor and UAS," *Sensors (Switzerland)*, vol. 18, no. 11, Nov. 2018, doi: 10.3390/s18113731.

[29] L. Kleeman and R. Kuc, "Sonar Sensing," in *Springer Handbook of Robotics*, Springer, Cham, 2016, pp. 753–782. doi: 10.1007/978-3-319-32552-1_30.

[30] Kyberd Stephen *et al.*, "The Hulk: Design and Development of a Weather-proof Vehicle for Long-term Autonomy in Outdoor Environments," 2019. Accessed: Nov. 28, 2021. [Online]. Available: https://www.researchgate.net/publication/338697975_The_Hulk_Design_and_Development_of_a_Weather-proof_Vehicle_for_Long-term_Autonomy_in_Outdoor_Environments

[31] F. Xu, L. Chen, J. Lou, and M. Ren, "A real-time road detection method based on reorganized lidar data," *PLoS ONE*, vol. 14, no. 4, Apr. 2019, doi: 10.1371/JOURNAL.PONE.0215159.

[32] A. Newman, G. Yang, B. Wang, D. Arnold, and J. Saniie, "Embedded Mobile ROS Platform for SLAM Application with RGB-D Cameras," *IEEE International Conference on Electro Information Technology*, vol. 2020-July, pp. 449–453, Jul. 2020, doi: 10.1109/EIT48999.2020.9208310.

[33] M. Hutter *et al.*, "ANYmal - toward legged robots for harsh environments," *Advanced Robotics*, vol. 31, no. 17, pp. 918–931, Sep. 2017, doi: 10.1080/01691864.2017.1378591.

[34] H. Durrant-Whyte and T. C. Henderson, "Multisensor Data Fusion," in *Springer Handbook of Robotics*, Springer, Cham, 2016, pp. 867–896. doi: 10.1007/978-3-319-32552-1_35.

[35] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2016-November, pp. 1184–1189, Nov. 2016, doi: 10.1109/IROS.2016.7759199.

[36] A. F. Said, "Robust and accurate objects measurement in real-world based on camera system," *Proceedings - Applied Imagery Pattern Recognition Workshop*, vol. 2017-October, Sep. 2018, doi: 10.1109/AIPR.2017.8457954.

[37] L. E. Kavraki and S. M. LaValle, "Motion Planning," in *Springer Handbook of Robotics*, Springer, Cham, 2016, pp. 139–162. doi: 10.1007/978-3-319-32552-1_7.

[38] J. Minguez, F. Lamiraux, and J. P. Laumond, "Motion planning and obstacle avoidance," in *Springer Handbook of Robotics*, Springer International Publishing, 2016, pp. 1177–1201. doi: 10.1007/978-3-319-32552-1_47.

[39] M. Hutter *et al.*, "ANYmal - toward legged robots for harsh environments," *Advanced Robotics*, vol. 31, no. 17, pp. 918–931, Sep. 2017, doi: 10.1080/01691864.2017.1378591.

[40] M. M. Varadharajan, "Optimization of Structures Shape and Weight Optimization of Structural Members," 2013.

[41] A. Kallaste, T. Vaimann, and A. Rassõlkin, "Additive Design Possibilities of Electrical Machines," in *59th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, 2018, pp. 1–5.

[42] T. Luedeke and M. Vielhaber, "Towards a Process Model for the Development of Light, Mechatronic Products," *Proceedings of NordDesign 2012, the 9th NordDesign conference*, 2012.

[43] Handson Technology, "Handson Technology User Guide L298N Dual H-Bridge Motor Driver." www.handsontec.com (accessed May 17, 2022).

[44] D. Valme, K. Kudelina, D. Belolipetskaja, A. Rassõlkin, T. Vaimann, and A. Kallaste, "Generative Design in Weight Optimization of Reconfigurable Continuous Track Robot," *28th International Workshop on Electric Drives: Improving Reliability of Electric Drives (IWED)*, 2021.

[45] Pololu Corporatiopn, "25D Metal Gearmotors," 2021. Accessed: May 17, 2022. [Online]. Available: www.pololu.com

[46] HobbyKing, "Turnigy nano-tech 1300mAh 4S 45~90C Lipo Pack," 2022. https://hobbyking.com/en_us/turnigy-nano-tech-1300mah-4s-45-90c-lipo-pack.html (accessed May 17, 2022).

[47] KeepPower, "KeepPower 3400mAh li ion protected 18650 battery NCR18650B," 2022. https://www.keeppower.com.cn/products_detail.php?id=375 (accessed May 17, 2022).

[48] ProtoSupplies, "Servo Motor Micro SG90 - ProtoSupplies." https://protosupplies.com/product/servo-motor-micro-sg90/ (accessed May 17, 2022).

[49] Guangzhou HC Information Technology, "HC-06 Bluetooth module datasheet," 2006. Accessed: May 17, 2022. [Online]. Available: https://eu.mouser.com/datasheet/2/306/hc06-1324356.pdf

[50] SHENZHEN GSM ELECTAONIC CO, "DC to DC Converter ," 2021. Accessed: May 17, 2022. [Online]. Available: http://www.086ic.cn

[51] DFRobot, "7A Dual DC Motor Driver." https://wiki.dfrobot.com/7A_Dual_DC_Motor_Driver_SKU__DRI0041 (accessed May 17, 2022).

[52] Benewake CO, "TF-Luna Instruction Manual," 2021. Accessed: May 17, 2022. [Online]. Available: http://benewake.com/en/mfeedback.html

[53] "Depth Camera D435i – Intel® RealSense™ Depth and Tracking Cameras." https://www.intelrealsense.com/depth-camera-d435i/ (accessed May 17, 2022).

[54] Sparkfun, "Ultrasonic Distance Sensor - HC-SR04 - SEN-15569 - SparkFun Electronics." https://www.sparkfun.com/products/15569 (accessed May 17, 2022).

APPENDICES

Appendix 1 Motion control unit main code

```
#include <Servo.h> //Including servo library

// A class to compute the control signal
class PIDcontrol{
  private:
    float kp, kd, ki, CSmax, CSmin; // Parameters
    float eprev, eintegral; // Storage

  public:
  // Constructor
  PIDcontrol() : kp(1), kd(0.02), ki(0), CSmax(255), CSmin (180), eprev(0.0),
eintegral(0.0){}
  // A function to set the parameters
  void setParameters(float kpIn, float kdIn, float kiIn, float CSmaxIn, float CSminIn){
    kp = kpIn; kd = kdIn; ki = kiIn; CSmax = CSmaxIn, CSmin = CSminIn;
  }

  // FUnction for the calculation of the control signal
  void evalu(int PRvalue, int SETvalue, float deltaT, int &pwr, int &dir, int &done){
    // Calculating error
    int error = SETvalue - PRvalue; //
    // Calculating derivative parameter
    float dedt = (error-eprev)/(deltaT);
    // Calculating integral parameter
    eintegral = eintegral + error*deltaT;
    // Calculating control sigral value
    float u = kp*error + kd*dedt + ki*eintegral;

    //Transforming control signal value to motor conrol function input
    pwr = (int) fabs(error);
    if ( pwr > CSmax ){
      pwr = CSmax;
    }
    if ( pwr < CSmin ){
      pwr = CSmin;
    }

    // Setting direction of the motor rotation
    dir = 1;
    if(error<0){
      dir = -1;
    }

    if ( fabs(error) < 0.3 ){
      dir = 0;
      done = 1;
    }

    // Storing previous error
    eprev = error;
  }
};

// Setting up servos
Servo myservoL;
Servo myservoR;
#define servoPinL 9 // Defining left servo pin
```

```
#define servoPinR 10 // Defining right servo pin

// Setting up DC motors
#define MOTORSNUM 2 // Defining the number of DC motors
char app_out = 0; //Command recieved via the Bluettoh module
int power = 0; // Variable of enabling control of the robot

// Defining the pins of the motors
const int encA[] = {2,3};
const int encB[] = {4,5};
const int pwm[] = {6,11};
const int IN1[] = {7,12};
const int IN2[] = {8,13};


long prevT = 0;
volatile int posi[] = {0,0};

// PID class instances
PIDcontrol pid[MOTORSNUM];

void setup() {
  Serial.begin(9600);
  for(int n = 0; n < MOTORSNUM; n++){
    pinMode(encA[n],INPUT);
    pinMode(encB[n],INPUT);
    pinMode(pwm[n],OUTPUT);
    pinMode(IN1[n],OUTPUT);
    pinMode(IN2[n],OUTPUT);
    pid[n].setParameters(0,0,0,255);
  }

  attachInterrupt(digitalPinToInterrupt(encA[0]),readEnc<0>,RISING);
  attachInterrupt(digitalPinToInterrupt(encA[1]),readEnc<1>,RISING);
  myservoL.attach(servoPinL); //Attaching servo variable to a pin
  myservoR.attach(servoPinR); //Attaching servo variable to a pin
}

void loop() {
  int done[MOTORSNUM] = {0,0}; // Variable responsible for the link locking process state
  int pos[MOTORSNUM]; // Variable for storing the position of the motor

  myservoL.write(0);
  delay(1000); //Bringing the left servo to the initial position
  myservoR.write(0);
  delay(1000); //Bringing the right servo to the initial position

  // Reading the command recieved from the smartphone application
  if (Serial.available()>0){
    app_out = Serial.read();
  }

// Turning ON the system
  if (toide == 0 && app_out == 'e')
  {
    toide = 1;
    app_out = 0;
  }

// Turning OFF the system
  if (toide == 1 && app_out == 'e')
  {
    toide = 0;
    app_out = 0;
  }

//Moving FORWARD
  if (power == 1 && app_out == 'f'){

     for(int n = 0; n < MOTORSNUM; n++){ // Safety stop
      conMotor(0, 0 ,pwm[n],IN1[n],IN2[n]);
      delay(500);
    }

    for(int n = 0; n < MOTORSNUM; n++){ //Enabling motors
      conMotor(1, 220 ,pwm[n],IN1[n],IN2[n]);
    }
  delay(500);
  app_out = 0;
  }

//STOPPING the motors
  else  if (power == 1 && app_out == 's'){
    for(int n = 0; n < MOTORSNUM; n++){
      conMotor(0, 0 ,pwm[n],IN1[n],IN2[n]);
      }
  app_out = 0;
  delay(500);
  }

//Moving REVERSE
  else  if (power == 1 && app_out == 'r'){
     for(int n = 0; n < MOTORSNUM; n++){ // Safety stop
      conMotor(0, 0 ,pwm[n],IN1[n],IN2[n]);
    }

    for(int n = 0; n < MOTORSNUM; n++){ //Enabling motors
      conMotor(-1, 220 ,pwm[n],IN1[n],IN2[n]);
    }
  app_out = 0;
  delay(500);
  }

// Locking the right link under 0 degrees
  else if (app_out == 'c'){
    myservoR.write(180);
    delay(1000); //Bringing the right servo to the locking position
    while (done[1] == 0){
      // set SETvalue position
      int SETvalue[MOTORSNUM];
      SETvalue[0] = 593; // Number of pulses required to rotate the sprocket wheel by 1 link
      SETvalue[1] = 593; // Number of pulses required to rotate the sprocket wheel by 1 link

      // Calculating time difference
      long curT = micros(); //Reading time value since Arduino starting running the program
      float deltaT = ((float) (curT - prevT))/( 1.0e6 ); //Calculating the time difference
      prevT = curT; // Updating previous time value

      // Reading position of the motor
      noInterrupts(); // disable interrupts temporarily while reading
      for(int n = 0; n < MOTORSNUM; n++){
          pos[n] = posi[n];
        }
      interrupts(); // turn interrupts back on

      // Loop through the motors
      for(int n = 0; n < MOTORSNUM; n++){
        int pwr, dir;
        // Calculate control paramaters
        pid[n].evalu(pos[n], SETvalue[n], deltaT, pwr, dir, done[n]);
        conMotor(dir,pwr,pwm[n],IN1[n],IN2[n]);
        }
      }

      //Cleaning the values after finishing
      for(int n = 0; n < MOTORSNUM; n++){
        pos[n] = 0;
        posi[n] = 0;
      }

      for(int n = 0; n < MOTORSNUM; n++){ // Stopping the motors
        conMotor(0, 0 ,pwm[n],IN1[n],IN2[n]);
      }
      app_out = 0;
  }

// Locking the left link under 15 degrees
  else if (app_out == 'd'){
    myservoL.write(180);
    delay(1000); //Bringing the right servo to the locking position
    while (done[1] == 0){
      // set SETvalue position
      int SETvalue[MOTORSNUM];
      SETvalue[0] = 593; // Number of pulses required to rotate the sprocket wheel by 1 link
      SETvalue[1] = 593; // Number of pulses required to rotate the sprocket wheel by 1 link

      // Calculating time difference
      long curT = micros(); //Reading time value since Arduino starting running the program
      float deltaT = ((float) (curT - prevT))/( 1.0e6 ); //Calculating the time difference
      prevT = curT; // Updating previous time value

      // Reading position of the motor
      noInterrupts(); // disable interrupts temporarily while reading
      for(int n = 0; n < MOTORSNUM; n++){
          pos[n] = posi[n];
        }
      interrupts(); // turn interrupts back on

      // Loop through the motors
      for(int n = 0; n < MOTORSNUM; n++){
        int pwr, dir;
        // Calculate control paramaters
        pid[n].evalu(pos[n], SETvalue[n], deltaT, pwr, dir, done[n]);
        conMotor(dir,pwr,pwm[n],IN1[n],IN2[n]);
        }
      }

      //Cleaning the values after finishing
      for(int n = 0; n < MOTORSNUM; n++){
        pos[n] = 0;
        posi[n] = 0;
      }

      for(int n = 0; n < MOTORSNUM; n++){ // Stopping the motors
        conMotor(0, 0 ,pwm[n],IN1[n],IN2[n]);
      }
      app_out = 0;
  }

  // Neutral state
  else {
    for(int n = 0; n < MOTORSNUM; n++){
      conMotor(0, 0 ,pwm[n],IN1[n],IN2[n]);
    }
  delay(500);
  app_out = 0;
  }
}


// FUNCTIONS

// Function to control the speed of and the direction of the motors using control signal
void conMotor(int dir, int pwmValue, int pwm, int IN1, int IN2){
  analogWrite(pwm,pwmValue);
  if(dir == 1){
    digitalWrite(IN1,HIGH);
    digitalWrite(IN2,LOW);
  }
  else if(dir == -1){
    digitalWrite(IN1,LOW);
    digitalWrite(IN2,HIGH);
  }
  else{
    digitalWrite(IN1,LOW);
    digitalWrite(IN2,LOW);
  }
}

// Function to read the encoder value
template <int x>
void readEnc(){
  int y = digitalRead(encB[x]);
  if(y > 0){
    posi[x]++;
  }
  else{
    posi[x]--;
  }
}
```

2021 28th International Workshop on Electric Drives: Improving Reliability of Electric Drives (IWED), Moscow, Russia. Jan 27 – 29, 2021

Generative Design in Weight Optimization of Reconfigurable Continuous Track Robot

Daniil Valme
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
davalm@taltech.ee

Karolina Kudelina
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
karolina.kudelina@taltech.ee

Diana Belolipetskaja
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
dibelo@taltech.ee

Anton Rassõlkin
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
anton.rassolkin@taltech.ee

Toomas Vaimann
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
toomas.vaimann@taltech.ee

Ants Kallaste
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
ants.kallaste@taltech.ee

***Abstract*—Rapid prototyping of mechatronic devices is becoming more and more popular nowadays. The rapid development of additive manufacturing technologies and a wide range of available open-source microcontrollers allows realizing ideas on both hobby and university project levels. Current work presents a weight optimization of the reconfigurable continuous track robot, developed as a student theses project at Tallinn University of Technology. The paper describes prototype development stages and post-development optimization methodology for optimal motor-drive selection based on the requirements of traction forces. As one of the trending methods for improving mechanical construction, the generative design technique was selected to further development of the prototype, and to reduce the rated power of the motor-drive system required for developed platform. Moreover, the time needed for the printing of the prototype was reduced.**

Keywords— Design optimization, Mobile robots, Motion planning, Robot control

I. Introduction

In recent centuries a significant improvement has been achieved in the development of land transport vehicles. During this period, the main concepts and the features of the vehicles were developed. However, some technical challenges exist, especially in the vehicles used in the harsh environment where the traditional wheels may not be useful. One of the world leaders in robotics, Boston dynamics developing bioinspired robots, its latest generation of legged machines can trudge up and down hills, clamber over obstacles, and even leap into the air like a gymnast [1].

However, there are also available some more classical solutions that are using swinging [2] or transformable [3] wheels. Simultaneously, developed as a "portable railway", tracked vehicles showed outstanding performance in agriculture, construction, logging, mining, off-road transport, recreation, and military operations since the turn of this century [4]. Withal, tracked vehicles facing problems when they need to crawl over obstacles. Literature review gives different solutions, mostly related to variable geometry tracked mechanism [5], [6], but the most attractive is reconfigurable track design presented by T. Kislassi and D. Zarrouk in [7].

The reconfigurable continuous track robot (RCTR), presented in this paper, due to reconfigurable geometry, can crawl over obstacles and adapt its external shape as it advances. RCTR design is modular, and its geometry can be modified. RCTR developed in recent work is inspired by one presented in [7], but was created from scratch by the students from Tallinn University of Technology (TalTech). The first design of the RCTR is presented in Fig. 1.

The small number of actuators in such type of RCTR design increases the system's reliability and decreases the weight, size, and cost [7]. However, the generative design process's implementation will allow even more efficient weight reduction, which will relieve the load on the traction drive.

In recent years, the generative design method has become more prevalent [8]. Generative design uses evolutionary algorithms to produce optimized detail. During the detail evolution cycle, the algorithm uses the results of each iteration to produce output with physical properties pre-determined by the user [9]. This paper describes the process of weight optimization using Autodesk Fusion 360 software Generative design toolbox. The paper presents a brief description of

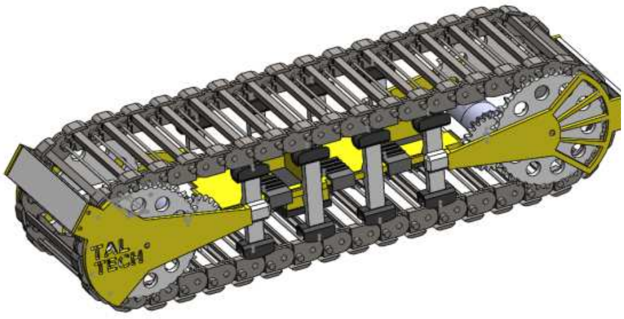

Fig. 1. General assembly of the reconfigurable continuous track robot (RCTR) developed in Tallinn University of Technology (TalTech)

The research has been supported by the Estonian Research Council under grant PSG453 "Digital twin for propulsion drive of autonomous electric vehicle".

978-0-7381-4443-6/21/$31.00 ©2021 IEEE

RCTR that follows with weight optimization and traction force calculation examples.

II. Development of the Track Robot

During the last years, several robotic concepts have been developed for use in complex terrain conditions. The existing solutions can be classified according to the principle of movement into four groups [10]: wheeled robots, legged robots, hybrid robots, track robots.

Each concept has its advantages and disadvantages that must be taken into account. First of all, it is the conditions under which the robotic system will be used. The universal concept is a RCTR, which has an even load distribution on the ground, ease of management, and better resistance to extreme conditions.

A. Electrical part

The following criteria were considered to select a development board for the RCTR: power consumption, development board price, development options, number of input-output pins and their functionality, program memory capacity, data memory capacity, number of timers, central processing unit (CPU) operating frequency. To select the solution, there were also considered the following criteria:

- the development board must have a functional environment,
- the development board must have user support,
- the overall dimensions of the development board must be possibly compact,
- development board must support smartphone control.

The development board is also supposed to allow the following components: two DC motors, driver, battery, servomotor, switch, and communication module. Taking these criteria into account, Arduino, an open-source single-board microcontroller, solutions were chosen. The pulse-width modulation (PWM) can be implemented on the Arduino in several ways, with a variety of 17 official versions of the Arduino microcontroller and an immense amount of expansion boards (shields) to provide different application extension, such as DC and SRM motors control, visualization, sensors, actuators, etc., gives an outstanding possibility to use Arduino boards for fast prototyping [11].

One of the main parts of the RCTR is an electric motor. The selection of electric motor depends, first of all, on the type of supply voltage to be used. Since the RCTR is supposed to drive in an outdoor environment, the system needs a portable power supply (battery), which provides DC. Besides that, the speed of the electric motor must be easily regulated. The brushed DC motor was selected as a primary propulsion device.

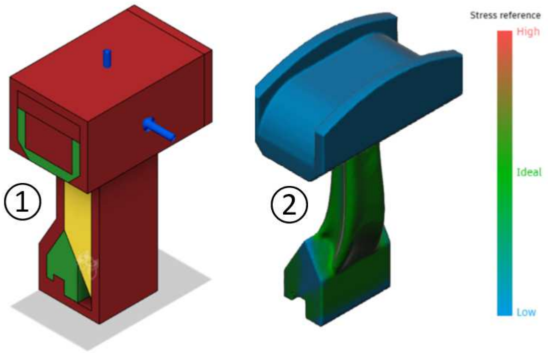

Fig. 2. Example of defining input geomertry example of stress analysis.

B. Mechanical part

The mechanical design of the RCTR was based on the electrical components used in platform – all the mechanical parts were constructed originating from the electrical components. Mechanical design of the track motor supposed to meet the following criteria:

- the platform must have a flexible and robust internal support framework,
- platform elements are easy to be disposed of and replaced,
- cooperation of the platform elements is easy to be monitored,
- platform track must be flexible and allow tests for special solutions,
- it must be possible to drive forthright on the platform,
- the platform must have a user-friendly management system.

The initial design was divided into the following groups: track, support frame, front motor housing, back motor housing, transmission mechanism, lock mechanism.

C. Control system

For creating the control system of the RCTR, the fact that the system must be adaptable and easily regulated was taken into account. Therefore, a smartphone-based solution was used to develop a user interface for the Arduino microcontroller. Using a smartphone-based application, the corresponding signals can be easily transmitted to the platform development board and after the controller apply a related part of the code. This solution allows the control system to control the speed, movement direction and turn on (/off) the control system.

III. Weight Optimization

Weight optimization is typically used to decrease the weight of the detail and meet detail performance requirements. The goal of weight optimization is to safely distribute loads inside the body [12]. Reducing the weight leads to more effective use of material used in detail production and costs, e.g., in additive manufacturing, also known as 3D printing. Using the additive manufacturing process to produce highly complicated structures is a novel and rapidly developing prototyping trend [13]. In the case of 3D printing technology, the smaller weight helps decrease printing time, which accelerates the whole prototyping/product development process and reduces the quantity of material. In automotive design, weight reduction is one of the main ways to decrease CO2 emissions. Therefore, plenty of weight reduction methods and design guidelines were developed, and defined with standards [14].

To start weight optimization of the RCTR platform, it was decided to choose three details of the platform's inner frame construction: chain support, middle plate, and sprocket wheel.

Firstly, the 3D models of the details were created. Then these parts were divided into sub-parts taking into consideration the structural loads and constraints. Fig. 2 represents an example of input geometry definition and stress simulation view. To define input structures for generative design, there were three possible types of geometry used which are:

- Preserve geometry is represented as green areas. To select the detailed geometry that can be changed during the generative design process, each part's functional blocks were defined. The functional block is a part of the detailed design that cannot be changed due to its functional task.
- Starting shape geometry is yellow-colored and can be changed during the generative design creation process.
- Obstacle geometry marked with red color creates the space in which the detail's new geometry can be generated. In the case of RCTR parts, the details' obstacle geometry was similar to detail original shape due to components interaction features.

Each optimized detail had an individual set of loads that were calculated according to its operating conditions and functional task. Then, it was supposed to combine loads into load cases, simulating different circumstances. For generative design, acrylonitrile butadiene styrene (ABS) was selected as the material for all details.

A. Sprocket wheel

In the sprocket wheel, there were three most essential load types presented: electrical motor torque, RCTR's weight, and sideload. To estimate the impact of the DC motor's torque T (Nm) on the sprocket wheel, the circumferential force was calculated, taking into consideration sprocket pitch diameter d_p [1]:

$$F_c = \frac{2000 \cdot T}{d_p} = \frac{2000 \cdot 075}{109.4} = 13.7\ N \qquad (1)$$

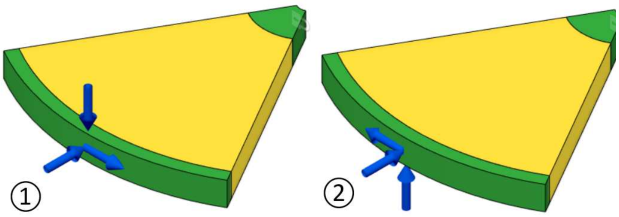

Fig. 5. Sprocket wheels load cases, for the left (1) and right (2) sides.

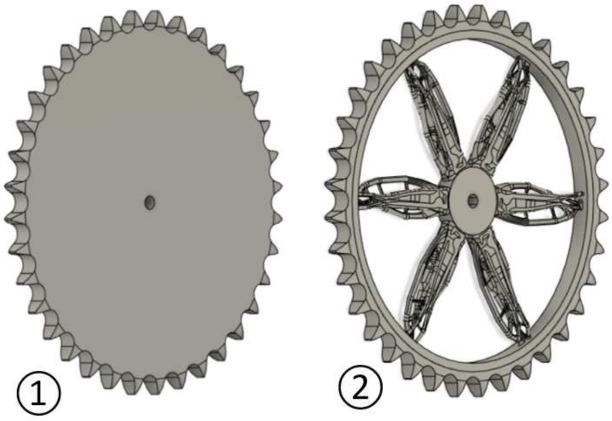

Fig. 6. Sprocket wheel before (1) and after (2) generative design optimization.

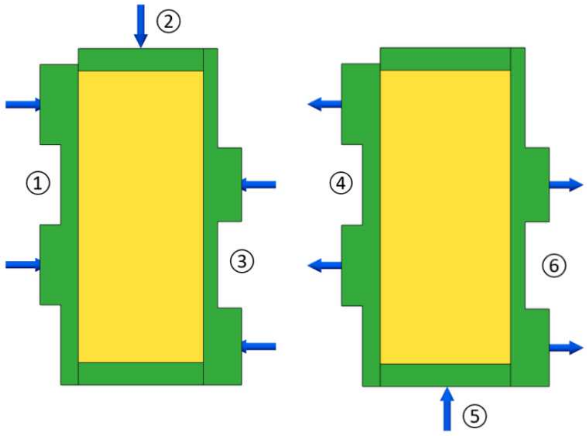

Fig. 3. Frame plate load cases: 1 and 3 - compression loads; 2 and 5 - side loads; 4 and 6 - stretching loads.

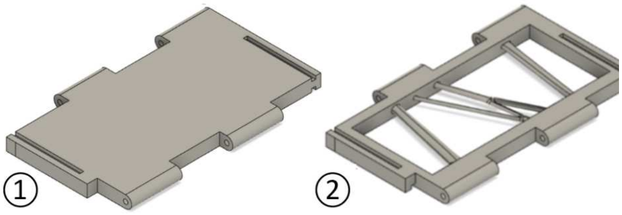

Fig. 4. Middle plate before (1) and after (2) generative design optimization.

The gravitational force's value equals 5.5 N and is represented as the RCTR platform's weight equally distributed between four sprocket wheels. Side load was added as an external load applied to the wheel's side surface. The approximate value of side load was calculated as half of weight 11 N. To create the sprocket wheel's design, and there were two load cases considered, as shown in Fig. 3. The direction of sprocket circumferential force and side load vectors in both cases had the opposite direction to simulate different load conditions. The optimization result of the sprocket wheel is shown in Fig. 4.

B. Frame plate

The frame plate had two types of loads applied: compression loads and stretching loads. As shown in Fig. 6, load cases 4 and 6 represent the stretching load, when the force of 22 N is applied to the inner surface on plate holes. Load cases 1 and 3 illustrate a similar situation but for compression loads. Load cases 2 and 5 are represented with side loads that imitate the impact of external forces affecting the frame plates' side surface. The optimization result of the frame plate is shown in Fig. 7.

C. Chain support

For the support chain, there were two loads takes into consideration. Weight of the RCTR platform 22 N applied on the upper surface of the detail. Also, there was a side load of 10 N applied on the side surface. Two load cases had the

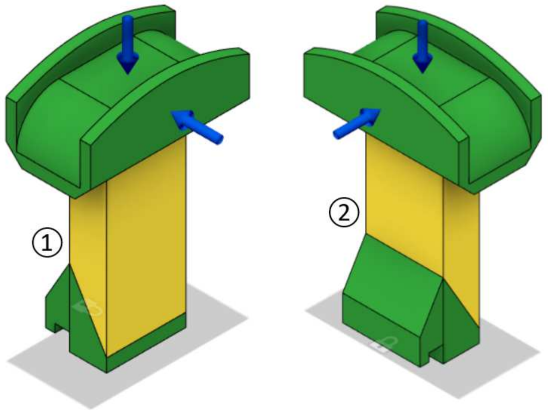

Fig. 8. Chain support load cases, for the left (1) and right (2) sides.

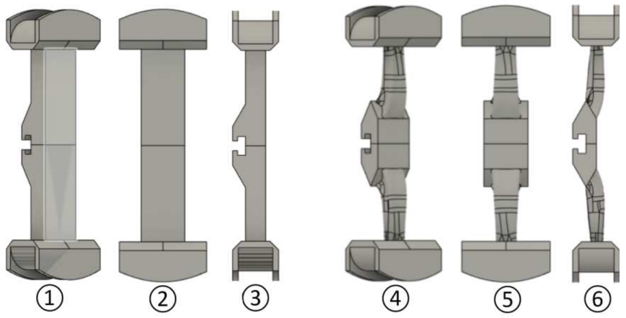

Fig. 9. Chain support geomtry before (1-3) and after (4-6) generative design optimization

same structure, but the side load vectors were used on opposite side surfaces, as shown in Fig.8. The optimization result of the chain support is shown in Fig. 9.4-6.

Two key parameters were selected to evaluate the impact of generative design implementation, such as printing time and material consumption. For calculating the printing time, the following parameters were used:

- Material – ABS,
- Infill rate 90% or 100%,
- Printing speed – 50 mm/s.

TABLE I. PRINTING TIME

Parameter	Sprocket wheel	Body plate	Support leg
Infill rate before, %	90	90	90
Time before, h	10 h 12′	7 h 34′	4 h 9′
Infill rate after, %	100	90	90
Time after, h	9 h 15′	3 h 26′	3 h 40′
Time change, h	57′	4 h 8′	29′
Time change. %	9,3%	54,6%	11,6%

To print the sprocket wheel with a modified design, the layer height was changed from 0.2 mm to 0.12 mm as the part geometry required a more precise printing resolution. Also, a smaller number of supports was used, and the infill rate was increased from 90% to 100%. From the calculation results presented in Table I, it is possible to conclude that generative design has a different influence on print time duration. Generally, the time required for printing is smaller comparing to original details. However, in the case of a more complex design, the time change is insignificant. Time optimization results are shown in Table II.

TABLE II. MATERIAL CONSUMPTION

Parameter	Sprocket wheel	Body plate	Support leg
Infill rate before, %	90	90	90
Mass before, g	55	41	24
Infill rate after, %	100	90	90
Mass after, g	23	18	17
Mass change, g	32	23	7
Mass change, %	58,2%	56,1%	29,2%

Using of generative design had a significant influence on RCTR parts' masses. The decrease in mass varied from 29.2% to 58.2%, accordingly. Weight optimization results are shown in Table III.

TABLE III. ASSEMBLY MASS CHANGE

Nr.	Detail Parameters			
	Part name	Number of parts in the assembly	Change of the mass, g	Mass change in assembly, g
1	Sprocket wheel	4	32	132
2	Frame plate	2	23	46
3	Chain support	8	7	56
			Total:	234

IV. MOTOR-DRIVE LOAD CALCULATION

To calculate the total traction force needed to start the RCTR movement, there should be taken into account all the forces having an impact on the body during the RCTR platform operation.

As shown in Fig. 10, several forces have an impact on the climbing-up track. First, the gravitational force *mg* acts vertically downwards (red arrow). Then, normal force *N* acts perpendicularly to the climbing surface (green arrow). It is driving force *F* that operated in parallel to the driving direction (blue arrow). And frictional force F_H that operates in the opposite direction with the driving force (yellow arrow).

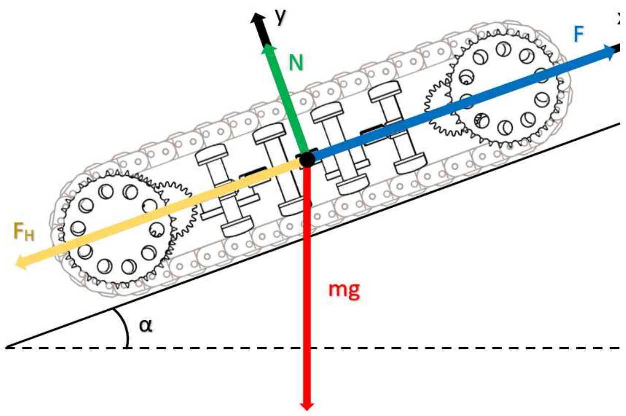

Fig. 10. Forces diagram of reconfigurable continuous track robot (RCTR).

Based on Newton's second law, the vector sum of all forces is equal to the product of body mass and acceleration, as follows in the equation:

$$F_H = \mu \cdot F_N \tag{2}$$

where F_H is a friction force (N), μ is a friction factor, and F_N is a normal force (N). The system equation is as follows:

$$F = m \cdot (a + \mu \cdot g \cdot cos\alpha + g \cdot sin\alpha) \tag{3}$$

where m is the weight of the RCTR platform (kg), a is an acceleration (m/s^2), g is a gravity acceleration (9,81 m/s^2), and α is a tilt angle (°).

All parameters required for calculating motor parameters, except for the tilt angle, are taken from Table IV. In calculations, there were considered the maximum values of the parameters.

TABLE IV. RECONFIGURABLE CONTINUOUS TRACK ROBOT (RCTR) PARAMETERS

Parameter	Value
Length, mm	542
Wight, mm	139
Weight, kg	1.945
Avarage speed, m/s	0.1-0.3
Acceleration time, s	1-2
Acceleration, m/s^2	0.2
Angular velocity, rad/s	5.48
Wheel radius, mm	0.0547

Tilt angle was taken from 0° to 90° in 15° increment angle to see the traction force change. Firstly, there were calculated minimum values of the required traction force depending on the inclination angle. The calculation results are shown in Table V.

TABLE V. TRACTION FORCE CALCULATION RESULTS

Angle	Traction force, N
0°	16,61
5°	18,21
10°	19,67
15°	20,99
20°	22,16
25°	23,15
30°	23,97
35°	24,62
40°	25,08
45°	25,35

As seen, the maximum traction force value is at an angle of 15°. This value was used to calculate the motor torque. The equation is as follows:

$$M = F \cdot D \tag{4}$$

where D is a maximum wheel radius. As a result, the motor torque is equal to 1.37 Nm. Accordingly, there can be found a motor power:

$$P = M \cdot \omega \tag{5}$$

where on the strength of the given equation, the motor power is equal to 7.54 W.

V. CONCLUSION

Traditionally, prototyping is an experimental process for implementing the designers' ideas into tangible forms from paper to digital. In the early stage, it seems impossible to consider an optimal solution for the product's different components. A post-development optimization methodology aimed to improve the performance and cost of further prototypes. With that research work and implementation of generative design algorithms in RCTR weight, optimization reduced the platform's mass by 234 grams, which equals 10.7% of the total weight. The output geometry of produced details was more complicated. Moreover, the printing time and material consumption in all cases decreased.

Further development of the platform will include an investigation in the development of achieving autonomy of the RCTR by implementing the additional sensors for the obstacles' detection.

REFERENCES

[1] E. Guizzo, "By leaps and bounds: An exclusive look at how Boston dynamics is redefining robot agility," *IEEE Spectr.*, vol. 56, no. 12, pp. 34–39, Dec. 2019, doi: 10.1109/MSPEC.2019.8913831.

[2] J. T. Karras *et al.*, "Pop-up mars rover with textile-enhanced rigid-flex PCB body," in *Proceedings - IEEE International Conference on Robotics and Automation*, Jul. 2017, pp. 5459–5466, doi: 10.1109/ICRA.2017.7989642.

[3] Y. S. Kim, G. P. Jung, H. Kim, K. J. Cho, and C. N. Chu, "Wheel Transformer: A wheel-leg hybrid robot with passive transformable wheels," *IEEE Trans. Robot.*, vol. 30, no. 6, pp. 1487–1498, Dec. 2014, doi: 10.1109/TRO.2014.2365651.

[4] J. Y. Wong, "Dynamics of tracked vehicles," *Veh. Syst. Dyn.*, vol. 28, no. 2–3, pp. 197–219, 1997, doi: 10.1080/00423119708969354.

[5] S. K. Lim, D. Il Park, Y. K. Kwak, B. S. Kim, and S. W. Jeon, "Variable geometry single-tracked mechanism for a rescue robot," Jul. 2005. doi: 10.1109/ssrr.2005.1501231.

[6] J.-L. Paillat, P. Lucidarme, and L. Hardouin, "Variable Geometry Tracked Vehicle (VGTV) prototype : conception, capability and problems.," *undefined*, 2008.

[7] T. Kislassi and D. Zarrouk, "A Minimally Actuated Reconfigurable Continuous Track Robot," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 652–659, Apr. 2020, doi: 10.1109/LRA.2019.2959237.

[8] A. Lukin *et al.*, "Generative Design in Development of Mechanical Components for Magnus Effect-Based Wind Turbine," *2020 XI Int. Conf. Electr. Power Drive Syst.*, 2020.

[9] G. S. Hornby;, H. Lipson;, and J. B. Pollack, "Evolution of generative design systems for modular physical robots," *Proc. 2001 ICRA. IEEE Int. Conf. Robot. Autom.*, 2001.

[10] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, D. M. Helmick, and L. Matthies, "Autonomous stair climbing for tracked vehicles," *Int. J. Rob. Res.*, vol. 26, no. 7, pp. 737–758, Jul. 2007, doi: 10.1177/0278364907080423.

[11] A. Rassolkin *et al.*, "Control Challenges of 3D Printed Switched Reluctance Motor," in *26th International Workshop on Electric Drives: Improvement in Efficiency of Electric Drives (IWED)*, Mar. 2019, p. 4, doi: 10.1109/IWED.2019.8664282.

[12] M. M. Varadharajan, "Optimization of Structures Shape and Weight Optimization of Structural Members," 2013.

[13] A. Kallaste, T. Vaimann, and A. Rassõlkin, "Additive Design Possibilities of Electrical Machines," in *59th International Scientific Conference on Power and Electrical Engineering of Riga Technical University (RTUCON)*, 2018, pp. 1–5.

[14] T. Luedeke and M. Vielhaber, "Towards a Process Model for the Development of Light, Mechatronic Products," *Proc. Nord. 2012, 9th Nord. Conf.*, 2012.

Preliminary Sensors Selection for Reconfigurable Continuous Track Robot Obstacle Detection

Daniil Valme, Karolina Kudelina, Anton Rassõlkin
Department of Electrical Power Engineering and Mechatronics
Tallinn University of Technology
Tallinn, Estonia
davalm@taltech.ee

Abstract — **Nowadays, the topic of autonomous vehicles is attracting attention. Such technologies and innovative approaches can help to facilitate people's life in different. However, practical implementation of autonomous vehicles requires innovation in different spheres such as regulations, cybersecurity, and ethics. The demand for applying autonomous vehicles in rugged terrain conditions is growing, bringing new challenges for both obstacle overcoming and detection. This paper describes a process of preliminary sensors selection obstacle detection in an autonomous continuous track robot. Different sensors are studied and chosen for a particular case. The initial algorithm for robot navigation is proposed.**

Keywords – autonomous vehicles, objects recognition, path planning, land vehicles, robot control

I. Introduction

In recent decades, land transport vehicles have been achieved significant improvement and development. One of the key questions in autonomous navigation is obstacle detection. Obstacle avoidance is an essential embodiment of the land mobile robot and the fundamental guarantee to perform various tasks [1]. This problem is widely discussed in the literature. Catapang and Ramos in [2] used 2D LiDAR (Light Detection And Ranging) sensor system in the obstacle detection via clustering for autonomous vehicles. Rozsa and Sziranyi in [3] find a solution for obstacle categorization without shape modeling using low-level scattered data sources for semantic level interpretation during the motion. Nowadays, artificial intelligence-based algorithms are used in different applications and propose intelligent solutions for various problems [4], and Ramos et al. in [5] suggest detecting unexpected obstacles for self-driving vehicles using deep learning and geometric modeling. Nguyen et al. in [6]propose a study that combines a novel deep learning approach to yield robust on-road vehicles providing its detection, recognition, and tracking. The robot can be used for assistance or inspection in difficult terrain conditions such as urban or industrial environments or forest areas [7]. However, an application of robots in natural areas (e.g., forest) conditions may be complicated to the uncertainty of the surrounding environment object shape.

According to the movement principle, several robotic concepts can be classified as follows: wheeled robots, legged robots, hybrid robots, and track robots [8]–[10]. Each of the concepts mentioned above has its benefits and drawbacks. However, an up-and-coming solution, a reconfigurable continuous track robot (RCTR) with an even load distribution on the ground, ease of management, and better resistance to extreme conditions, is not widely discussed. RCTR has several prototypes such as [11] or [12], one of them is shown in Fig. 1.

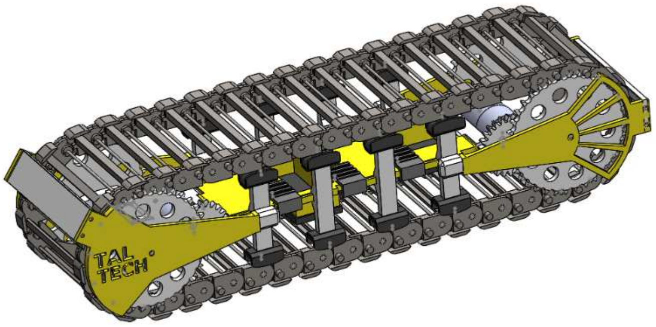

Fig. 1. Studied reconfigurable continuous track robot (RCTR).

In this paper, the further development of RCTR developed in Tallinn University of Technology (TalTech) is presented. Section II presents a sensors' overview and preliminary selection methodology based on the RCTR functionality. Section III describes the control algorithm to avoid obstacles. The conclusion summarize the article and define future works.

II. Sensors for RCTR

A vehicle (incl. RCTR) with autonomous functionality must be provided with accurate and robust real-time perception systems. Implementation of autonomous vehicles gives many advantages such as optimized traffic, quicker response to changing the environment, and ability to operate under challenging conditions. On the other side, this technology requires new approaches and solutions in other fields as it is closely related to the legal framework, cybersecurity, and privacy regulations [13]. A classification standard J3016 [14], firstly published in 2014 by SAE International (Society of Automotive Engineers), explains car driving automation levels and is based mainly on the amount of driver intervention and attentiveness required. The latest update of the standard was

The research has been supported by the Estonian Research Council under grant PSG453 "Digital twin for propulsion drive of an autonomous electric vehicle".

978-1-6654-4928-1/21/$31.00 ©2021 IEEE

done in 2018 and defined six levels of automation. 0 level corresponds to the vehicle with no driving automation systems, and level 5 represents a fully automated system [13].

Navigation in an environment requires detailed information such as distance to the closest objects, object's types, speed, color, etc., for that the multiple sensors are used. Traditionally there are distinguished two main types: active and passive sensors [15]. Active sensors have their energy source, and for collecting the information, energy is transmitted by them. LiDARs, radars (radio detection and ranging), time-of-flight (ToF) cameras, and ultrasonic sensors relate to them. The sensors that do not have their energy source are called passive sensors, using energy from an external source. Visual cameras and infrared are examples of passive sensors. Active sensors are typically more expensive. However, they are more effective in position and speed measurement. Camera-based solutions, on the other hand, can provide detailed information about object type [16]. Defining the vehicle's exact position is usually done with Inertial Measurement Units (IMU), odometers, and Global Navigation Satellite Systems (GNSS) [17].

A. LiDARs

This is a remote sensing technology that gives accurate depth information about the surrounding environment. LiDAR systems base their operation on the ToF principle measurements of pulsed light emitted from a laser diode until an emitter receives it. LIDARs can be classified according to the type of information they obtain from the environment in 2D or 3D LiDARs or give according to their construction rotary or solid-state [18]. Comparing to cameras, they are more reliable as they do not depend on weather conditions. Typical LiDARs are not able to capture the fine structures of the object. However, in recent times the new types of LiDARs were developed, e.g., flash LiDARs give detailed object information like camera images, and Frequency Modulated Continuous Wave LIDARs can also provide velocity information [17].

B. Radar

Radars use electromagnetic waves to measure the distance to the object. It sends out a signal during the measurement, and then it waits until it is sent back. The frequency of the sent back signal in case of mutual movement slightly changes (Doppler's effect), so it is possible to instantly calculate the objects' radial velocity and direction of moving. Similar to LiDARs, radars are robust to weather conditions. However, radars cannot define the objects' shape due to low resolution and may struggle with particular material [13], [18].

C. Ultrasonic Sensors

In this case, sonic waves are generated by a magnetically resistive membrane to measure an object's distance. The principle is also based on the ToF measurement of the sonic wave when emitted until the echo is received [18]. The obtained data quality strongly depends on air humidity, temperature, and dust particles due to sensing properties.

D. Cameras

Images can be captured by visual (VIS) and thermal (IR) cameras that could provide detailed texture information of a vehicle's surroundings. VIS cameras are sensitive to weather and lighting conditions, whereas thermal cameras are more robust to daytime/nighttime changes as they detect infrared radiation related to heat from objects.

There are two types of camera vision-based techniques: monocular vision and stereo vision. In the case of monocular, there is only one camera used [19]. The main disadvantage of such a method is the inability to measure the depth. In stereo vision-based approach uses two cameras instead of one. It allows stereoscopy vision to be performed and adds a new channel called depth information. Cameras with such properties are called RGB-D cameras or depth cameras [18].

Different types of camera sensors used in autonomous navigation are ToF cameras used to get the 3D image of objects. ToF cameras emit near-infrared light pulses and measure the difference in phase between modulated signal emitted and signal received to compute the distance.

E. Hyperspectral Imaging

There are advanced sensors that use hyperspectral imaging in order to get detailed information about the environment. Such techniques help to find objects, identify the material and detect specific processes [20]. Hyperspectral image sensors in the context of autonomous navigation can be implemented in off-road conditions. In complex terrain conditions, the objects such as swamps, water puddles, or wetlands cannot be easily detected due to similar RGB image texture [21].

F. Inertial Measurement Unit

Additionally, in autonomous navigation, orientation and angular rate sensors are used. Inertial Measurement Unit (IMU) and odometer sensors provide information about the vehicle's internal information related to proprioceptive sensors. IMUs combine accelerometers, gyroscopes, and magnetometers to qualify slippages, lateral movements of the vehicle, while odometers give information about odometry [17], [18].

G. Global Navigation Satellite System

GNSS provides accurate 3D object positions by a global satellite system and the receiver (i.e., GPS, Galileo, and GLONASS). GPS keeps the vehicle on its intended route with an accuracy of 30 centimeters. With the help of GNSS, other systems are used to determine the absolute position [13], [17], [18].

H. Data Fuse

After the data from all sensors is observed, it has to be fused. Data fusion aims to improve the measurement of different data sensor sources and has great importance in disparate data. Sensorial fusion applied to the measurement of redundant data reduces the uncertainty of the measurement, increases the accuracy, and improves the system's integrity, improving fault tolerance [18].

III. Control Algorithm and Sensor's Placement

As the central RCTR controller, Raspberry Pi 4 with 4 GB of RAM single-board computer is used, it has appropriate dimensions and performance for the first stage of navigation system development.

A preliminary selection for the RTCR was made, considering the robot's main tasks and functionality. The RTCR needs to have the capability to detect and overcome the static obstacle in front of the robot autonomously by changing the configuration on track links. It is possible to define two types of static obstacles, which are concave and convex. To concave obstacles, potholes, ditches, and clefts can be related. In case of concave obstacles, the track links are locked in a neutral position that allows the robot to overcome them. Convex obstacles can be represented with stairs, fallen trees, the rubble of collapsed buildings, rocks, and many other urban landscape forms. So, both types of obstacles shave to detected.

Two ultrasonic ranging modules, HC - SR04 with a sensing range of 2 – 400 cm, are used for obstacle detection. The modules include an ultrasonic transmitter, receiver, and control circuit. HC-SR04 sensors are attached to the panel located in the front part of the RTCT and have different measuring directions. The first one is to measure the distance to the ground to find concave obstacles if observed with a camera (down-looking). The second one detects the objects in front of the robot and helps estimate the camera image's depth (forward-looking). For object detection, the Raspberry Pi Camera Module V2 with an 8-megapixel module was selected as it has high enough resolution to observe needed information from the environment and easy integration with the central computer. Possible sensor placement on the track robot is shown in Fig 2.

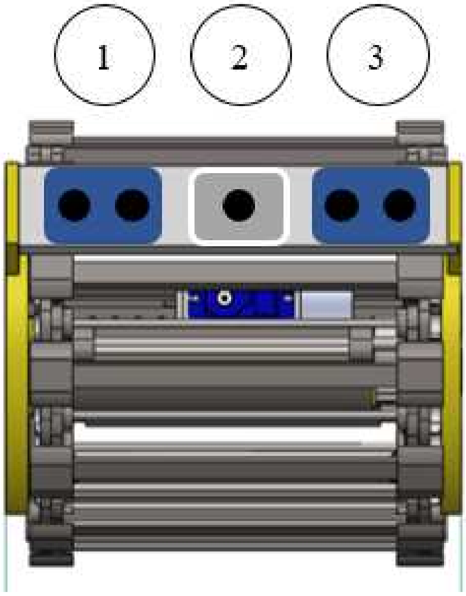

Fig. 2. Possible sensors placement on RTCT: 1) forward-looking ultrasonic sensor, 2) Raspberry camera, 3) down-looking ultrasonic sensor.

The Robot Operating System (ROS) combined with the Gazebo simulator is used to get quicker hands-on experience with image processing. ROS has broad functionality and easy-to-use modular architecture that makes the development process more manageable. The control algorithm is important for quick dataset generation in the simulator. Using simulation and modeling creates tools for analysis, acquisition, and training of the robot.

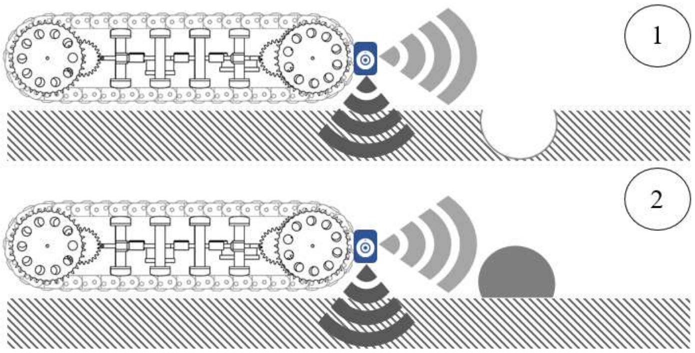

Fig. 3. RTCT driving scenarios: 1) concave obstacle, 2) convex obstacle.

In order to tackle obstacles, the preliminary algorithm was developed. Two main driving scenarios were considered, as shown in Fig. 3. It takes into consideration the advantages of both camera and ultrasonic sensors and integrates them for better performance. As object recognition does not always perform well, it is better to use ultrasonic sensors to understand the exact type of obstacle. The developed algorithm is shown in Fig. 4.

The obstacle detection code module has two main tasks. The first one is to define the type of obstacle in front of the RTCR in case it appears. The second task is to provide the robot with a set of instructions to overcome the obstacle.

Firstly, the robot finds the new path and moves in a selected direction. The camera continuously monitors the presence of obstacles on the robot's path. In case some object is detected, the algorithm defines a concave or convex surface with the forward-looking sensor. There are two possible obstacle types: concave surface (pothole) and concave surfaces (body).

In the case of concave surfaces, the robot moves to the edge of the pothole and measures the length (l_{ct}) of the pothole by application of the triangulation principles. As input data for the calculations, the algorithm uses camera input data. The maximal length for the pothole is defined as l_{max}. After comparing maximal and measured values, the algorithm decides if the robot can overcome the obstacle. If the pothole is too large, the robot starts searching for a new path. Otherwise, it overcomes the obstacle and continues navigation.

In the case of convex surfaces, the algorithm measures the height of the objects h_{ct} with the camera and decides if that is possible to climb on it. If the object height is within the range of maximal height h_{max}, the algorithm calculates the configuration of the track end launches climbing process. If it is not possible to overcome the obstacle, the robot starts looking for a new path.

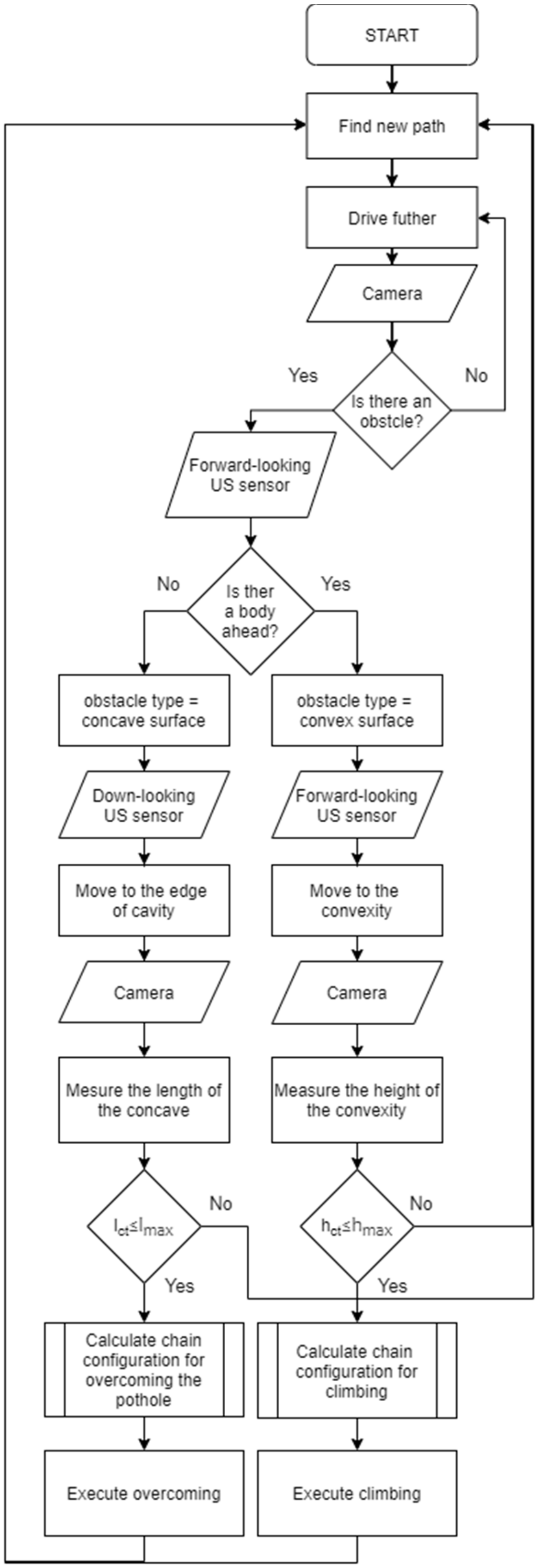
START
Find new path
Drive futher
Camera
Yes
Is there an obstcle?
No
Forward-looking US sensor
No
Is ther a body ahead?
Yes
obstacle type = concave surface
obstacle type = convex surface
Down-looking US sensor
Forward-looking US sensor
Move to the edge of cavity
Move to the convexity
Camera
Camera
Mesure the length of the concave
Measure the height of the convexity
lct≤lmax
No
Yes
hct≤hmax
No
Yes
Calculate chain configuration for overcoming the pothole
Calculate chain configuration for climbing
Execute overcoming
Execute climbing

Fig. 4. Obstacle overcoming algorithm of track robot.

Gazebo simulator has multiple tools for imitating different terrain conditions. The context of sensor data analysis allows generating multiple datasets, which are essential for object recognition training. The opportunity to change light source parameters and object position accelerates the feedback loop for the preliminary tests of the algorithm. Fig 5. represents the example of terrain conditions that can be simulated.

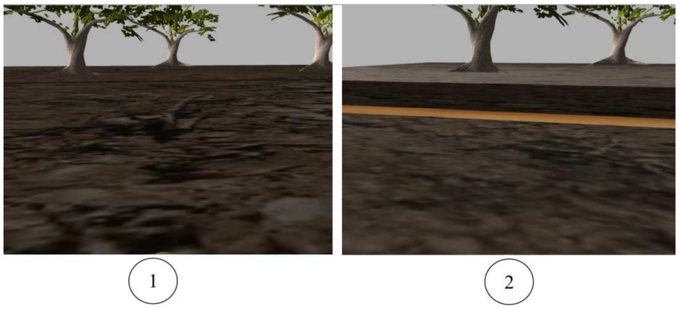

Fig. 5. Gazebo simulator: 1) flat terrain, 2) rough terrain.

Creating the cascade classifiers for the machine vision based object recognition requires gathering a large amount of positive and negative image samples. This can be done by creating the algorithm for data collection.

IV. Conclusion

The sensors' selection is very closely connected to the specific task that a particular vehicle will perform. In this paper, a preliminary set of sensors was chosen to match the requirements shaped by the task structure the best. A combination of hypersonic sensor and Raspberry Pi camera can provide the robot with information about obstacle type and distance.

The following steps are testing the algorithm in real life and the Gazebo simulator, shaping out the best picture processing techniques and creating a more precise algorithm.

The goal is to create a machine vision algorithm that can distinguish obstacles and track them in real-time. It requires implementing additional computer vision or machine intelligence libraries and processing tools such as OpenCV or TensorFlow.

References

[1] Y. Peng, D. Qu, Y. Zhong, S. Xie, J. Luo, and J. Gu, "The Obstacle Detection and Obstacle Avoidance Algorithm Based on 2-D Lidar," *2015 IEEE Int. Conf. Inf. Autom. ICIA*, no. August, pp. 1648–1653, 2015, doi: 10.1109/ICInfA.2015.7279550.

[2] A. N. Catapang and M. Ramos, "Obstacle Detection using a 2D LIDAR System for an Autonomous Vehicle," *6th IEEE Int. Conf. Control Syst. Comput. Eng. ICCSCE*, pp. 441–445, 2016, doi: 10.1109/ICCSCE.2016.7893614.

[3] Z. Rozsa and T. Sziranyi, "Obstacle Prediction for Automated Guided Vehicles Based on Point Clouds Measured by a Tilted LIDAR Sensor," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 8, pp. 2708–2720, 2018, doi: 10.1109/TITS.2018.2790264.

[4] K. Kudelina, T. Vaimann, B. Asad, A. Rassõlkin, A. Kallaste, and G. Demidova, "Trends and Challenges in Intelligent Condition Monitoring of Electrical Machines Using Machine Learning," *Appl. Sci.*, vol. 11, no. 6, 2021.

[5] S. Ramos, S. Gehrig, P. Pinggera, U. Franke, and C. Rother,

"Detecting Unexpected Obstacles for Self-Driving Cars: Fusing Deep Learning and Geometric Modeling," *IEEE Intell. Veh. Symp.*, pp. 1025–1032, 2017, doi: 10.1109/IVS.2017.7995849.

[6] V. D. Nguyen, H. Van Nguyen, D. T. Tran, S. J. Lee, and J. W. Jeon, "Learning Framework for Robust Obstacle Detection, Recognition, and Tracking," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1633–1646, 2017, doi: 10.1109/TITS.2016.2614818.

[7] A. Marjovi, L. Marques, and J. Penders, "Guardians robot swarm exploration and firefighter assistance," *Work. NRS IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, no. January 2015, 2009.

[8] H. Yang, X. Fan, P. Shi, and C. Hua, "Nonlinear Control for Tracking and Obstacle Avoidance of a Wheeled Mobile Robot with Nonholonomic Constraint," *IEEE Trans. Control Syst. Technol.*, vol. 24, no. 2, pp. 741–746, 2016, doi: 10.1109/TCST.2015.2457877.

[9] A. I. Mourikis, N. Trawny, S. I. Roumeliotis, D. M. Helmick, and L. Matthies, "Autonomous stair climbing for tracked vehicles," *Int. J. Rob. Res.*, vol. 26, no. 7, pp. 737–758, Jul. 2007, doi: 10.1177/0278364907080423.

[10] J. Liu, Y. Wang, S. Ma, and B. Li, "Analysis of stairs-climbing ability for a tracked reconfigurable modular robot," *Proc. 2005 IEEE Int. Work. Safety, Secure. Rescue Robot.*, pp. 36–41, 2005, doi: 10.1109/SSRR.2005.1501238.

[11] T. Kislassi and D. Zarrouk, "A Minimally Actuated Reconfigurable Continuous Track Robot," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 652–659, Apr. 2020, doi: 10.1109/LRA.2019.2959237.

[12] D. Valme, K. Kudelina, D. Belolipetskaja, A. Rassõlkin, T. Vaimann, and A. Kallaste, "Generative Design in Weight Optimization of Reconfigurable Continuous Track Robot," *28th Int. Work. Electr. Drives Improv. Reliab. Electr. Drives*, 2021.

[13] J. Ondruš, E. Kolla, P. Vertaľ, and Ž. Šarić, "How Do Autonomous Cars Work?," *Transp. Res. Procedia*, vol. 44, no. 2019, pp. 226–233, 2020, doi: 10.1016/j.trpro.2020.02.049.

[14] SAE, "J3016B Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles," *SAE Int.*, p. 35, 2018, doi: https://doi.org/10.4271/J3016_201806.

[15] N. Ben Romdhane, M. Hammami, and H. Ben-Abdallah, "A generic obstacle detection method for collision avoidance," *IEEE Intell. Veh. Symp. Proc.*, no. Iv, pp. 491–496, 2011, doi: 10.1109/IVS.2011.5940503.

[16] K. Tadakuma *et al.*, "Mechanical design of the wheel-leg hybrid mobile robot to realize a large wheel diameter," in *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 2010, pp. 3358–3365, doi: 10.1109/IROS.2010.5651912.

[17] D. Feng *et al.*, "Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1341–1360, 2021, doi: 10.1109/TITS.2020.2972974.

[18] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A systematic review of perception system and simulators for autonomous vehicles research," *Sensors (Switzerland)*, vol. 19, no. 3, 2019, doi: 10.3390/s19030648.

[19] Y. Singh and L. Kaur, "Obstacle Detection Techniques in Outdoor Environment: Process, Study and Analysis," *Int. J. Image, Graph. Signal Process.*, vol. 9, no. 5, pp. 35–53, 2017, doi: 10.5815/ijigsp.2017.05.05.

[20] J. Behmann *et al.*, "Specim IQ: Evaluation of a new, miniaturized handheld hyperspectral camera and its application for plant phenotyping and disease detection," *Sensors (Switzerland)*, vol. 18, no. 2, 2018, doi: 10.3390/s18020441.

[21] D. C. Liyanage, R. Hudjakov, and M. Tamre, "Hyperspectral imaging methods improve RGB image semantic segmentation of unstructured terrains," *15th Int. Conf. Mechatron. Syst. Mater. MSM 2020*, 2020, doi: 10.1109/MSM49833.2020.9201738.

[22] R. Pi, "Raspberry Pi 4 Computer Model B," *Raspberrypi.Org*, no. May, pp. 1–6, 2020.