İhsan Doğramacı Bilkent University
Electrical and Electronics Engineering Department
EEE 493 - Industrial Design Project
Committee Meeting 3 Report
May 2020

Project Title: Accompanying Humans and Achieving Designated Tasks with Autonomous Mobile Robots using Swarm Intelligence

Group Members: Arda Yüksel, Bilgehan Başpınar, Cevahir Köprülü, Mert Acar, Oğuz Altan

Academic Advisor: Dr. Aykut Koç

Company Advisors: Dr. Bilge Kaan Görür & Muhammed Yüksel

Teaching Assistant: Emir Ceyani

Company Name: ROKETSAN A.Ş.

Company Info: ROKETSAN is a major Turkish weapons manufacturer and defense contractor based in the central Anatolian province of Ankara, incorporated in 1988 by Turkey's Defense Industry Executive Committee (SSİK) in order to establish the nation's industrial base on rocket technology. ROKETSAN is best known for its vast range of unguided rockets as well as laser and infrared guided missiles such as Cirit and UMTAS. The company also produces subsystems for Stinger and Rapier missiles and provides technology and engineering solutions for other integrated civilian and military platforms [1].

Contents

1 Project Summary **4**

2 Motivation and Novelty **4**

3 Design and Performance Requirements **6**
3.1 Functional Requirements 6
3.2 Non-Functional Requirements/Constraints 6

4 Big Picture **7**
4.1 Previous Big Picture (Before the Outbreak) 7
4.2 Current Big Picture (After the Outbreak) 8

5 Methods and Implementation Details **9**
5.1 Work Breakdown Structure and Project Plan 9
5.1.1 Mechanical Setup of Robot 9
5.1.1.1 Land Robot Chassis 10
5.1.1.2 Batteries and Chargers 10
5.1.1.3 Motor Drivers 10
5.1.2 Object Tracking and Avoidance 10
5.1.2.1 YOLO 10
5.1.2.2 LIDAR 11
5.1.2.3 Path Planning 11
5.1.3 Simulations and Real-Life Testing 11
5.1.3.1 Gazebo 11
5.1.3.2 Control with ROS 12
5.1.3.3 Modelling LIDAR 12
5.1.3.4 Beacon 12
5.1.3.5 Leg Tracker 12
5.1.4 Swarm Behaviour and Intelligence 12
5.2 Methods and Progress 14
5.2.1 Mechanical Setup of Robot 14
5.2.1.1 Land Robot Chassis 14
5.2.1.2 Batteries and Chargers 15
5.2.1.3 Motor Drivers 16
5.2.2 Object Tracking and Avoidance 17
5.2.2.1 Target Detection and Identification using YOLO Network 17
5.2.2.2 LIDAR 20
5.2.2.3 Path Planning 24
5.2.3 Simulations and Real Life Testing 29
5.2.3.1 Gazebo 29
5.2.3.2 Control with ROS 33

5.2.3.3 Modelling Lidar . . . 34
5.2.3.4 Beacon . . . 36
5.2.3.5 Leg Tracker . . . 38
5.2.4 Swarm Intelligence . . . 40

6 Online CM3 Presentation and Critics **41**

7 Equipment List **43**

List of Figures

1 Big Picture . . . 7
2 Big Picture . . . 8
3 Work Breakdown Structure of The Project . . . 9
4 Gantt Chart of The Project . . . 13
5 Dagu Wild Thumper 6WD All-Terrain Chassis, 34:1 . . . 14
6 2s 5200 mah 30C LiPo Battery 7.4V . . . 15
7 BTS7960B 40A Motor Driver Module . . . 16
8 Bounding boxes with dimension priors and location predictions [16] . . . 18
9 Tiny-YOLO Architecture [17] . . . 18
10 Tiny-YOLO Testing . . . 19
11 Tiny-YOLO Real-Time Test . . . 19
12 The structure of RPLIDAR A2M8 [18] . . . 20
13 The structure of the data given by RPLIDAR A2M8 [18] . . . 21
14 Working principle of Lidar [19] . . . 21
15 The debugging GUI of RPLIDAR [18] . . . 22
16 Lidar output when the hand takes a horizontal pose . . . 23
17 Lidar output when the hand takes a normal pose . . . 23
18 A high-level view of the move_base node and its interaction with other components [23] . . . 25
19 A costmap example [24] . . . 25
20 Closed-Loop TEB Control [25] . . . 26
21 A snapshot of a TEB planning scenario [26] . . . 27
22 A snapshot of a TEB demo from CM2 Presentation . . . 27
23 Gazebo Simulation Software [27] . . . 30
24 2 Wheeled Robot with Hokuyo Laser[28] . . . 31
25 DAGU Wild Thumper 4WD on Gazebo . . . 31
26 Nodes of the Simulation . . . 32
27 Simulation Launch with RVIZ and GAZEBO . . . 32
28 CM3 Presentation ROS Nodes . . . 33
29 Lidar simulation result before CM I using a simpler Lidar Gazebo model . . . 35
30 The RPLIDAR data flowing on the left, RViz visualization on the right which shows the obstacles in the simulation world . . . 35

31 HW v4.9 Starter Kit: 5 x HW v4.9 Beacon & 1 x Modem [30] 36
32 Test on Dashboard with 3 HW v4.9 Beacons as Stationary and 1 as Hedgehog . . . 37
33 Gazebo Simulation World designed for Leg Tracker Test 38
34 Leg Tracker Test World's Identified Legs in RVIZ 39
35 Online CM3 Simulation Demo 43

List of Tables

1 BoM of the Project 43

1 Project Summary

This project aims to to implement mobile robots that can participate in swarm behaviour based activities, in collaboration with ROKETSAN. These autonomous mobile land robots are intended to track a moving object, which is human in our case, and adjust their motion parameters such as speed and direction, with respect to the motion of the tracked object. Additional features of these robots are the ability to avoid nearby obstacles by detecting and dodging them, and to smoothly move on the terrain. The tracking part is to be performed by using Computer Vision and object tracking algorithms, using the real – time image data captured by an action camera, and the control of the robot is to be first simulated on AI – robotics simulation software Gazebo and afterwards to be implemented on the framework Robot Operation System (ROS). Lastly, for the obstacle avoidance concern, a Laser Imaging Detection and Ranging (LIDAR) sensor is to be used to detect the obstacles nearby the robot [2] and inform the control system on the robot. The software components of this project are running on AI-specialized mini computer Nvidia Jetson Nano, the brain of the robot. For the mechanical part of the robot, Dagu Wild Thumper 6WD All-Terrain Chassis [3] is to be used with specified motor drivers and battery, to provide optimal performance. The validation of the proposed implementation is to be tested on various simulation programs such as aforementioned software Gazebo. Computer Vision algorithms are first to be tested on team members' personal computers, before implementing them on Jetson Nano [4] and the mechanical structure is to be tested on the laboratory environment. Due to Covid-19 outbreak, the aforementioned works on hardware implementations and systems that are expected to be worked on in laboratory are cancelled.

2 Motivation and Novelty

These autonomous mobile land robots are intended to be used for various military services of Türk Silahlı Kuvvetleri (TSK), or Turkish Armed Forces, in English. ROKETSAN is trying to create and implement autonomous systems that can be used for several tasks such as accompanying soldiers in the field such as tracking and scouting, transporting equipment and following troops, search and rescue operations and gathering military intelligence. Carrying out these tasks with military personnel risks casualties and is inefficient for economic reasons. This issue has significant importance for national armies in the world and various military and defense companies around the globe try to solve this issue by developing different technologies [5] that aims to decrease the aforementioned costs. One of the solutions is developing autonomous mobile robots [6] that can be used to yield lower costs and an effective alternative for human life and this solution is what ROKETSAN intends to obtain in collaboration with our project team. To increase the functionality and enriching these proposed autonomous systems, we also work on implementing swarm behavior and joint intelligence that significantly boost the robustness of the performance and increase the feasibility of accomplishing military tasks, stated above. As a consequent step of successfully completing our project, ROKETSAN plans to use these autonomous mobile land robots in the military field as a stand-alone project, meaning that these explained autonomous systems are to be sustainable by themselves without needing any human assistance. As swarm intelligence is one of the main parts

of our project, ROKETSAN may increase the number of robots in the swarm and add additional features and capabilities to the robots which can be useful in the military service.

The target end-user of our product is the military personnel of TSK. According to our scenario, these autonomous robot swarms have several duties in the field, such as following the soldier that the swarm is assigned to with avoiding obstacles, as well as calculating and tracking its path autonomously, carrying military equipment, understanding military personnel's specific directions such as stop, go right and left and proceed. Therefore, the military service, especially soldiers in the field will use and command these swarms. As it is told by our ROKETSAN advisors, the cost of the robots that we work on in our project is affordable and cost efficient, specifically the hardware components and equipment are financially efficient and we use free open source software technologies. Therefore, as ROKETSAN produces the defense industry technologies for TSK [7], it can also be stated that the only cost of this project is the total cost of used equipment. The robots in this project are autonomous and therefore, there is no need for any technical instructions for using it in the field, any soldier can use it without having any technical background. A situation where the military personnel with technical background may be needed is the case of any breakdown and malfunctioning of the systems. As the robots in our projects consist of separate modules, the repair is not complex and finding the cause of malfunctioning is not time consuming.

As we have shown and explained in our project presentations, there are several projects which intend to develop similar technologies to our, such as human identifying drones [8] and drone swarms moving in pre-determined trajectory to accomplish various tasks [9]. Comparing them to our project, firstly it can be stated that our project includes many stages and parts, while these similar projects perform only a few methods that we want to have in our project. In other words, these similar projects intend to develop one or two of the functionalities that we intend to have, they do not contain all of the techniques and algorithms that we aim to have on our robots. Therefore, the cost of our systems may be different from those of these projects. An example project, called Real-time 3D Human Tracking for Mobile Robots with Multisensors, developed in The University of Technology, Sydney, Australia, aims to perform similar tasks like we do [10]. Comparing speed and maximum weight carrying capabilities, the robot mentioned in this paper can speed up to 2.88 km/h and carry up to 2 kg, meanwhile our robot can speed up to 7 km/h and carry up to 5 kg. Another advantage of our project is that due to its highly complex structure and integrity of different tasks and functionalities, it can accomplish a high variety of tasks, while this may also result in a disadvantage as due to high complexity of our project, the implementation and maintenance is more difficult to handle.

Due to the integrity of many technologies and complex structures, our project can bring many novelties and innovations. For example, the concept of autonomous robots having swarm intelligence is quite innovative in the military technologies in the world. The aforementioned similar products have several problems especially in the implementation of swarm technology, for instance, the importance of avoiding collisions of swarm members arises frequently. In our project, we intend to solve these problems by developing new algorithms and technologies. For the patent concerns, there are not any patented solutions to the stated problems. However, we think that especially the swarm technology part in our project may be a powerful candidate for the patent, as it brings many innovations and novelties.

3 Design and Performance Requirements

3.1 Functional Requirements

In our previous report hardware related functional requirements are stated as follows:

1. 5 km/h is required to follow walking or running soldier. Specifications for these units state the maximum speed as 7 km/h. However, under load and on terrain conditions, we require the robot not to drop below 5 km/h.

2. Mechanical units are expected to carry 4-5 kg of load.

3. Our model is required to detect the target (marked soldier) from no further than 15 meters with a minimum confidence of 45 percent.

4. Under full operation the algorithm is expected to run at 12 frames per second.

These requirements are no longer meaningful, since we are expected to complete software related parts of the project during this pandemia period. Software related functional requirements can be stated as follows:

1. As for the simulation part, we require a LIDAR scan rate of 7-9 Hz.

2. Leg Tracker module requires maximum of 5m range in simulations.

3.2 Non-Functional Requirements/Constraints

1. We were planning to make 2 robots for 3323 TL, including the components that ROKETSAN has provided to us. We do not have a financial limitation imposed by the company and the aim is to assemble robot with reasonable budget. However, we are unable to assemble the robots due to current circumstances.

2. Each robot is expected to carry 4-5 kg load. The total weight of the chassis and mechanical units shouldn't exceed 3 kg. Dagu Wild Thumper 6WD, the robot chassis, weights 2.5 kg and we plan to mount other peripherals and components, therefore the total weight of the robot without any additional load is expected to be about 3 kg. However, we are unable to assemble those components due to current circumstances.

3. Mobile Units are expected to draw a maximum of 18 A of current under full load and no more than 6.6 A during jump-start. These units are expected to operate for approximately 25 - 30 mins with one charge of LiPo Battery. However, the project is being conducted in simulation environment due to current circumstances.

4. Robots were planned to be designed to help armed forces by following them on terrain operations, therefore a robot chassis with tires that are suitable for terrain conditions picked for such missions, which cannot be used anymore due to current circumstances.

5. Our plan A was to use a LiPo battery as the power supply of the robots, yet, LiPo batteries can be somewhat dangerous to handle sometimes. Overcharging a LiPo battery can even cause fire, thus one needs to be cautious working on it.

6. Modem of the HW v4.9 Starter hardware set has 100 m range for radio signal coverage

7. HW v4.9 Beacons need to be places at least 2 m apart from each other

8. HW v4.9 Beacons should be facing each other in order to receive and transmit ultrasonic wave for localization

9. If HW v4.9 Beacons are not placed at the same height from the ground, their positions on the z-axis should be entered to the map.

10. We do not have any health constraints or requirements.

11. As mentioned earlier, our main objective is to help armed forces by following them on terrain operations. Therefore, if this project is to be used in real life, its design would be affected by global and political changes, which also would change the defending style.

12. Our project will be compatible with Standard for Connected, Automated and Intelligent Vehicles: Overview and Architecture, IEEE Std P2040.1 [11] and P2040.2 [12] in terms of automation.

4 Big Picture

4.1 Previous Big Picture (Before the Outbreak)

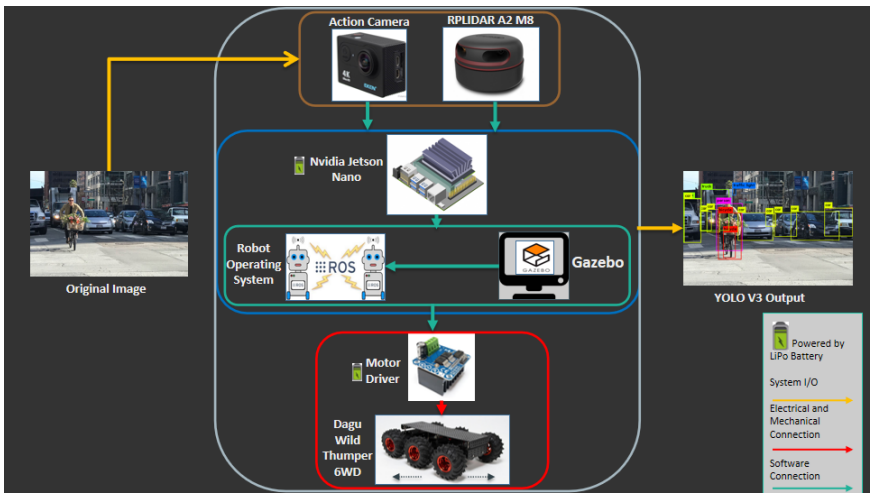

Figure 1: Big Picture

The system senses the world using action camera and LIDAR. Camera provides real-time video footage to the system and LIDAR detects the surrounding objects in the field. These data are processed in the Robot Operation System (ROS) working on Nvidia Jetson Nano computer. YOLO real-time object detection system software takes the input video footage taken with action camera and detects the human in the video frame. Meanwhile, the point cloud data taken with LIDAR is converted to useful data in ROS and the nearby obstacles are detected. Finally, the control algorithms working on ROS leads the robot and perform motion operations, for instance tracking the computed path and avoiding obstacles. These control algorithms provide digital input data for the motor drivers that controls motors of the robot. The power of the system is provided by the LiPo Battery. For the simulation purposes, Gazebo Simulation Software is used, running on our computers. In this simulations, the virtual model of the robot in our project is tested in a virtual world with several obstacles. The developed technologies and algorithms is first tested in these simulations and possible malfunctions are observed and debugged. Consequently, these technologies are deployed to real robots controlled by ROS. Due to Covid19 outbreak, this Big Picture has been changed. The new one can be found in the following subsection.

4.2 Current Big Picture (After the Outbreak)

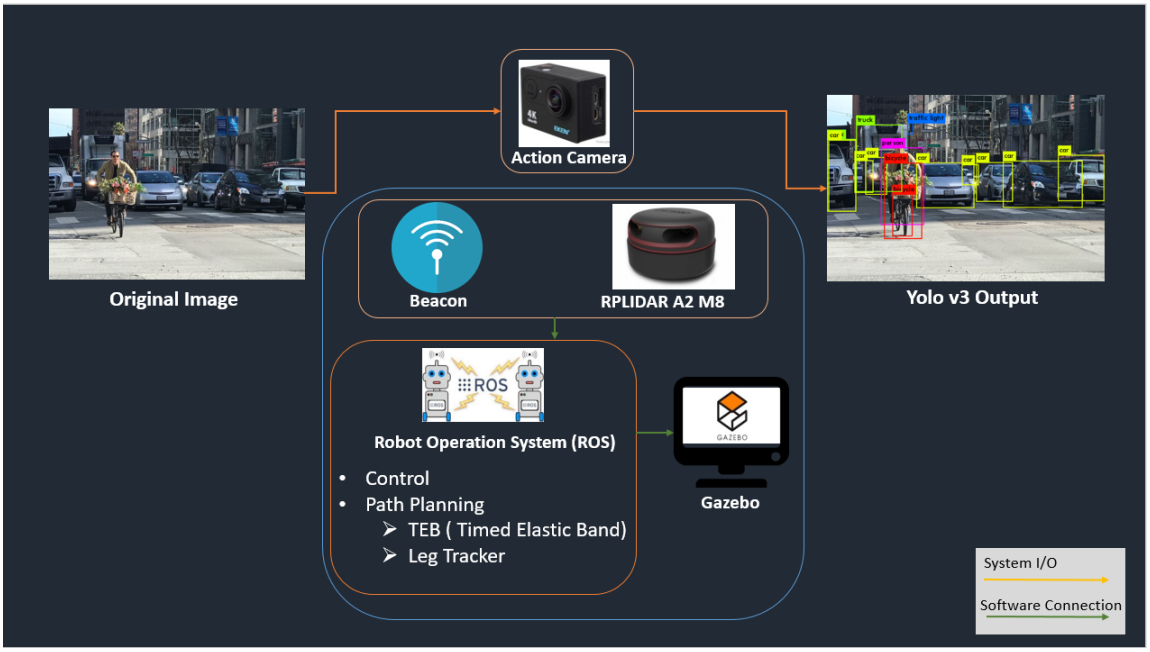

Figure 2: Big Picture

Due to Corona Pandemic, some of the parts that are required to be worked on in laboratory conditions are ommitted and taken out from the Big Picture. These parts are chassis of robot, motor drivers, motors, Lipo and Jetson Nano. Although our goals on these hardware components were achieved by the deadlines, these components left in the laboratory and they are inaccessible. New Big Picture contains computer vision software components such as object classification with

YoLo and obstacle sensing with LIDAR. Using the leg detection algorithm and LIDAR obstacle sensing, in the simulated world in Gazebo, the simulation model of our robot can detect the human in the surrounding environment and follow the human, using control algorithms implemented on Gazebo and ROS. In addition, Beacon data is used for odometry input of the robot's controller and timed elastic band (TEB). Controller and path planning parts are done on ROS with simulations on Gazebo.

5 Methods and Implementation Details

5.1 Work Breakdown Structure and Project Plan

This project now includes three different milestones. Each milestone will be broken down to be explained in detail and given details about the time and people allocation. These allocations will be demonstrated using a Gantt chart integrate the progress better.

Due to Corona Pandemic, last milestone of the CM2 is omitted which was Swarm Behaviour and Intelligence. Under that milestone Pattern Formation, Human-Swarm Interaction and Task Allocation parts were listed. However the Simulations and Real Life Testing milestone includes two new packages named as Beacon and Leg Tracker.

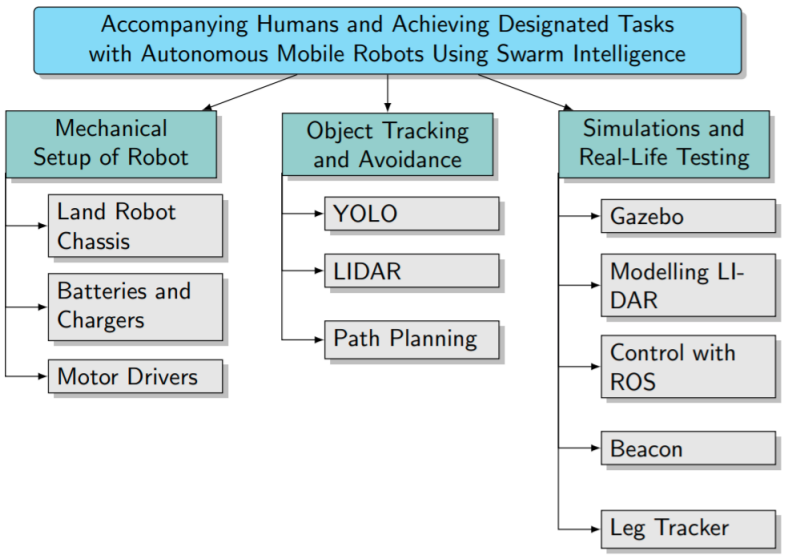

Figure 3: Work Breakdown Structure of The Project

5.1.1 Mechanical Setup of Robot

Mechanical setup of robot comprises three sub-tasks: Land Robot Chassis Assembly, Batteries and Chargers, Motor Drivers Selection and Testing. The plan can be seen in the Gantt Chart provided in Figure 4. All the subtasks are completed by the beginning of the second semester.

5.1.1.1 Land Robot Chassis

Land Robot Chasis subtask includes the assembly of the robot setup by combining the robot parts. The model for this part is Dagu Wild Thumper 6WD 34:1. The main criterion is the robot's ability to handle the terrain conditions as the main scenario demands. The selected model satisfies this condition. This sub-task is completed before the 1st of January, 2020. The Dagu Model is assembled. Therefore, criteria of success which was the establishment of working model is accomplished. This milestone was under Oğuz Altan's responsibility.

5.1.1.2 Batteries and Chargers

In this part, using the model selected for the first sub-task under Mechanical Setup of the Robot, the necessary voltage and current level for each component is calculated. For the calculations datasheet of the Dagu Wild Thumper is used as basis. The assembled robot is tested for calculated values and equipment compatible with the measured values are selected. The purchase of the batteries are completed by the beginning of the second semester.

This sub-task was under Oğuz Altan's responsibility. Success criteria is satisfied which is confirming the desired power levels through the purchased batteries and chargers.

5.1.1.3 Motor Drivers

In this part, the motor drivers for the selected Chassis is aimed to be selected. In order to reach a conclusion data sheet of the selected chassis is examined. Then, the robot is tested in the lab for both Batteries and Chargers and Motor Drivers sub-tasks. After the testing procedure necessary components are identified for the purchase. The order of the components are expected be given. This sub-task is completed in early January.

This sub-task was under Oğuz Altan's responsibility. Success criteria for this sub-task is moving the robot through the usage of purchased motor drivers.

5.1.2 Object Tracking and Avoidance

Under this milestone, YOLO, LIDAR and Path Planning are given as sub categories. Path Planning algorithm was being tested during the second committee meeting. However, now it is officially Timed Elastic Band(TEB) algorithm. All the subtasks in this part is completed in the first semester as it can be seen from 4.

5.1.2.1 YOLO

The first sub-task concerns about getting YOLO environment ready which comes with a simple installation process. However, using default installation process, it does not process real time data. Therefore one needs to compile it from source using Nvidia's CUDA and OpenCV which is also built from source with CUDA. Once these steps are achieved then a webcam feed can be an input to YOLO network and real time object detection can be used.

This subsection includes filtering process to identify selected personnel. Currently using the location of the target provided by YOLO, the angle for rotation is calculated.

The criteria of success was achieving 12 FPS in Computer Vision applications with 45 percent success rate in the range of 15 meters. This criteria has been tested both for live feed and sample video taken from the Action Camera which can be seen in the equipment list. This sub-task is completed before the 15th of December. It was assigned to Mert Acar.

5.1.2.2 LIDAR

The second sub-task consists of selecting the LIDAR and visualizing the real world data using the selected LIDAR. The LIDAR was selected before Committee 1 as RPLIDAR A2M8. After obtaining the model from ROKETSAN, using the provided SDK, real world data is analyzed through the model. The obtained data is tested and compared to expected outcomes. Visualization is achieved.

The success criteria was providing the LIDAR output that can be used for path planning case. After comparing the obtained data to the expected output accurate representation of the real world is attained. This milestone was expected to be completed before 1st of January. This aim is achieved by Bilgehan Başpınar.

5.1.2.3 Path Planning

This milestone includes usage and optimization Timed Elastic Band algorithm. Path Planning is expected to sustain robot to follow the target without interfering obstacles in the path. This part integrates the LIDAR and YOLO outcomes. It integrates Timed Elastic Band algorithm for that purpose.

This subtask is assigned to Arda Yüksel, Cevahir Köprülü and Bilgehan Başpınar due to increase in complexity as it was mentioned in CM2. The simulations are commenced in sample robots before CM2 and integrated into DAGU Simulations during the second semester. Stable version on sample robot was the success criteria which is achieved by the end of January.

5.1.3 Simulations and Real-Life Testing

The simulation and real-life testing of the project will be broken down to five sub-tasks which are Gazebo, Control with ROS and Modelling Lidar, Beacon and Leg Tracker. Compared to CM2, Beacon and and Leg Tracker parts are added. Real-life Testing aim for the most of components are omitted due to Corona Virus outbreak. Demo is commenced in simulated environments.

5.1.3.1 Gazebo

In this part the installation and basic comprehension of the ROS and Gazebo was aimed initially. This milestone is expanded into demonstration of the DAGU Wild Thumper with the selected LIDAR as the products are decided. After the selection of the equipment, Gazebo environment are prepared for the testing of possible simulations.

This sub-task was given to Arda Yüksel before Committee Meeting 1. After the necessity of simulations in Path Planning algorithms, this milestone is now correlated to Path Planning Simulations in Gazebo. Thus, its timeline is expanded into the beginning of 1st of February, 2020 in CM2. The success criteria was running sample simulations in Gazebo for Wild Thumper and

provide feedback for the applied algorithms. Now this task is under Arda Yüksel and Cevahir Köprülü. Gazebo simulations for the necessary work packages like LIDAR and Path Planning are accomplished before the deadline.

5.1.3.2 Control with ROS

The aim of this sub-task is to simulate and smoothen the behaviour of robot. Due to Corona Pandemic, its focus is turned into control in the simulation. The deadline is extended to second half of April, 2020 and completed before the CM3 presentations. The success criteria is obtaining stable control over the designed DAGU Wild Thumper Gazebo Simulation. This sub-task is assigned to Oğuz Altan.

5.1.3.3 Modelling LIDAR

This subtask aims to obtain the LIDAR data of the simulated robot and visualize the outputs in Point Cloud format through RVIZ. In addition to that, using programs to obtain the LIDAR outputs via ROS was expected to be completed.

Bilgehan Başpınar was responsible for Modelling LIDAR and she finished her duties before the 1st January, 2020 as expected.

5.1.3.4 Beacon

After first semester, ROKETSAN advised us to integrate beacon system into current setup in order to gain control over the localization process. The items are provided by ROKETSAN. In this subtask the aim is to identify location of the robot and visualize it in a given environment.

This sub-task is given to Cevahir Köprülü since he had the access to hardware during the pandemic. The deadline was selected as the second half of the April, 2020. This task is completed and demonstrated in the CM3 presentation.

5.1.3.5 Leg Tracker

This subtask aimed towards identifying humans within LIDAR's range using the point cloud data provided by RPLIDAR. This part consists of the simulation of Leg Tracking and selecting human goals for the TEB Local planner. Due to Corona virus, real world demonstration is omitted.

The deadline was the second half of April, 2020 and it is accomplished by the given deadline. Leg Tracker demo as well as its integration to the overall simulation is provided in CM3 presentation. It was assigned to Arda Yüksel and Mert Acar.

5.1.4 Swarm Behaviour and Intelligence

This milestones and its relative subtasks are omitted due to technical difficulties issued by the Corona Virus outbreak.

It was aimed at achieving swarm behaviour with multiple robots. This milestone is divided into three sub-tasks which are Pattern Formation, Task Allocation and Human-Swarm Interaction.

Project Calendar			2019						2020									
Accompanying humans and achieving designated tasks with autonomous mobile robots using swarm intelligence			10	10	11	11	12	12	1	1	2	2	3	3	4	4	5	5
#	Work package	Members	1	15	1	15	1	15	1	15	1	15	1	15	1	15	1	15
1	**Mechanic Setup of Robot**																	
	Land Robot Chasis	OA	100% COMPLETE															
	Batteries and Chargers	OA			100% COMPLETE													
	Motor Drivers	OA					100% COMPLETE											
2	**Object Tracking and Avoidance**																	
	YOLO	MA	100% COMPLETE															
	LIDAR	BB					100% COMP											
	Path Planning	CK, AY, BB					100% COMPLETE											

Project Calendar			2019						2020									
Accompanying humans and achieving designated tasks with autonomous mobile robots using swarm intelligence			10	10	11	11	12	12	1	1	2	2	3	3	4	4	5	5
#	Work package	Members	1	15	1	15	1	15	1	15	1	15	1	15	1	15	1	15
3	**Simulations and Real Life Testing**																	
	Gazebo	CK, AY			100% COMPLETE													
	Control with ROS	OA, MA, BB, AY			100% COMPLETE													
	Modelling LIDAR	BB			100% COMPLETE													
	Beacon	CK										100% COMPLETE						
	Leg Tracker	AY, MA											100% COMPLETE					
	LEGEND																	
	OA = OĞUZ ALTAN	AY = ARDA YÜKSEL																
	MA = MERT ACAR	CK = CEVAHİR KÖPRÜLÜ																
	BB = BİLGEHAN BAŞPINAR	CORONA OUTBREAK																

Figure 4: Gantt Chart of The Project

5.2 Methods and Progress

5.2.1 Mechanical Setup of Robot

5.2.1.1 Land Robot Chassis

Up to CM1

The mechanical structure and chassis of the robot is discussed and several mechanical designs are proposed, considering specifications and requirements. The robot has to carry at least 4-5 kg load and in addition, move on a terrain with obstacles, without any malfunctioning. the first plan was to construct the robot ourselves, by first making the design, then obtaining the required construction materials and finally build the robot. After discussions with ROKETSAN, the plans have been changed and it is decided to make a market research about a robot chassis that is appropriate for aforementioned mechanical requirements. Dagu Wild Thumper 4WD is selected as the robot chassis.

Between CM1 and CM2

The model has been updated to Dagu Wild Thumper 6WD from 4WD, with the proposal of ROKETSAN.

Figure 5: Dagu Wild Thumper 6WD All-Terrain Chassis, 34:1

The gear ratio is also updated to 34:1. This robot chassis is expected to carry around 7 kg of payload. In addition, it is a differential-drive chassis, meaning that turning is accomplished by driving the motors on the two sides of the platform at different speeds. For instance, steering to right is managed by changing the relative speeds of right three and left three wheels of robot, where the speed of right ones are higher than the left ones.

Between CM2 and CM3

Until the Covid-19 outbreak, the plan was to mount all of the hardware modules on top of the robot chassis and complete the electrical connections. Motor drivers are mounted before the outbreak but connecting LiPo batteries and setting up other hardware modules are currently not feasible.

Future Plans

There is not any future plan for this part, as the mentioned modules are inaccessible.

5.2.1.2 Batteries and Chargers

Up to CM1

Two candidates for battery were discussed, which were LiPo battery and NiMH (Nickel–metal hydride) battery. As LiPo battery is lighter and more robust compared to NiMH battery, LiPo is selected as the battery of the system. The exact product is planned to be selected after having decided the robot chassis, motors and motor drivers.

Between CM1 and CM2

For the powering part, LiPo (Lithium Polymer) battery technology is chosen as the power source for the motors, as these batteries are lightweight and robust. The 7.4V 2s 5200 mAh 30C LiPo battery is planned to be used in the project, it is suitable to the power specifications of the motors and the chosen motor driver. The estimated time of operation of the robot using mentioned LiPo battery is 25-30 minutes. For the time efficiency reasons, it is planned to acquire 2 of these LiPo batteries, so that while one of them is getting charged, the other one can be used to power up the robot. Finally, for powering the Jetson Nano, which consumes 10W and requires 5V/2A, the proposed solution is to use a powerbank, plugged to Jetson Nano via USB cable. The Xiaomi 10000 mAh powerbank , which can provide 5V/2A [13], is appropriate for this aim.

Figure 6: 2s 5200 mah 30C LiPo Battery 7.4V

Between CM2 and CM3

Due to the Covid-19 outbreak, there is not any work done in this period.

Future Plans

There is not any future plan for this part, as the mentioned modules are inaccessible.

5.2.1.3 Motor Drivers

Up to CM1

The motor drivers were planned to be selected after having decided the robot chassis and motors.

Between CM1 and CM2

The three motors on each side of the robot are wired in parallel, therefore only two channels of motor control are required to get this chassis moving. The motors are intended for a maximum nominal operating voltage of 7.2 V (2V minimum), and each has a stall current of 6.6 A and a no-load current of 420 mA at 7.2 V. The motors briefly draw the full stall current when abruptly starting from rest (and nearly twice the stall current when abruptly going from full speed in one direction to full speed in the other). The BTS7960B motor drivers are chosen according to these specifications. These single-channel motor drivers can provide up to 40A and works in the the operating voltage interval 5.5V - 28V [14]. As the motor driver is single channel, it is planned to use 2 of these motor drivers, one for each side of the robot, namely 3 motors wired in parallel. In addition, the mentioned motor driver can be used with Pulse Width Modulation (PWM) technique, and this technique is planned to be used for the motion control of the robot.

Figure 7: BTS7960B 40A Motor Driver Module

For the mechanical setup of the robot, there are possible risks. As this part is about batteries, motors and motor drivers, which are all working on considerably high voltages, currents and amperes, there may arise electrical connection problems and power specification mismatching. For instance, if motor driver provides more current than motors of the robot can handle, it may burn the motors and rend them dysfunctional. Another problem about power mismatching may be the scenario that LiPo battery can not provide enough power, when it is on low charge, to the motor drivers, thus robot does not move at all, despite Jetson Nano works and can send appropriate control commands. These can be solved by limiting the output current of the motor driver and replacing the current LiPo battery with the backup battery, respectively. Another bad scenario

about LiPo batteries is caused by the nature of the battery itself. If the charge of the battery goes below a critical level, depends on the specifications of battery, to work with that battery reduces its lifespan and may even cause explosion [15]. To use LiPo batteries always above the critical charging level is an important practice.

Between CM2 and CM3

From the CM2 until the outbreak, motor drivers are mounted on top of the robot and the electrical connections are completed. Motor drivers are programmed with Jetson Nano to control the motors on the robot, using PWM signals. The wheels on the robot can be controlled and the direction and speed can be adjusted. Due to Covid-19 outbreak, the mentioned hardware parts are inaccessible.

Future Plans

There is not any future plan for this part, as the mentioned modules are inaccessible.

5.2.2 Object Tracking and Avoidance

5.2.2.1 Target Detection and Identification using YOLO Network

Up to CM1

Target tracking consists of two fundamental parts, the 1st one being the detection of the target and the 2nd part being tracking the location of it. To construct a complete method, we begin with proposing the Deep Convolutional Neural Network architecture YOLO V3 as the image classifier/detector. YOLO V3 takes a high resolution image and outputs its predictions about objects that might be on the image, and about the bounding box of the object on the image. This bounding box prediction consists of predicting 4 coordinates (t_x, t_y, t_w, t_h) for each bounding box. Given the offset of the cell (c_x, c_y) and prior on width p_w and p_h, the predictions correspond to:

$$b_x = \sigma(t_x) + c_x \tag{1}$$
$$b_y = \sigma(t_y) + c_y \tag{2}$$
$$b_w = p_w e^{t_w} \tag{3}$$
$$b_h = p_h e^{t_h} \tag{4}$$
$$\tag{5}$$

An example of bounding box prediction is given below in Figure 8

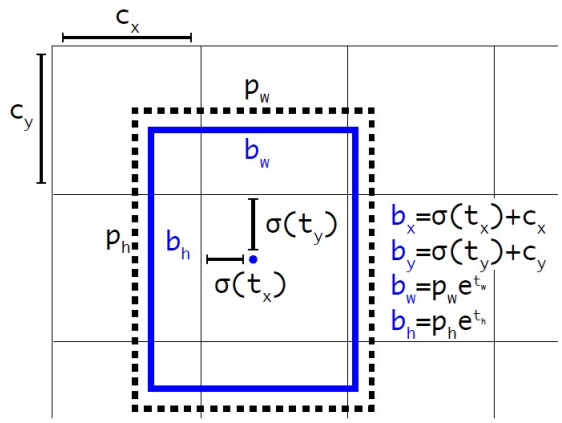

Figure 8: Bounding boxes with dimension priors and location predictions [16]

To get deeper into the logic of class predictions, each box is designed to predict the classes the bounding box may contain, thus it essentially utilizes multilabel classification. This is quite helpful for our project, since although initially our objective is to detect a human-being, one of the complementary targets of the project was to predict various tools that the targeted human carries with him/herself. Considering the availability of multiclass prediction, we could benefit from Transfer Learning in order to classify more customized objects.

Considering the limitations of the main board, Jetson Nano, in order to avoid lagging in real-time detection, we will utilize another YOLO architecture called Tiny-YOLO, whose structure is given below in Figure 9:

layer	filters	size	input		output
0 conv	16	3 x 3 / 1	224 x 224 x 3	->	224 x 224 x 16
1 max		2 x 2 / 2	224 x 224 x 16	->	112 x 112 x 16
2 conv	32	3 x 3 / 1	112 x 112 x 16	->	112 x 112 x 32
3 max		2 x 2 / 2	112 x 112 x 32	->	56 x 56 x 32
4 conv	16	1 x 1 / 1	56 x 56 x 32	->	56 x 56 x 16
5 conv	128	3 x 3 / 1	56 x 56 x 16	->	56 x 56 x 128
6 conv	16	1 x 1 / 1	56 x 56 x 128	->	56 x 56 x 16
7 conv	128	3 x 3 / 1	56 x 56 x 16	->	56 x 56 x 128
8 max		2 x 2 / 2	56 x 56 x 128	->	28 x 28 x 128
9 conv	32	1 x 1 / 1	28 x 28 x 128	->	28 x 28 x 32
10 conv	256	3 x 3 / 1	28 x 28 x 32	->	28 x 28 x 256
11 conv	32	1 x 1 / 1	28 x 28 x 256	->	28 x 28 x 32
12 conv	256	3 x 3 / 1	28 x 28 x 32	->	28 x 28 x 256
13 max		2 x 2 / 2	28 x 28 x 256	->	14 x 14 x 256
14 conv	64	1 x 1 / 1	14 x 14 x 256	->	14 x 14 x 64
15 conv	512	3 x 3 / 1	14 x 14 x 64	->	14 x 14 x 512
16 conv	64	1 x 1 / 1	14 x 14 x 512	->	14 x 14 x 64
17 conv	512	3 x 3 / 1	14 x 14 x 64	->	14 x 14 x 512
18 conv	128	1 x 1 / 1	14 x 14 x 512	->	14 x 14 x 128
19 conv	1000	1 x 1 / 1	14 x 14 x 128	->	14 x 14 x1000
20 avg			14 x 14 x1000	->	1000
21 softmax					1000
22 cost					1000

Figure 9: Tiny-YOLO Architecture [17]

Our first implementation was compiling YOLO on Jetson Nano with CUDA and OPENCV, and then giving the network a generic image of Figure 10:

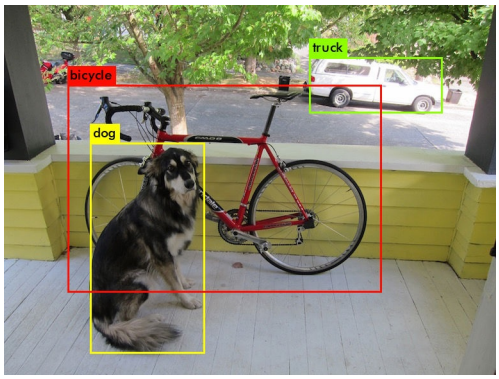

Figure 10: Tiny-YOLO Testing

Following this step, we connected our 4K action camera to Jetson Nano in order to do a real-time testing, and took the following frame given in Figure 11 from the tested live-feed:

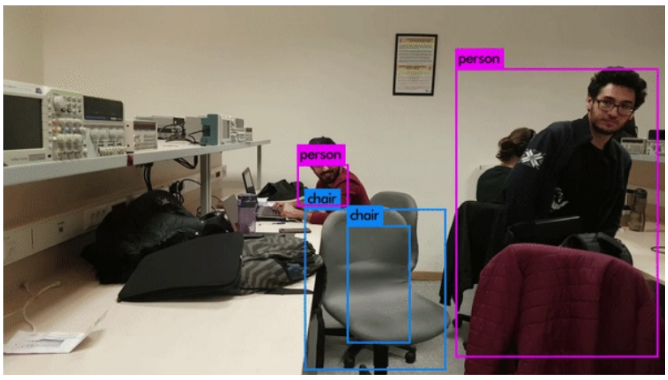

Figure 11: Tiny-YOLO Real-Time Test

As of this testing, we completed the real-time tests of Tiny-YOLO and successfully showed that Jetson Nano can run Tiny-YOLO in 45-50 frame per second. Therefore, our next step will be tracking already detected bounding boxes in a computationally efficient manner.

Up to CM2

In an environment with multiple humans in the field of view of the robot, the master unit will determine the target it will follow with a marker. This procedure will also be done using the robot's camera. The target will be equipped with a yellow-and-black-striped belt that will act as a marker for the robot. In the choice of the marker, the important criteria was it to be easily distinguishable in nature with a simple pattern. The detection of the marker is done by a kernel based line detector with parameters fine tuned to match the gradients of the striped belt. The kernel function is centered around the origin and matches the lines on the belt with a size of 7 x 7.

For scalability, multiple kernel functions with varying sizes like 3 x 3, 5 x 5 and 7 x 7 have planned to be employed however, as the target moves further away from the robot's camera, the smaller kernels did not perform as expected while also damaging the performance heavily since the

smaller kernel convolutions are expensive to compute. Therefore, a 7 x 7 fixed symmetric kernel is employed with the color matching algorithm of the mean-shift method to compensate for the scalable target detection problem. However, the color matching presents itself with calibration and performance fluctuation under different light conditions. In order to aid this problem, a reflector coating will be used to help the robot distinguish the pattern.

Between CM2 and CM3

Once the engine is performing as expected, the output of the algorithm is ported to the ROS framework. This framework provides the simple messaging service between computation nodes. A computation node is a program that controls an aspect of the robot's main framework. The differential drive node that is incorporated in Gazebo simulation software is responsible for driving the motor drivers of the robot and expects the commands from the ROS messaging service. By piping the control output from the algorithm to this messaging service the simulation is connected to the camera module and the simulated unit tracks the person in front of the camera.

Future Plans

We are able to process the data stream from the camera efficiently and adequately. Therefore this work packet is successfully completed and ready to deploy.

5.2.2.2 LIDAR

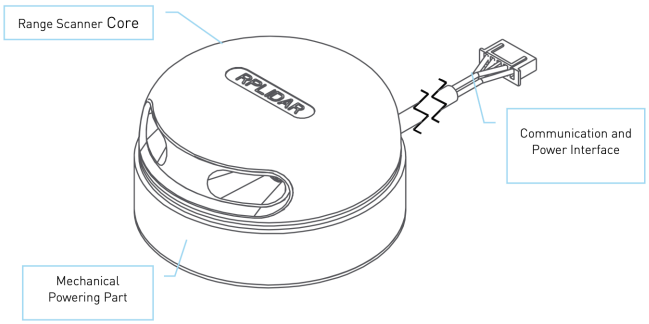

Figure 12: The structure of RPLIDAR A2M8 [18]

For tracking and object avoidance purposes, a Lidar of model RPLIDAR A2M8 is being used. RPLIDAR A2M8 measures the distance varying with the angle by using laser triangulation ranging principle. During every ranging process, the RPLIDAR emits modulated infrared laser signal and the laser signal is then reflected by the object to be detected. The returning signal is then sampled by vision acquisition system in RPLIDAR and the DSP embedded in RPLIDAR starts processing the sample data and outputs distance value and angle value between object and RPLIDAR via communication interface. [18]

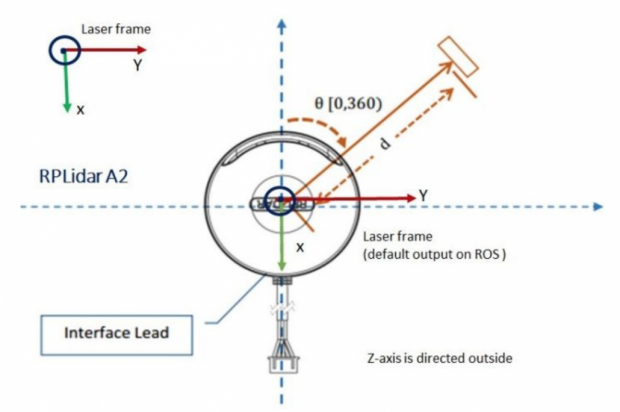

Figure 13: The structure of the data given by RPLIDAR A2M8 [18]

This (distance, angle) data in the form of a point cloud is to be used by the path planning algorithm.

Up to CM2

In this period, the working principle of Lidar was studied. Although RPLIDAR A2M8 has already relevant documentation and libraries such as and most most importantly SDK, it is not compulsory to complete this step, yet it is important to beware of the working algorithm in the sense that if a problem occurs, we should at least be able to know or search where it is. In this period, the algorithmic system behind Lidar that links the output to the input was found to be as follows:

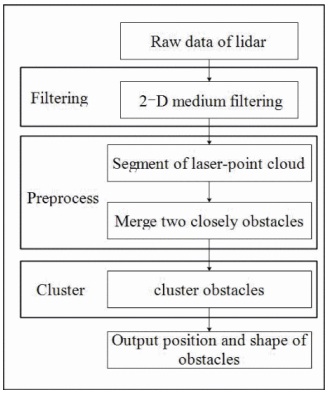

Figure 14: Working principle of Lidar [19]

As it can be understood from the figure above, the output which consists of positions and shapes of obstacle is obtained from the raw data in three major steps: filtering, preprocessing and clustering. In the filtering part the main objective is to remove the noise which is supposed to ease the preprocessing step, and for that, median liters are being used. On the other hand, each data point in the distance map generated by the Lidar has a coordinate. In the second step which is preprocessing, the relative distance to the Lidar corresponding to one coordinate exceeds the

maximum range, then such points are placed into different blocks. If it is not, such points are placed into the same block. The final step is done to simplify the complex point cloud data. For that, the blocks generated in the preprocessing step are classified into line, circle and rectangle categories based on the detection algorithm. [20]

RPLIDAR has been modeled in simulation which is to be mentioned later and the actual Lidar has been successfully operated using the SDK written for the RPLIDAR A2 family.

Inside the SDK there are three main functions defined: ultrasimple, simplegrabber and framegrabber. Among these, ultrasimple simply connects to an RPLIDAR device and outputs the scan data to the console. Simplegrabber, demonstrates the process of getting RPLIDAR's serial number, firmware version and healthy status after connecting the PC and RPLIDAR. Then the demo application grabs two round of scan data and shows the range data as histogram in the command line mode which is built with asterisk symbols. The final one, framegrabber, shows the laser scans in the GUI such as the following image:

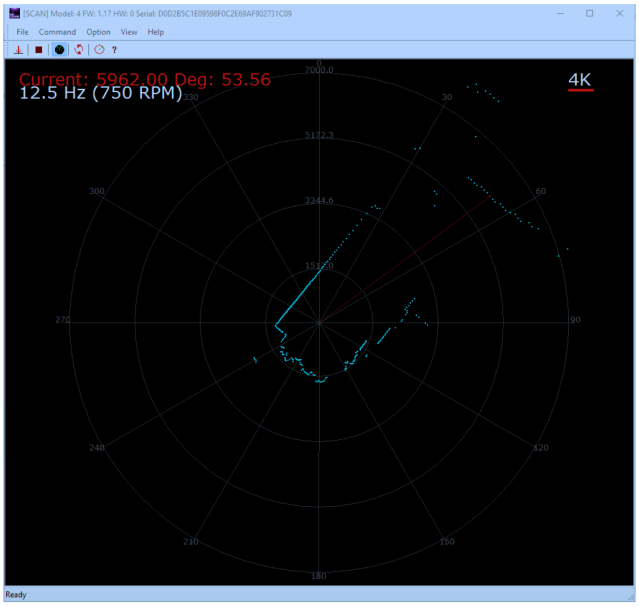

Figure 15: The debugging GUI of RPLIDAR [18]

Between CM2 and CM3

During this period, the modeling of the RPLIDAR in the simulation environment Gazebo and getting data from the real Lidar goals are achieved. Even small objects can be detected accurately. To test that, the position data of a hand in different poses has been drawn, and as the pose changes, the red line which represents the hand (in the middle of the RViz image) changes as well as in the figures below:

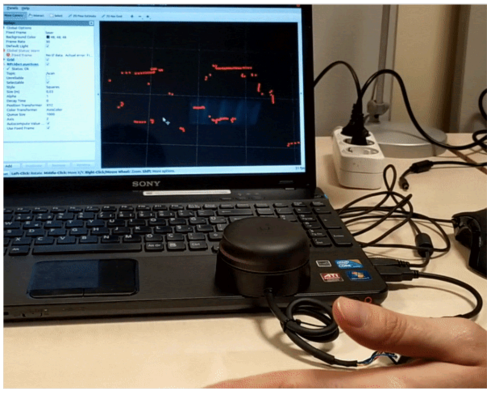

Figure 16: Lidar output when the hand takes a horizontal pose

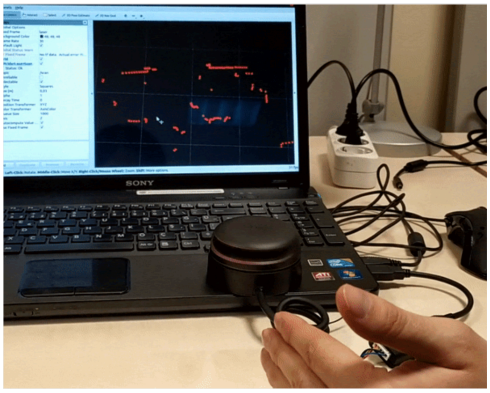

Figure 17: Lidar output when the hand takes a normal pose

In addition, the lidar data has been incorporated into navigation stack. Real time collection and publishing of the lidar scan is achieved to provide robust input to TEB local planner. Therefore, TEB local planner can use the relevant Lidar data and draw the path which the robot is supposed to follow.

Future Plans

We are able to get accurate data both in real life and simulation and this data can be fed into TEB planner. Therefore, Lidar work package is complete now and the experiments regarding this work package have been conducted by A. Bilgehan Baspinar.

Possible Risks

Since our actual Lidar has already shown to work, the only risk may occur in the future when integrating all these submodules of the robots together. For instance, Jetson Nano may not be able to supply all the Lidar, camera etc. This can be solved decreasing the resolution of the data to be processed or slowing the rate of the Lidar data flowing to Jetson Nano. However, this is not our concern right now since the robots cannot be assembled due to current circumstances.

5.2.2.3 Path Planning

Integrating target tracking and obstacle avoidance tasks into the motion of the robot is a critical stage of this project. Using the information provided by target tracking and obstacle detecting LIDAR modules, a mobile robot should able to construct a 2D map that demonstrates the relative position, velocity and acceleration of any object. Essentially, a mobile robot should be capable of planning its path through this map by analyzing the information delivered by aforementioned modules.

Up to CM2

The literature research we had done up until the Committee Meeting 1 showed us that, for path planning for obstacle avoidance and target tracking, "Potential Field Method" was quite a clearly defined and easy-to-implement method.

Yin - Yin [21] defines an attractive potential function with respect to the relative position, velocity and acceleration between the goal and the mobile robot, additionally a repulsive potential function considering aforementioned relative differences between the obstacle and the mobile robot in a dynamic environment (i.e. moving target and obstacles). This method benefits from a virtual force, calculated from the potential field of the ego robot, to plan the ego robot's motion with right positions and velocities to provide a similar motion trend with the goal and a contrary trend with surrounding obstacles.

However, our research following CM1 guided us to understand the limitations of this method more evidently. Koren and Borenstein [22] show that considerable defects are found due to the inherent limitations of the method. Even though Potential Field Method seems to be elegant and simplistic enough to attract many people, there are a couple of crucial complications that forestall its usefulness:

- Local minima may occur when a robot ends up stuck at a dead-end. Although there are heuristic solutions, they often cause non-optimal global paths.
- When there are closely spaced obstacles ahead of the robot, it may not find any passage due to the combined repulsive force which may be greater than the attractive force.
- Oscillations may occur when the robot is at a narrow passage or in the presence of obstacles.

As we became aware of these limitations, and the complexity of the solutions that attempt to solve these defects of the method, it seemed clear to us that another method should be proposed for path planning task.

ROS provides us a great opportunity to construct a well-connected system. In order to exploit the advantages of using ROS, built-in Navigation package is determined to be utilized. This package requires information from odometry (data from the motion sensors), sensor streams such as laser scan, and a goal pose as its input in order to output velocity commands which are sent to the mobile base. One of the main perks of Navigation stack, it allows us to utilize "move_base" package which is a major component of the Navigation stack.

Provided a goal in the 2D world, the ”move_base” package enables the mobile base to act in order to reach the goal. The implementation behind it is linking a global and local planner which require 2 costs map, global cost-map and local cost-map.

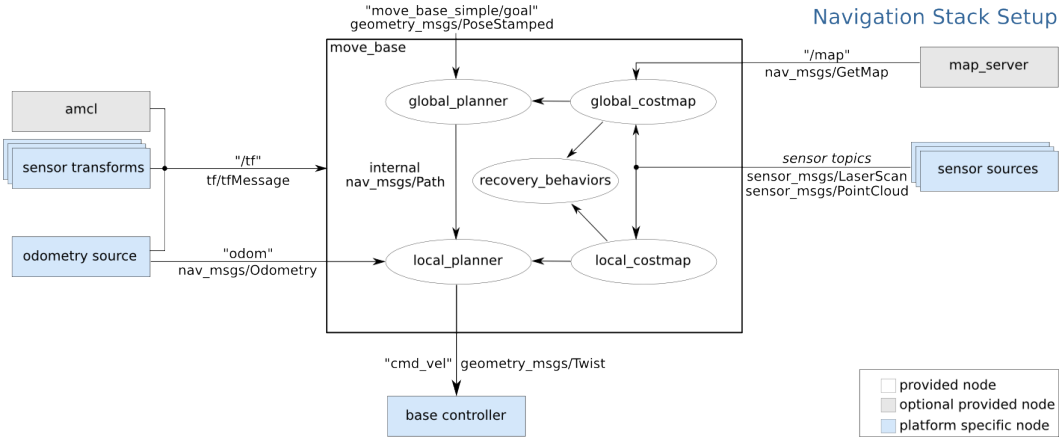

Figure 18: A high-level view of the move_base node and its interaction with other components [23]

In order to have costmaps, we implement ”costmap_2D” package, which constructs a 2D costmap by processing the sensor data such as laser scan or point cloud, which can be created from Lidar data. An example is given below:

Figure 19: A costmap example [24]

In Figure 19, the obstacles are represented by the red cells, the blue cells are the regions where the center of the robot should never be in to prevent collisions, and the red polygon is the footprint of the robot. Essentially, to avoid collisions, the footprint should never intersect a red cell.

Our new proposed method which replaces the Potential Field Method is Timed Elastic Band (TEB), which in its essence, built-upon the aforementioned internal packages of ROS.

In their paper, Rösmann et al. [25] show that in the context of Model Predictive Control (MPC), a joint trajectory representation by merging control inputs, time intervals and states is built in order

to plan time-optimal trajectories. In the control loop of TEB, MPC combines the planning with the state feedback, thus it approximates the time-optimal trajectory solution of analytical analysis in a few iterations during its runtime.

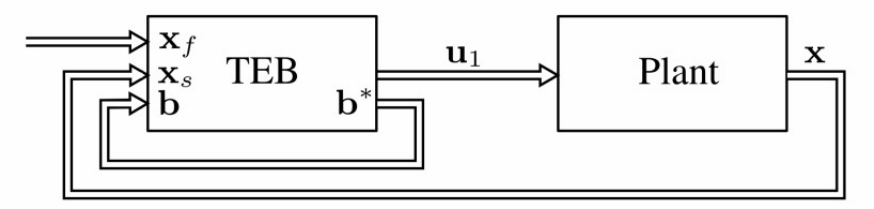

Figure 20: Closed-Loop TEB Control [25]

where x_f is the final state, b^* is the optimal trajectory, u_1 is the imminent control action of the planned sequence, x_s is the initial state and x is the state output of the plant.

An online optimal local planner is implemented for navigation in TEB package which is a plugin for the ROS navigation package. A global planner constructs an initial trajectory which is optimized during the runtime of TEB considering the following:

- Minimization of the trajectory execution time
- Maximization of the separation from obstacles
- Compliance with kinodynamic constraints

A multi-objective optimization problem is solve to obtain the optimal trajectory, and since the weights of the optimization problem is accesible by the user, we can determine the general behaviour of the trajectories and thus fine-tune it for customization purposed.

The optimal trajectory is efficiently obtained by solving a sparse scalarized multi-objective optimization problem. The user can provide weights to the optimization problem in order to specify the behavior in case of conflicting objectives.

In order to solve the local minima problem, an extension to TEB is developed. In paralel, the extension develops a subset of trajectories and the local planner selects the globally optimal one among the candidates.

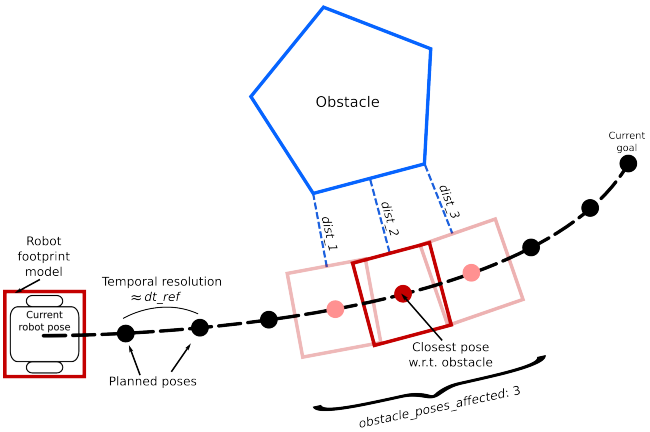

Figure 21: A snapshot of a TEB planning scenario [26]

The demo we made for CM2 presentation is given below in Figure 22. A 2-wheel differential-drive non-holonomic robot is used to test TEB method. As in the real-life scenario, we are not going to have a map of the area that the mobile robots are moving around, in this Gazebo simulation, an empty map, which means an empty global costmap, is given to the global planner. Therefore, the global planner does not possess any information about the objects existing in the simulation world. Considering Figure 18, the global planner of TEB takes the information of an empty map, the sensor information and the location of the goal, which is demonstrated with the green arrow on Figure 22, thus provides a global trajectory to the local planner.The local planner of TEB optimizes the global trajectory utilizing the sensor information which can consist of laser scan or point cloud data. The optimized local trajectories are feed to the mobile base as commands of navigation via "cmd_vel" channel which is used to transfer velocity commands.

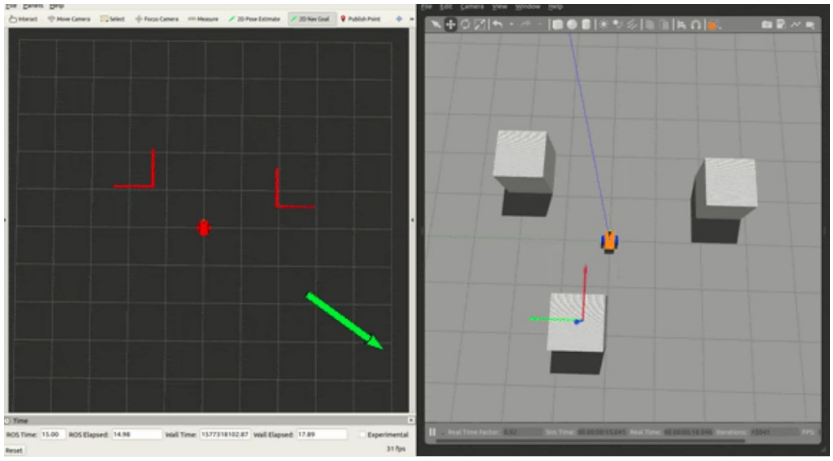

Figure 22: A snapshot of a TEB demo from CM2 Presentation

Between CM2 and Present

Initially, we implemented the TEB method to the Dagu Wild-Thumper robot in Gazebo environment. The parameters of the TEB local planner are configured via *base_local_planner_params.yaml* file. Some of the critical parameters are the following:

1. $odom\_topic$: $odom$
2. map_frame : $odom$
3. Trajectory Related Parameters:
   (a) $feasibility\_check\_no\_poses$: $5 \Rightarrow$ This parameter specifies the maximum number of poses on the predicted plan the feasibility should be checked each sampling interval.
   (b) $max\_global\_plan\_lookahead\_dist$: $2.0 \Rightarrow$ This parameter specifies the maximum total Euclidean distances of the subset of the global plan taken into account for optimization.
   (c) $min\_samples$: $5 \Rightarrow$ This is the minimum number of samples.
4. Robot Footprint Parameters:
   (a) $footprint\_model$:
      i. $type$: "$line$"
      ii. $min\_obstacle\_dist$: 0.55
      iii. $line\_start$: $[-0.1, 0.0]$
      iv. $line\_end$: $[0.1, 0.0]$
      v. $vertices$: $[[0.2, -0.1], [0.2, 0.1], [-0.2, 0.1], [-0.2, -0.1]]$
5. Goal Tolerance Parameters:
   (a) $xy\_goal\_tolerance$: 0.35
   (b) $yaw\_goal\_tolerance$: 0.5
6. Obstacle Related Paramaters:
   (a) $min\_obstacle\_dist$: 0.55
   (b) $costmap\_obstacles\_behind\_robot\_dist$: $0.3 \Rightarrow$ This parameter is the maximum distance to the obstacles behind the robot which should be taken into account.
   (c) $costmap\_converter\_plugin$: "$Lines$" $\Rightarrow$ This is the plugin that converts the cells that are marked as obstacles in the cost map to line obstacle if they are connected in a linear fashion.
   (d) $costmap\_converter\_rate$: $5 \Rightarrow$ This is the rate that defines how often the *costmap_converter* plugin processes the current costmap.
7. Optimization Parameters

(a) $weight\_obstacle : 2 \Rightarrow$ This is the optimization weight for keeping a minimum distance from obstacles. (Default: 50)

Originally, TEB is developed to work in an environment whose map is already acquired. However, in this project we assume to have no prior information about the environment, thus TEB module is required to work without a map. To this aim, *global_costmap_params.yaml* and *global_costmap_params.yaml* are configured to have the following parameters:

- $static\_map : false$
- $rolling\_window : true \Rightarrow$ This parameter indicates that the cost-map only represents the local surrounding of the robot (10m x 10m)

Possible risks that might occur are as follows:

1. Due to highly tolerant parametrization, path planning module might not seem to reach the target, in terms of staying relatively far from the target in a seemingly inaccurate pose
2. Due to having a "line" cost-map converter, some obstacles may not be correctly identified and thus the planned path might seem sub-optimal.

Future Plans

Considering the fact that Leg Tracker module is completed, Path Planning part is completed, as well after the parametrization was carried out as described above. Therefore, in this part, no further steps need to be taken.

5.2.3 Simulations and Real Life Testing

5.2.3.1 Gazebo

Gazebo is a platform that works as simulation atmosphere for the Robotic Operating System(ROS). ROS is for the back-end development area which the control algorithms and Artificial Intelligence techniques are implemented. For the testings of real-world like environment, Gazebo appears to be decent alternative. Implemented algorithms are tested upon 3D modeled robots and used peripherals. Sample image for a gazebo simulation can be demonstrated as such:

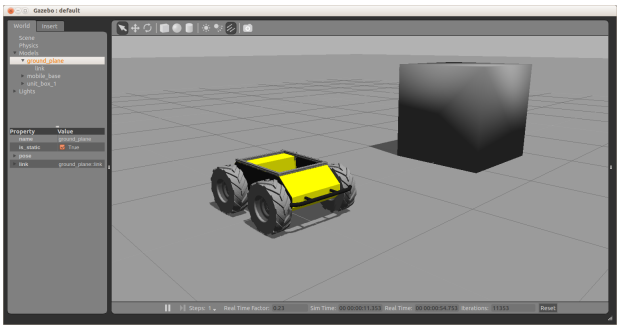

Figure 23: Gazebo Simulation Software [27]

Up to CM2

All the member who have been working on ROS and Gazebo completed setup process and launched empty simulations on the software. The ROS version for the all member are selected as the current version of ROS which was Melodic. Using the documentation provided for the installation and initial launch methods are studied.

Gazebo simulations are done on a sample model called Husky Robot which is the robot provided in the Figure 23 before CM1. Husky online tutorial is completed from SmartLab website[27] by Arda Yüksel. The online documentation regarding the usage of peripherals such as Action Camera and LIDAR is analyzed. As the robot model was decided in the Land Robot Chassis sub-task of Mechanical Setup, pre-built DAGU Wild Thumper models for Gazebo are searched. Up to CM2,a working Wild Thumper Model used instead of Husky Robot was aimed.

During the period between CM1 and CM2, various robot models are tested such as Turtlebot and online tutorials in the fields of navigation, SLAM and robot control is done through usage of Turtlebot and a designed 2 wheeled robot model. In order generate this model online tutorial that covers the basics of robot design for Gazebo are studied by Arda Yüksel, Mert Acar, Bilgehan Başpınar and Cevahir Köprülü from MooreRobots[28]. Since the fundamentals that the tutorial covers are also related to the development of LIDAR, Path Planning and Camera Output Integration over ROS, all the member associated with the relative tasks completed this set of tutorials successfully. Using their knowledge, Arda Yüksel and Cevahir Köprülü generated a sample 2 wheeled robot to comprehend the robot design in Gazebo. The model simulations can be seen from the image below as:

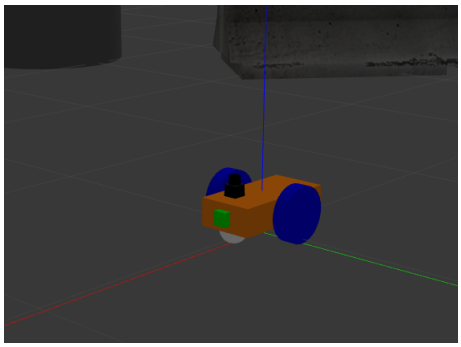

Figure 24: 2 Wheeled Robot with Hokuyo Laser[28]

By using this model, movement and output channels of simulations are learned. Furthermore, by experimenting in this setup the concept such as Navigation, Odometry and Teleop operations which are fundamental for ROS are examined. In this model, aside from the 2 wheeled robot model, one black Hokuyo Laser, which is built-in model for SLAM is generated to study the basics of LIDAR integration. In addition to this model, Cevahir Köprülü found a DAGU Wild Thumper 4WD model that works in previous versions of ROS.

The URDF file, which contains the information related to DAGU model and its components such as wheels and peripherals, is analyzed and adjustments are done in order to generate a launch file compatible with ROS Melodic. After these adjustments, using the world launch file designed for the 2 wheeled setup, DAGU Wild Thumper is generated as such on Gazebo Environment:

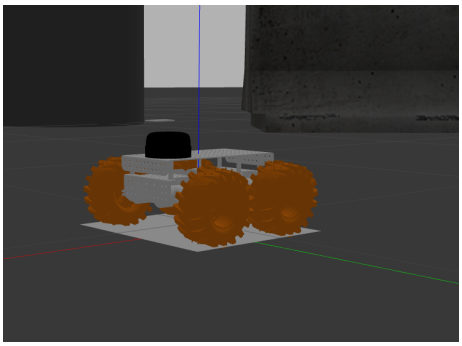

Figure 25: DAGU Wild Thumper 4WD on Gazebo

Using the established robot model, custom model is generated with the ROS essential tools which are publishers and subscribers. ROS publishers return the outcome of the peripherals such as LIDAR and subscribers send commands to robot and their components such as motor drivers defined in URDF file. The nodes of the current setup is given as:

```
lordspoiler@arda-GT60:~/mybot_ws$ rostopic list
/clock
/cmd_vel_out
/diff_drive_controller/cmd_vel
/diff_drive_controller/odom
/diff_drive_controller/parameter_descriptions
/diff_drive_controller/parameter_updates
/gazebo/link_states
/gazebo/model_states
/gazebo/parameter_descriptions
/gazebo/parameter_updates
/gazebo/set_link_state
/gazebo/set_model_state
/imu
/joint_states
/odom
/rosout
/rosout_agg
/scan
/tf
```

Figure 26: Nodes of the Simulation

In the setup /scan and /diff_driver/cmd are crucial. In their simulation, system uses /scan data to visualize the LIDAR output as in the Point Cloud Format. The /cmd is necessary to move the robot. These input and output channels are tested for the usage in the Integration ROS with YOLO and LIDAR. After confirming their validity, the simulation files are used in other sections.

Between CM2 and CM3

This task was mainly finished by the second Committee Meeting. According to the necessities of the other simulation components additional ROS packages are added into the system in the mean time. The simulation files for the RVIZ and GAZEBO are combined into single launch file. Furthermore, the controller is initiated when the simulation starts. Simulation launch can be demonstrated as:

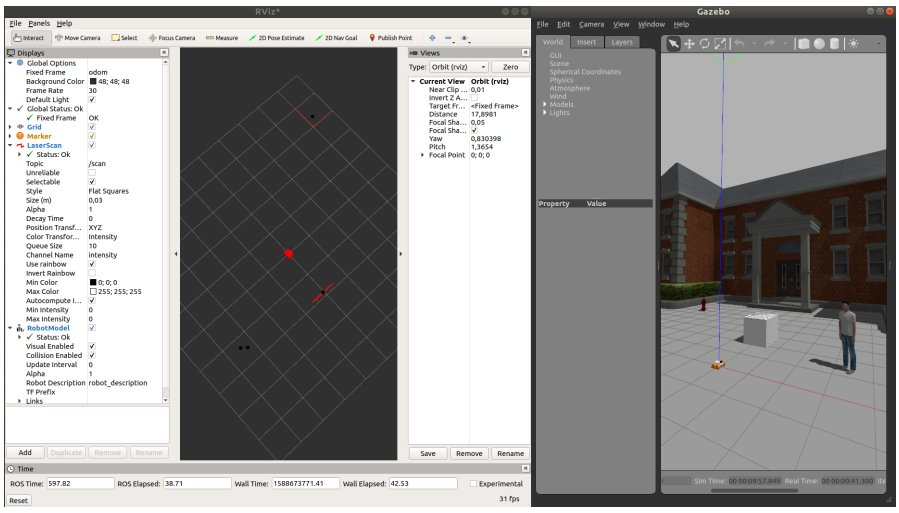

Figure 27: Simulation Launch with RVIZ and GAZEBO

The packages for the Leg Tracker and TEB Local Planner can be observed from the Gazebo

simulations. It is due to the fact that these packages are launched with the simulations beforehand. In addition to that /diff_drive packages are adjusted in the Control with ROS section for the simulations. Nodes are organized compared to the first semester. Move base is for the TEB and and joint leg tracker is for the Leg tracker. Odometry issue is solved by relay odom node. Controller nodes are obtained from diff drive package. The overall nodes are:

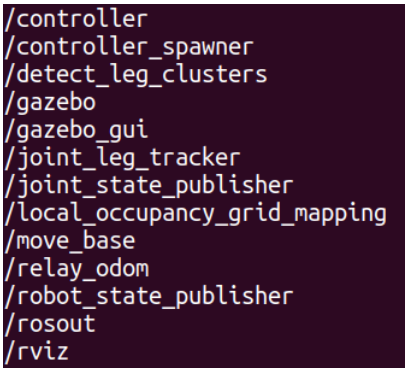

Figure 28: CM3 Presentation ROS Nodes

Since Corona Outbreak negated the usage of hardwares, Gazebo simulation gained more significance for the final demo. Real world like environment based on the EE/EB budiling is designed for the test case. In addition to that environment dedicated for the testing procedure of the Leg Tracker is also designed.

Future Plans

The final demo for the GAZEBO simulations are already demonstrated in the Committee Meeting 2 Presentation. The room for improvement lies within the world building for the simulations. The world may have dynamic objects. However, the possibility of the phenomena for the GAZEBO environment is unknown. Other than this, as it can be seen from the Figure 4, this subtask is finished.

5.2.3.2 Control with ROS

Up to CM1

The need for controllers and possible motions that has to be performed by robot were discussed.

Between CM1 and CM2

Designing a PID (proportional–integral–derivative) controller using ROS to control the robot is planned. As selected motor drivers can be controlled with Pulse Width Modulation (PWM) technique, the controller can control the robot by outputting PWM signals and send command to motor drivers.

Up to CM3

A control algorithm gets its inputs from the odometry frame and the navigation stack of the robot. However, these inputs are very high frequency and erratic. In order to overcome this problem, a running average filter was implemented to smooth out the incoming input and make the controller's job easier. After the filter the inputs are fed into a differential drive controller which takes the published velocity commands and passes it through a closed loop PI controller to determine the PWM commands for wheels at each side.

Future Plans

The PID controller will be designed on ROS. This controller will control all of the motions of the robot, taking the input from the computer vision system in the robot and by running control algorithms, it will output control commands. This commands will be taken as input by motor drivers and the motors and wheel of the robot will move according to these control commands, hence robot will have an algorithmic motion. During the controlling process of the simulated robot, we may encounter issues due to the usage of Wild Thumper 4WD in Gazebo. To overcome this issue, the concept of Skid Skewer Driver control which allows multiple wheel setups to work properly both in simulations and real world testing can be integrated if necessary.

5.2.3.3 Modelling Lidar

One of our project's major parts is implementing Lidar since the autonomy of our robots is based on tracking and obstacle avoidance, therefore path planning. As mentioned, for the path planning algorithm we used to plan to use Potential Field Method and we have switched it to Timed Elastic Band Method, which requires a data flow in the form (distance, angle) and in the structure of sensormsgs/LaserScan Message, which is a class in the Lidar ROS library.

Up to CM2

In this period, since our simulation skills were not good enough, we could only manage to simulate a Lidar in Gazebo, but not the one that we planned to use, that is RPLIDAR A2M8. Still, at the time it was important to be able to simulate a Lidar in the sense of learning generating simulations that are close to our case. Therefore, the following simulation and visualization was made using a 180-degree Lidar:

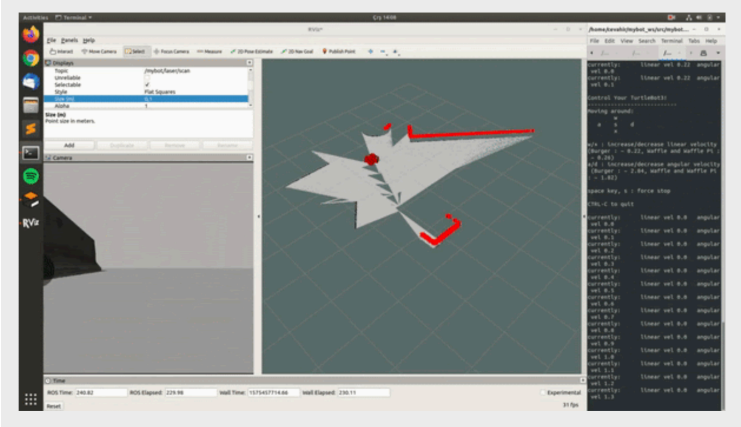

Figure 29: Lidar simulation result before CM I using a simpler Lidar Gazebo model

In the second progress meeting of first semester, when performing a simple simulation a simple laser model was used as the initial step. After that, this simple laser model was successfully replaced by the real Gazebo model of RPLIDAR A2M8 and using and modifying the ROS node written for RPLIDAR [29], a subscriber node was written in order to be able to listen the data coming from the "/scan" channel, in the structure of sensormsgs/LaserScan Message, published by the Lidar.

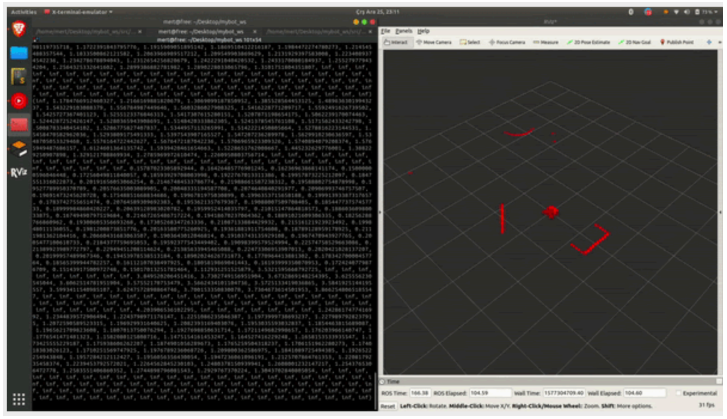

Figure 30: The RPLIDAR data flowing on the left, RViz visualization on the right which shows the obstacles in the simulation world

As it can be seen from the figure above, before CM2, we were able to get the Lidar data in the form we need for the path planning algorithm and we can visualize this data on RViz, using the real model for RPLIDAR A2M8 this time as opposed to the case before the Committee Meeting I.

Between CM2 and CM3

In the TEB Local Planner integration the unstable nature of the LIDAR component is issued. LIDAR demonstrated false positives on the backside of simulated DAGU model. These issues are resolved by using the preferences provided by RPLIDAR[18] documents. The Xacro file of the

the robot is adjusted. Stable solution for the TEB is attained after the solution. The LIDAR is integrated accurately to the Path Planning simulations before Progress Meeting of the second semester. Before Corona outbreak, this package was already completed in simulation environment.

Future Plans

This subtask is mainly completed before the first Progress Meeting of second semester. Simulations are verified in the demo presented in the Path Planning part of the respective meeting. Stability of the LIDAR simulations can be derived from the success of TEB Local Planner.

The difficulties that are addressed in the section Between CM2 and CM3, are solved by the members. However, real life testing of the LIDAR cannot be demonstrated since the main assembly of the robot is unable to be generated due to circumstances raised by Corona Virus.

The limitations of the simulations are quite clear. As these programs are designed, the world building prepares models for the general cases. For more specification and testing, real world testing is also essential. Creating a real world like environment and designing robot for simulations are also difficult. However as it addressed by Gazebo section, real world inspired world is designed to generate accurate results.

5.2.3.4 Beacon

Up to CM2

Beacon related work was initiated in the second semester.

Between CM2 and CM3

The beacon sets were provided to us towards the end of February, thus the related work only includes testing of the set in Dashboard, the official Marvelmind software. Marvelmind HW v4.9 Starter Kit is used in this part includes 5 HW v4.9 Beacons and 1 Modem, as seen in Figure 31.

Figure 31: HW v4.9 Starter Kit: 5 x HW v4.9 Beacon & 1 x Modem [30]

HW v4.9 Beacon utilize ultrasonic receiver/transmitter for localization. They have 2 different functions:

1. Stationary Beacon:
   (a) In this mode, the beacon is mounted on the wall or ceilings above the robot with ultrasonic sensors facing down.
   (b) The functionality is to provide reference points for localization of the mobile robot.
2. Mobile Beacon a.k.a ”Hedgehog”:
   (a) In this mode, the beacon is placed on a robotic vehicle, drone etc
   (b) Its functionality is to trace the location of the mobile robot according to stationary beacons.

Modem is the central controller of the system. During the operation of the Navigation System, it must be powered on. They are used to set up the system, monitor it and interact with Dashboard. The radio connection with modem is handled on 915 MHz band via on-board antennas.

The architecture of the program uploaded to beacons and modems is called ”Non-Inverse Architecture” (NIA). The general case for utilizing NIA is when mobile beacon is installed on a noisy vehicle, and the stationary beacons are in quieter places. The testing included 3 stationary beacons and 1 hedgehog. Since the testing was carried out in a small room where no critical disturbance was observed, NIA was appropriate. An instance from the testing can be viewed in Figure 32.

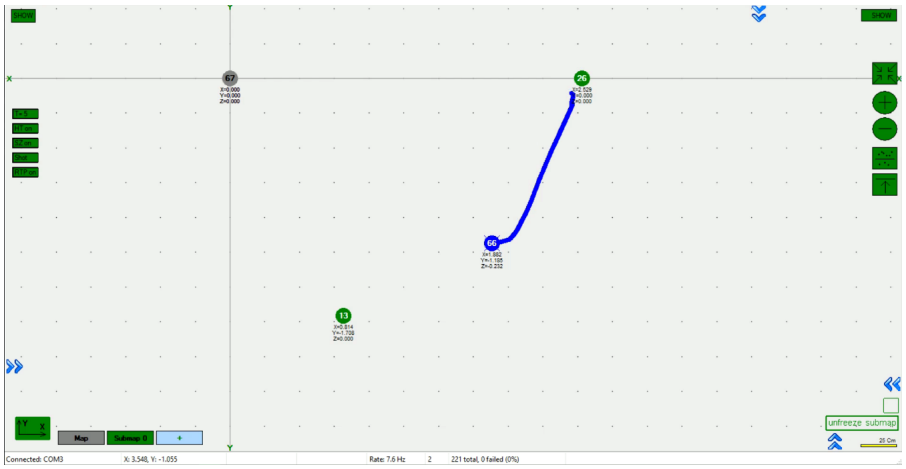

Figure 32: Test on Dashboard with 3 HW v4.9 Beacons as Stationary and 1 as Hedgehog

Future Plans

Beacon ROS module was developed for ROS Kinetic, which is an older version compared to what has been used since the the start of the semester, ROS Melodic. This compatibility issue prevents us from moving further to visualize the Beacon data on Rviz which would show that the localization data can be utilized in ROS. Since Beacon data cannot be integrated to any other module that we have developed before, even though Beacon data is visualized in Rviz on ROS Kinetic by installing it to a PC from scratch, it would not be more useful than the level reached in the current progress.

5.2.3.5 Leg Tracker

This package is added after the first Progress Meeting of the second semester. In this subtask, the aim is to identify human objects within robots range and generate goals for the Path Planning part. It is named after the ROS package obtained from the research of A. Leigh et al[31]. This Leg Tracker uses LIDAR point cloud input to detect human legs in the given environment.

Up to CM2

This package is listed in the second semester in order to meet the demands of the Path Planning. Therefore, the work done is after the CM2.

Between CM2 and CM3

The necessity of the package is addressed firstly in the second semester. Before that, TEB mainly worked with user inputs as the goals for the robot. In order to possess fully functional autonomous robot system, the goal should be determined by the robot itself. Leg Detection algorithms are selected as a way to obtain the human position.

For this purpose, Leg Detector algorithms of ROS tools are searched from ROS Leg Detector[32]. As it was mentioned in the Progress Meeting 1 of second semester, the node resulted in mainly false positives and alternative methodology is needed to generate accurate results. For the Progress Meeting demo, the Path Planning demonstration was handled by giving goal information via RVIZ and the Python script implemented with manual goals.

After the Progress Meeting viable options are found from the research of McGill University and University of Alberta[31]. The option is called as Leg Tracker. This is the enhanced version ROS's original Leg Detector. It had higher accuracy and compatible with RPLIDAR.

In order to check the validity of the Leg Tracker, a world with various objects are designed as:

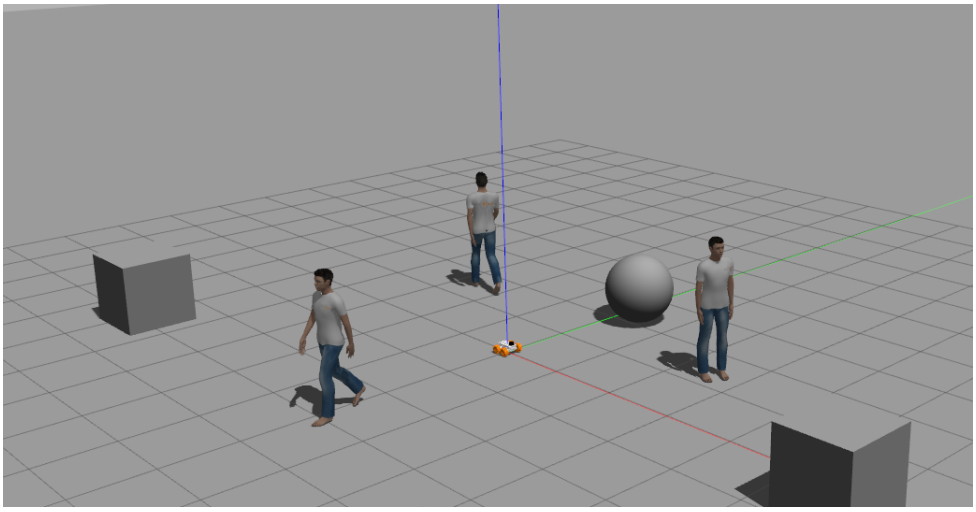

Figure 33: Gazebo Simulation World designed for Leg Tracker Test

The humans in that world are correctly identified by Leg Tracker. Blue dots indicate the identified Leg. Due to viability issues in long range, identify single leg is seemed to be a sufficient solution for the goal selection. The results are demonstrated with RVIZ tool:

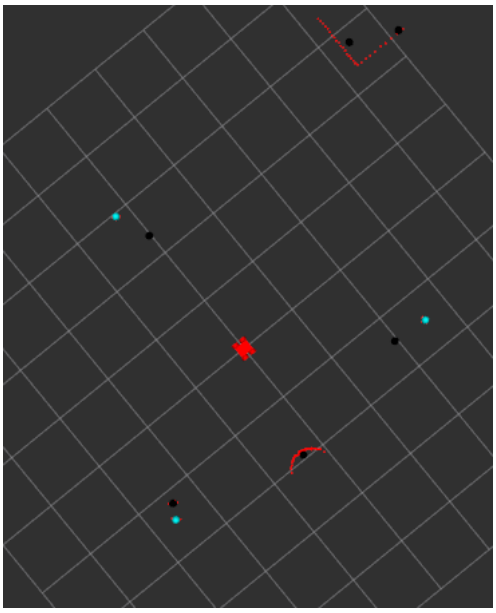

Figure 34: Leg Tracker Test World's Identified Legs in RVIZ

Using the initial python script implemented for manual goals, People Tracker python script is implemented by. In this script, selected human is first identified by Leg Tracker. Later on the position and the orientation of the human object is distributed over a ROS topic. This topic is later by the controller Python script in order to send the goal to TEB Local Planner. By combining these two scripts in the simulation, a final demo is accomplished by Committee Meeting 3 Presentation.

Future Plans

Due to Corona Virus, this package cannot be demonstrated in a real world environment. Therefore the validity of the given Leg Tracker is not confirmed for the real world implementation. However, the simulation world is designed considering the attributes of the original demo setup.

In the simulation setup, the Leg Tracker is not tested for moving obstacles and people. Therefore, for the future that can be tested. It is also commented during the presentation and appears to be the risky component of the Leg Tracker. Due to lack of YOLO in simulation environment, Leg Tracker algorithm is not adjusted to work in a wider range desired for the real world. LIDAR range is not as suitable as camera for our purposes to detect humans.

The filtering part designed for YOLO cannot be integrated due to its incompatibility to GAZEBO. The simulation and software aspect of the Leg Tracker is fully completed. However the hardware demonstration is adjourned due outbreak.

5.2.4 Swarm Intelligence

Swarm Intelligence is omitted due to Corona Virus.

For multiple and varying number of robots in the tasks, it is crucial to have a general structure for tasks that is mentioned in previous subsections. In order to generate a working model that can deliberately organize work distribution with multiple robots and control their fundamental interactions with humans assigned for the operations, swarm intelligence was planned to be implemented.

Up to CM2

As Brambilla [33] defines it, swarm intelligence is "an approach to collective robotics that takes inspiration from the self-organized behaviors of social animals". This concept mainly developed by imitating insect swarms on the basis of their communication within swarm and work distribution. Its applications on robotics is contemporary research topic and throughout this project, it is expected to be implemented for assisting as robot group for military operations. As it can be seen from the Work Breakdown Structure and Gantt Chart, the application procedure is omitted due to current events.

During research process for the swarm intelligence, the properties of possible robot swarms and specifications for the design procedure have been established. It is expected for robot swarms to fulfill the following requirements[34]:

- **Autonomous:** Robots should be able to decide on the actions given circumstances without human control
- **Ability to comprehend environment:** Robots should be able to respond to nature through data processing and decision making algorithms
- **Identical robots:** Robots should have similar equipment in order to take measures in case of possible losses in swarm
- **Local interaction abilities:** Robots should have a local network to share information
- **Cooperation abilities:** Robots should be able to perform operations cooperating with each other

For the development of the robot swarm with multiple robots, these properties will be taken into consideration. Since the robots must act on without any human control autonomous property is a must regardless the usage of swarm intelligence. In addition to that ability to comprehend environment is also a crucial factor which will be implemented before the development of swarm intelligence through SLAM and Computer Vision techniques.

Robots are expected to switch roles as one can be master or slave for different tasks, therefore identical robots will make it easier to implement varying work distribution mechanic. Local interaction abilities are planned to perform via networks which will be covered later on. Cooperation abilities is one of the important aspects which will allow exploration tasks to perform more efficiently.

One of the most important reasons that swarm intelligence is essential for the future of the project is that its viability in behavior predictions for multi robot systems. For this task following algorithms and techniques will be analyzed and simulated:

- **Probabilistic finite state machine design(PFSM):** According to input data, robots alter their behavior in a predefined probabilistic finite state machine model.
- **Reinforcement learning:** Through trial and error in simulations, swarm learn to tackle sample situations.
- **Evolutionary robotics:** Effectiveness of the behavior is tested using evolutionary computation

Of all the algorithms mentioned above, PFSM is crucial since it appears in most of the researches on the swarm intelligence as Brambilla[33] describes. It allows robots to select appropriate behaviours with the probabilities attained during the simulation process. For the generation of the probabilities reinforcement learning and evolutionary robotics are used.

Between CM2 and CM3

Due to technical difficulties involving the lack of hardware and change in the timeline of the project, the packages are omitted. Before the Corona outbreak this package was planned to be started in the second half of the second semester. Therefore, the work done in this package is the researches commenced before CM2.

Future Work

Since the plan of the project is adjusted according to Corona Virus status report, this work package will not be included in the final product.

6 Online CM3 Presentation and Critics

Following the feedback to the proposed plan for the rest of the semester in the status report, we have submitted the report with included feedback to our supervisors from ROKETSAN A.Ş. The response to the proposed plan came after 10 days of analysis. The response included:

1. Submission of current progress with visual content including screen casts, screen shots etc
2. Submission of the Gazebo simulation package in a deployable form
3. To complement the work in control module, addition of noise to Gazebo odom data to demonstrate the robustness of the controller
4. Submission of progress made with Beacon HW set by providing visual content

5. As a bonus, addition of camera module to Gazebo and implementation of Computer Vision module in Gazebo

Considering the fact that we had about 8 days to the online CM3 Presentation on the day we received this response, as a group, we aimed to prioritize the tasks that can be completed in that limited amount of time. Those tasks were the following:

1. Integrating leg tracker module to identify the human target
2. Designing a simple filter unit in a moving average fashion to smooth out the TEB movement command output
3. Testing beacon hardware in its official software Dashboard to understand how it works, what the limitations are etc.

Designing a robust controller unit that takes a movement command as its reference input and utilizes odom data for feedback was not a feasible plan to be carried out in 8 days, thus we decided to implement the moving average filter which provides its output to already existing differential drive unit which consists of a PI controller. This filter was already a planned module, as in the real set-up we were supposed to smooth out the TEB outputs in time due to possible erratic commands received while being surrounded by obstacles which were in close proximity.

The work done in the beacon part includes introductory tasks, essentially, after understanding how the hardware works, what the functions of Beacons are, how maps are built etc, we planned to use the ROS module to receive and visualize data on Rviz. Due to both time limitation and incompatibility of the ROS version. Until now, all ROS related work has been done in ROS Melodic. However, Beacon documentation indicate the necessary version is ROS Kinetic, which is the prior one to ROS Melodic. Therefore, we couldn't deliver this task as we have planned.

Leg Tracker module has actually exceeded our expectations, as before the outbreak, our efforts to find or develop a feasible leg tracker did not conclude in satisfactory results. However, current module works quite well when the robot is close enough to the human target, which is what we have been trying to achieve.

Camera related work was not initiated, as we explained in CM2, the virtual image obtained in Gazebo cannot be inputted to Computer Vision module, due to the fact that YOLO V3, which is trained on real-world images, would not make any sense out of it.

The Gazebo simulation package including all the work done in ROS is submitted to our supervisors from ROKETSAN A.Ş.

Methods and Progress part includes figures which are obtained to demonstrate our efforts, thus this section can be visited to see the illustrations related to the work done for the Online CM3 Presentation.

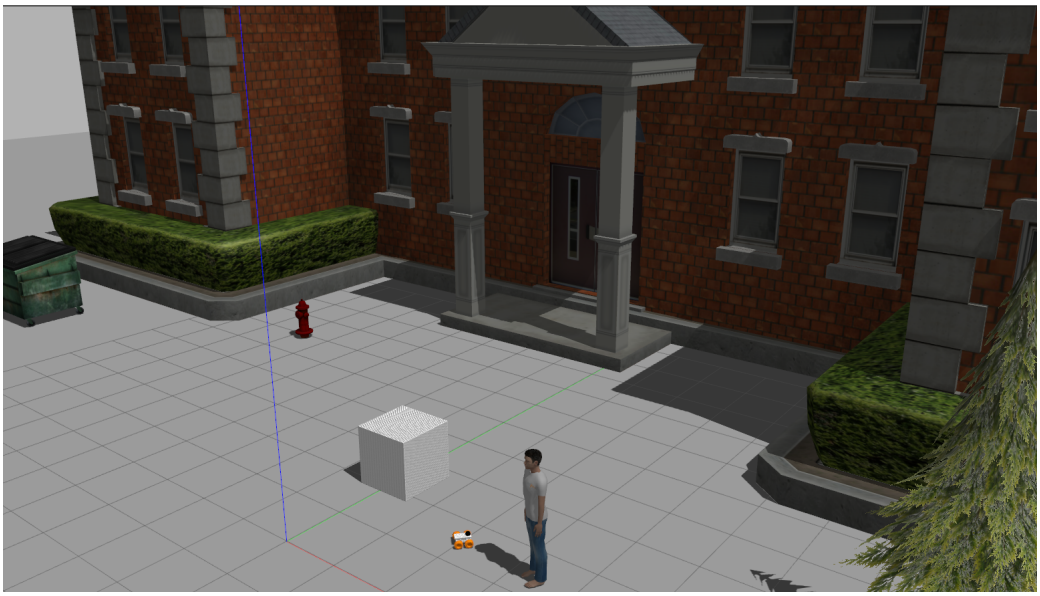

Figure 35: Online CM3 Simulation Demo

7 Equipment List

Equipment	Cost (in TL)	Obtained
Nvidia Jetson Nano (for First Robot)	1115	by purchase
Nvidia Jetson Nano (for Second Robot)	No Cost	from ROKETSAN
Dagu Wild Thumper 6WD All-Terrain Chassis (x2)	No Cost	from ROKETSAN
Eken H9R 4K Action Camera (x2)	756	by purchase
RPLIDAR A2 M8 (x2)	No Cost	from ROKETSAN
Beacon (x2)	No Cost	from ROKETSAN
LiPo 7.4V 5200 mAh 2S (x4)	1072	by purchase
BTS7960B 40A Single Channel Motor Driver (x4)	180	by purchase
Xiaomi 10000 mAh Power Bank (x2)	200	by purchase
Marvelmind HW v4.9 Starter Kit Beacons	No Cost	from ROKETSAN
Total Cost	3323	

Table 1: BoM of the Project

Single robot consists of one Nvidia Jetson Nano, one Dagu Wild Thumper 6WD All-Terrain Chassis, one Eken H9R 4K Action Camera, one RPLIDAR A2 M8, two LiPo 7.4V 5200 mAh 2S, two BTS7960B 40A Single Channel Motor Drivers and one Xiaomi 10000 mAh Power Bank. The Bill of Material is given in Table 1 consists of all the materials for two robots. Our project objective was to build at least two robots in order to construct the swarm behaviour ,however due to Covid-19 outbreak, the implementation of second robot is not possible. ROKETSAN has provided us Marvelmind HW v4.9 Starter Kit for odometry and TEB purposes of the robot.

References

[1] Wikipedia.org. *ROKETSAN*. URL: https://en.wikipedia.org/wiki/ROKETSAN.

[2] Angelo Nikko Catapang and Manuel Ramos.
"Obstacle detection using a 2D LIDAR system for an Autonomous Vehicle".
In: *2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)* (2016). DOI: 10.1109/iccsce.2016.7893614.

[3] Pololu Robotics and Electronics.
Dagu Wild Thumper 6WD All-Terrain Chassis, Silver, 34:1. URL: https://www.tme.eu/Document/3042ea29739d84f52c2c511fe0a15337/DAGU-RS003B75.pdf.

[4] Nvidia. *Jetson Nano*. URL: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-nano/.

[5] NATO. *Autonomous Systems: Issues for Defence Policymakers*.
URL: https://www.act.nato.int/images/stories/media/capdev/capdev_02.pdf.

[6] Michael Barnes et al.
"Designing for Humans in Autonomous Systems: Military Applications". In: (Mar. 2014).
DOI: 10.13140/2.1.3500.7688.

[7] ROKETSAN. *ROKETSAN: About Us*. URL: http://www.roketsan.com.tr/en?current_ajax_id=9868&current_ajax_page_type=page.

[8] Oli Moser. *Introduction to Computer Vision With OpenCV and Python*. URL: https://dzone.com/articles/introduction-to-computer-vision-with-opencv-and-py.

[9] Brian Barrett. *Inside the Olympics Opening Ceremony World-Record Drone Show*.
URL: https://www.wired.com/story/olympics-opening-ceremony-drone-show/.

[10] Mengmeng Wang et al.
"Real-time 3D Human Tracking for Mobile Robots with Multisensors". In: (Mar. 2017).
DOI: 10.13140/2.1.3500.7688.

[11] *2040.1 - Standard for Connected, Automated and Intelligent Vehicles: Taxonomy and Definitions*. URL: https://standards.ieee.org/project/2040_1.html.

[12] *2040.2 - Standard for Connected, Automated and Intelligent Vehicles: Testing and Verification*. URL: https://standards.ieee.org/project/2040_2.html.

[13] Ltd. Xiaomi Communications Co. *10000mAh Mi Power Bank 2S User Manual*.
URL: https://i01.appmifile.com/webfile/globalimg/UK/Manual/10000mAh_Mi_Power_Bank_2S.pdf.

[14] NovalithIC. *BTS 7960 High Current PN Half Bridge Motor Driver*.
URL: http://www.robotpower.com/downloads/BTS7960_v1.1_2004-12-07.pdf.

[15] Brian Schneider. *A Guide to Understanding LiPo Batteries*.
URL: https://rogershobbycenter.com/lipoguide.

[16] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement".
In: *CoRR* abs/1804.02767 (2018). arXiv: 1804.02767.
URL: http://arxiv.org/abs/1804.02767.

[17] Joseph Redmon. URL: https://pjreddie.com/darknet/tiny-darknet/.

[18] SLAMTEC.
RPLIDAR A2 Low Cost 360 Degree Laser Range Scanner Introduction and Datasheet.
URL: https://cdn.sparkfun.com/assets/e/a/f/9/8%20LD208_SLAMTEC_rplidar_datasheet_A2M8_v1.0_en.pdf.

[19] Yan Peng et al.
"The obstacle detection and obstacle avoidance algorithm based on 2-D lidar".
In: *2015 IEEE International Conference on Information and Automation* (2015).
DOI: 10.1109/icinfa.2015.7279550.

[20] Angelo Nikko Catapang and Manuel Ramos.
"Obstacle detection using a 2D LIDAR system for an Autonomous Vehicle".
In: *2016 6th IEEE International Conference on Control System, Computing and Engineering (ICCSCE)* (2016). DOI: 10.1109/iccsce.2016.7893614.

[21] Lu Yin and Yixin Yin. "An improved potential field method for mobile robot path planning in dynamic environments".
In: *2008 7th World Congress on Intelligent Control and Automation* (2008).
DOI: 10.1109/wcica.2008.4593709.

[22] Y. Koren and J. Borenstein.
"Potential field methods and their inherent limitations for mobile robot navigation".
In: *Proceedings. 1991 IEEE International Conference on Robotics and Automation* ().
DOI: 10.1109/robot.1991.131810.

[23] *ROSWiki*. URL: http://wiki.ros.org/move_base?distro=melodic.

[24] *ROSWiki*. URL: http://wiki.ros.org/costmap_2d?distro=melodic.

[25] Christoph Rosmann, Frank Hoffmann, and Torsten Bertram.
"Timed-Elastic-Bands for time-optimal point-to-point nonlinear model predictive control".
In: *2015 European Control Conference (ECC)* (2015). DOI: 10.1109/ecc.2015.7331052.

[26] *ROSWiki*. URL: http://wiki.ros.org/teb_local_planner?distro=melodic.

[27] SMARTlab-Purdue. *SMARTlab-Purdue/ros-tutorial-gazebo-simulation.*
URL: https://github.com/SMARTlab-Purdue/ros-tutorial-gazebo-simulation/wiki/Sec.-2:-Driving-the-Husky-robot-in-Gazebo.

[28] Rpw. URL: http://moorerobots.com/blog.

[29] ROS.org. *Rplidar Ros Package Summary*. URL: http://wiki.ros.org/rplidar_ros.

[30] *Marvelmind Robotics*. May 2020. URL: https://marvelmind.com/.

[31] Angus Leigh et al. "Person tracking and following with 2D laser scanners".
In: *2015 IEEE International Conference on Robotics and Automation (ICRA)* (2015).
DOI: 10.1109/icra.2015.7139259.

[32] *Leg Detector*. URL: http://wiki.ros.org/leg_detector.

[33] Manuele Brambilla et al. *Swarm robotics: a review from the swarm engineering perspective*. Jan. 2013. URL: https://link.springer.com/article/10.1007/s11721-012-0075-2.

[34] Erol Şahin et al. *Swarm Robotics*. Jan. 1970. URL: https://link.springer.com/chapter/10.1007/978-3-540-74089-6_3.