

**RAFIK HARIRI UNIVERSITY**

**SLUG PICKING ROBOT**

Done by

**ABDULRAHIM EL MOHAMED**

**JANA MARZOUK**

**NABIL MIRI**

**YAHYA AL JAMAL**

Submitted to

**DR. HASSAN HARIRI**

This senior project submitted in partial fulfillment of the requirements of the  
BE degree of Mechatronics Engineering Major of the College of Engineering at Rafik Hariri  
University

**MECHREF, LEBANON**

**May 2022**

Copyright©2022, all rights reserved

Abdulrahim El Mohamed (2018-0191)

Nabil Miri (2018-0003)

Jana Marzouk (2018-0001)

Yahya Al Jamal (2012-0143)

## ACKNOWLEDGMENTS

The completion of this project could not have been done without the guidance of our advisor Dr. Hassan Hariri.

A debt of gratitude is owed to Spexal Co. advisors, especially Eng. Ziad Chelala (CEO), Eng. Khalil Rai (CTO), Eng. Afif Swaidan and Eng. Ghassan Muhaidly for the technical, managerial, and financial support.

Special thanks to our parents and friends who walked with us through all the phases of the project and provided us with the emotional and mental support.

## ABSTRACT

The project discussed in this paper is a robot that aims to solve a big agricultural problem the world is facing today, which is slug detection and their collection safely to ensure the protection of plants and crops. The report involves design, realization, and validation of an autonomous robot able to drive on rough agricultural terrain. The robot is equipped with a camera that detects slugs using Machine Learning and navigates autonomously using ROS and GPS system. The slugs detected should be collected safely, using a well-designed collection mechanism, and released in a specific storage.

This report explains, in six chapters, all the steps followed to implement the proof of concept, of a compact, robust, serializable, and cost-effective robot that serves the general purpose stated above. First, we discussed the literature review and our competitors work, then we detailed our work. Finally, we prepared detailed tutorials, found at the end of the report, including all the technical steps applied in the autonomous navigation, and the computer vision sections.

The content of this report is confidential and should not be shared with any external party.

**Keywords:** Slugs, Autonomous Navigation, Detection, Artificial Intelligence, Machine Learning, OpenCV, ROS, Collection Mechanism, Raspberry Pi, Camera

# TABLE OF CONTENTS

## Contents

ACKNOWLEDGMENTS .....	i
ABSTRACT.....	ii
LIST OF FIGURES .....	vi
CHAPTER 1 INTRODUCTION .....	1
1.1 Problem Statement .....	1
1.2 Solution .....	1
1.3 Goals.....	1
1.4 Objectives.....	2
1.5 Conclusion.....	2
2.1 Slug Types in Crops .....	3
2.1.1 Slugs in Europe and their Types in Crops .....	3
2.1.2 Distribution .....	3
2.1.3 Identification.....	3
2.1.4 Life Cycle.....	4
2.1.5 Risk Period.....	4
2.1.6 Damage .....	4
2.1.7 Monitoring .....	4
2.1.8 Action Level.....	5
2.2 In-Market Competitors.....	5
2.2.1 SlugBot (CHAP).....	5
2.2.2 MSR-bot-Project: University of Kassel .....	6
2.2.3 Slugbot “9000”.....	7
2.3 Target Market and Lead Users .....	8
2.3.1 Target Market.....	8
2.3.2 Lead Users .....	9
2.4 End-Product Requirements .....	9
2.4.1 Non-Technical Requirements .....	9
2.4.2 Technical Requirements.....	10
2.5 Chosen Specifications .....	10
2.6 Our Approach.....	11
2.6.1 Navigation Approach.....	11

2.6.2	Detection Approach .....	15
2.6.3	Drivetrain Approach .....	17
2.6.4	Collection Approach .....	19
CHAPTER 3	DESIGN.....	25
3.1	Drivetrain Design .....	25
3.2	Weight of the Robot .....	27
3.3	Motor Selection .....	28
3.3.1	Motor Selection Criteria .....	28
3.3.2	Speed and Torque Formulas .....	29
3.4	Collection Mechanism.....	29
3.4.1	First Iteration.....	32
3.4.2	Second Iteration .....	32
3.4.3	Third Iteration .....	32
3.4.4	Integration with Drivetrain .....	33
3.5	Battery Sizing.....	34
3.5.1	Motor Selection Criteria .....	34
3.5.2	Based on Operational Time .....	34
3.5.3	Battery C Rating .....	35
CHAPTER 4	HARDWARE .....	36
4.1	Drivetrain .....	36
4.1.1	Choosing Aluminum Profile Size .....	36
4.1.2	Cutting Aluminum Profiles.....	37
4.1.3	Chassis Attaching Process .....	37
4.1.4	Cutting Wooden Plates .....	38
4.1.5	Fixing Motors.....	39
4.1.6	Adding Wheels.....	39
4.1.7	Components on the Lower Plate .....	40
4.1.8	Components on the Upper Plate .....	41
4.2	Motors and Motor Drives Selection for the Drivetrain .....	41
4.2.1	Motors .....	41
4.2.2	Motor Drives .....	43
4.3	Sensors .....	43
4.3.1	Raspberry Pi Camera Module V2 .....	43
4.3.2	Marvelmind Indoor GPS Kit.....	44

4.3.3	MPU-6050.....	45
4.4	Collection Mechanism.....	46
4.4.1	Motor for Rotating the Mechanism.....	46
4.4.2	Motor for Lifting the Mechanism Radially.....	46
4.5	Bill of Materials .....	47
4.6	Wiring Diagram.....	49
CHAPTER 5	IMPLEMENTATION AND RESULTS .....	51
5.1	Navigation Implementation.....	51
5.1.1	Mapping .....	51
5.1.2	Localization (GPS+IMU).....	52
5.1.3	Path Planning .....	52
5.2	Detection Implementation .....	52
5.2.1	Dataset Collection.....	53
5.2.2	Data LabelImg .....	54
5.2.3	Train Our Custom YOLOv5 Model.....	54
5.2.4	Run Inference with Trained Weights.....	54
5.2.5	Sending Commands to the Robot .....	55
5.3	Collection Mechanism Implementation .....	55
5.4	System Integration Implementation .....	55
CHAPTER 6	CONCLUSION AND FUTURE WORK.....	58
6.1	Future Work in Navigation .....	58
6.2	Future Work in Slug Detection .....	58
6.3	Future Work in Slug Collection .....	58
APPENDIX A	NAVIGATION TUTORIALS .....	60
APPENDIX B	DETECTION TUTORIALS .....	84
APPENDIX C	SURVEY .....	102
APPENDIX D	CE CERTIFICATE .....	105
APPENDIX E	STANDARDS .....	106
APPENDIX F	MEETING MINUTES.....	113
References	.....	127

## LIST OF FIGURES

Figure 1: Slug Types in European Backyards.....	3
Figure 2: SlugBot (CHAP).....	5
Figure 3: MSR-bot.....	6
Figure 4: Slug Density Approach .....	6
Figure 5: Slugbot " 9000" .....	7
Figure 6: Shares' Percentages in European Countries.....	9
Figure 7: 1 meter x 1 meter Occupancy Grid Map (Addison, 2021).....	13
Figure 8: Marvelmind GPS Basic Technology.....	14
Figure 9: Vision Path .....	17
Figure 10: Sketch of Roller with Conveyer Belt .....	19
Figure 11: Sketch of Suction Tube with a Gripper .....	20
Figure 12: Sketch of Two Rollers.....	20
Figure 13: Sketch of Vertical Gripper with a Drawer V1.....	21
Figure 14: Sketch of Vertical Gripper with a Drawer V2.....	21
Figure 15: Sketch of Horizontal Gripper .....	22
Figure 16: Scara Mechanism .....	22
Figure 17: Bulldozer-front storage.....	22
Figure 18: Sketch of Stick Collection Mechanism .....	23
Figure 19: Side-Back View of the Robot CAD's Design.....	25
Figure 20: Back View of the Robot CAD's Design .....	25
Figure 21: Side-Front View of the Robot's CAD Design .....	26
Figure 22: Top View of the Robot's CAD Design.....	26
Figure 23: Robot's Lower Part Dimensions .....	26
Figure 24: Robot's Upper Part Dimensions .....	27
Figure 25: CAD of First Iterations of Collection Mechanism .....	30
Figure 26: Design of First Iteration .....	32
Figure 27: Design of Second Iteration .....	32
Figure 28: Design of Third Iteration.....	33
Figure 29: Collection Mechanism and Drivetrain Design .....	33
Figure 30: Robot's Width and Length.....	36



Figure 31: 3030 Aluminum Profile Pattern Used .....	37
Figure 32: Aluminum Profiles cutting Machine at Raymond Feghali Co .....	37
Figure 33: Robot Chassis .....	38
Figure 34: T-NUT, WASHER, AND SCREW ASSEMBLY .....	38
Figure 35: Wooden Plate Used .....	39
Figure 36: Robot Chassis with the Wooden Plate and Motor Brackets.....	39
Figure 37: Chosen Wheel.....	40
Figure 38: Motor Coupling .....	40
Figure 39: Robot Chassis with Motors and Wheels.....	40
Figure 40: 37GB520 Motor Specifications (NOSTOP ELECTRIC CO.).....	42
Figure 41: Motor Specifications at our Operating Point using Desmos Graphing Calculator .....	42
Figure 42: 37GB520 Chosen Motor .....	43
Figure 43: L298N Motor Driver (Last Minute Engineers, 2021) .....	43
Figure 44: Raspberry Pi Camera Module V2 .....	44
Figure 45: Marvelmind Indoor GPS Kit .....	45
Figure 46: MPU-6050 .....	46
Figure 47: Motor for Rotating the Mechanism .....	46
Figure 48: Servo Motor for Lifting the Mechanism .....	47
Figure 49: Wiring Diagram.....	50
Figure 50: Map Yaml file .....	51
Figure 51: Map Launch File .....	52
Figure 52: Four Sample Images from the Dataset .....	53
Figure 53: YOLOv5 Model Types.....	54
Figure 54: Twist_MUX (NickLamprianidis, 2018).....	56
Figure 55: Priority of Topics in MUX Yaml Files .....	57
Figure 56: Downloading and Installing the Dashboard Driver.....	60
Figure 57: Downloading SW Pack .....	60
Figure 58: Dashboard API in Directory.....	60
Figure 59: Access Serial Port.....	61
Figure 60: Modem Firmware .....	61
Figure 61: Marvelmind Dashboard after Connecting all the Beacons.....	62

Figure 62: Marvelmind Origin Selection in the Dashboard.....	62
Figure 63: Marvelmind Dashboard of Frozen Map .....	63
Figure 64: Accelerometer Calibration .....	63
Figure 65: High Performance of the IMU .....	64
Figure 66: ROS - Workspace .....	64
Figure 67: ROS - hedge_rcv_bin .....	65
Figure 68: ROS - Subscriber_test .....	66
Figure 69: Rviz - Stationary and Mobile Beacons Markers .....	67
Figure 70: ROS - Run gps2odom Package .....	67
Figure 71: ROS - Frames in rqt tree.....	68
Figure 72: Rviz - Frames TF.....	68
Figure 73: ROS - Odometry Received.....	69
Figure 374: Rviz - Demo Map With Frames .....	70
Figure 75: ROS - Cmd_vel Echo .....	71
Figure 76: Rviz - Frames Movement .....	71
Figure 77: Ubuntu bashrc.....	72
Figure 78: Raspberry Pi bashrc.....	72
Figure 79: ROS - Catkin_make Install Command.....	73
Figure 80: ROS - Run Marvelmind Package .....	74
Figure 81: ROS - Messages Output from hedge_imu_fusion Node .....	74
Figure 82: ROS - Turtlebot3 Bringup.....	75
Figure 83: Parameters of Local Costmap Yaml File .....	75
Figure 84: Parameters of Global Costmap Yaml File.....	76
Figure 85: Parameters of move_base Launch File.....	76
Figure 86: Rqt Tree.....	77
Figure 87: ROS - Run GPS2ODOM Python Code.....	77
Figure 88: ROS - Launch Turtlebot3 Navigation Launch file .....	77
Figure 89: ROS - Goal Reached Command Line Indication .....	78
Figure 90: Speed Parameters of dwa_local_planner_params_burger Yaml File.....	78
Figure 91: Send Goal Through Python File .....	79
Figure 92: Roscore of Ubuntu.....	80

Figure 93: Source bashrc and Workspace.....	80
Figure 94: Rosserial .....	81
Figure 95: Launch the turtlebot3_teleop.....	81
Figure 96: Run rqt_graph.....	82
Figure 97: Node Graph .....	82
Figure 98: LabelImg Download Zip File .....	84
Figure 99: Creating Virtual Environment .....	85
Figure 100: Virtual Environment Activation.....	85
Figure 101: Navigating Using Ubuntu Terminal .....	86
Figure 102: Installation Steps .....	86
Figure 103: LabelImg Program Interface .....	87
Figure 104: LabelImg Annotation Process .....	87
Figure 105: Resulted Files From Annotation Process .....	88
Figure 106: Bounding Box Coordinates of a Slug and Class Number .....	88
Figure 107: YOLOV5 in Google Colab.....	89
Figure 108: Cloning the YOLOV5 Repository and Importing Necessary Libraries .....	89
Figure 109: Train.py File with Input Arguments.....	90
Figure 110: Tensorboard Graphs .....	90
Figure 111: Detect.py Arguments for Weights and Image Source.....	91
Figure 112: Detect.py Arguments Passed in the Code Cells .....	91
Figure 113: Testing the Model on a Slug Image and Extracting the Boundary Boxes Coordinates .....	92
Figure 114: Testing the Inference on the Webcam of the Laptop .....	92
Figure 115: Inference Results .....	93
Figure 116: Slug Centroid is in the Left ROI and cmd_vel Value Turns the Robot to the Left.....	95
Figure 117: Slug Centroid is in the Right ROI and cmd_vel Value turns the robot to the right ....	95
Figure 118: Slug Centroid is in the Forward ROI and cmd_vel Value Moves Forward the Robot	96
Figure 119: Slug Centroid is in the Collection ROI and cmd_vel Values Stops the Robot and Starts Collecricion Process .....	97
Figure 120: OpenCV-ROS Interface .....	98
Figure 121: Turtlebot3 Commands for Camera Bringups and Image Uncompression Test #1 ...	100

Figure 122: Codes to Use to Convert Frame to Image and Starting Inference using RPI Camera ..... 100

Figure 123: Test #1 on Turtlebot ..... 101

## LIST OF TABLES

Table 1: Competitors' Comparison Table .....	8
Table 2: Specifications.....	11
Table 3: Sketches' Comparison.....	24
Table 4: Total Weight of the Robot (Kg).....	27
Table 5: Chassis' Weight of Different Aluminum Profiles .....	28
Table 6: Criteria of Motor Selection.....	29
Table 7: Results of Collection Mechanism Testing.....	31
Table 8: Battery Sizing .....	34
Table 9: Power-Bank Sizing .....	34
Table 10: Battery Sizing - Operational Time.....	34
Table 11: Length and Width of needed Aluminum Profile .....	36
Table 12: Aluminum Profiles Sizes and Quantities .....	37
Table 13: Mini-TX GPS Beacon Specifications (Swaidan, Baradhi , Mechlawi , & Yahya, 2021) .....	45
Table 14: Bill of Materials.....	48

# CHAPTER 1

## INTRODUCTION

Agricultural robots automate slow, repetitive, and dull tasks for farmers, allowing them to focus more on improving overall production yields. Some of the most common robots in agriculture are used for harvesting and picking. Our senior project is an agricultural robot of a different kind, where it aims to detect, collect, and store slugs while autonomously navigating in home's backyards. The project solves a major problem in agriculture which is protecting plants from slug's damage.

The upcoming sections includes the problem statement, the solution, along with the goals and objectives, followed by a conclusion that describes the report's content.

### **1.1 Problem Statement**

It may be a surprise, but slugs do cause harm for plants in backyards. Slugs eat a wide variety of broadleaf plants and grasses, including most crops and many weeds. They harm crops by killing seedlings outright, causing poor stands, and by damaging leaves on young plants. They feed by scraping the surface of their food, which can include seeds, roots, stems, and leaves. Snails and slugs are found in greatest abundance after rains and after the plants are watered. Therefore, slugs cause a major problem for farmers and for backyards owners.

### **1.2 Solution**

The robot discussed in this paper serves as a solution for any person who plant crops in private backyards. It aims to remove slugs from ground before reaching the plant and dispose it in the robot's storage without harming the slug.

### **1.3 Goals**

By the end of the senior year, the team's target is to achieve the following goals:

- Design and realize an autonomous robot able to navigate on a backyard's rough agriculture terrain.
- Integrate an on-board camera to detect slugs using Artificial Intelligence.

- On-board collection mechanism that's able to collect the detected slugs without harming them.
- Store safely the detected slugs in an integrated storage during the collection process.

#### **1.4 Objectives**

Team's objectives to be achieved successfully by the end of the senior year are the following:

- Presenting a well-functioning prototype.
- Slug detection through Artificial Intelligence.
- Collecting and storing slugs safely.
- Autonomous navigation on a rough terrain.

#### **1.5 Conclusion**

In this chapter, we discussed the general idea of the project, the problem and its solution. Also, the team's goals and objects, that should be achieved by the end of the senior year, were stated.

The target of this report is to explain deeply all the details of the project, so six chapters are well detailed to show the whole process that we went through to reach our goal. Thoroughly, chapter one includes an introduction that gives an overview of the project. Then, the literature review, in chapter two, states all the requirements and expected output, along with the approach that we will follow to reach our target. In addition, chapter three includes all the calculations that lead to our chosen design, and chapter four shows the process followed to choose the hardware components used. Afterwards, in chapter five the results of the implementation are stated clearly. Finally, the future work that can be applied to the project is stated in chapter six.

## CHAPTER 2

### LITERATURE REVIEW

Robots, agricultural robots in specific, have undergone significant advancement and recently robots are highly integrated in agriculture to solve various problems. In our case, a slug picking robot is discussed to solve a major agricultural problem, which is creating an ultimate solution for the crops damage caused by the slugs. Our target market is backyard owners in Europe, so all the following sections will be discussed accordingly.

#### 2.1 Slug Types in Crops

##### 2.1.1 *Slugs in Europe and their Types in Crops*

Black-keeled slugs and reticulated slugs.

##### 2.1.2 *Distribution*

Originally from Europe, both species are now present in the Mediterranean basin, North America, South America, New Zealand, and Australia. In Australia, both species are mainly pests in the high rainfall (> 500 mm pa) areas of the cropping zone. Pest status on canola: Major, restricted, regular.

##### 2.1.3 *Identification*

The reticulated slug is colored light grey to fawn with dark brown mottling and grows to 4 cm long. The keel of the reticulated slug is only present at the posterior end of the body. The black-keeled slug is uniform grey to black in color and grows to 5 cm. The keel on the black-keeled slug runs from the tip of the tail to the mantle. (Ed., 2007)



Figure 1: Slug Types in European Backyards



The reticulated slug (upper; 3 cm long), showing brown mottling and milky slime and the black-keeled slug (lower) with uniform dark coloration.

#### **2.1.4 *Life Cycle***

In Europe, the life cycle of the reticulated slug is completed in less than 1 year (Ed., 2007). Slugs, which are hermaphrodites, lay eggs in the soil, probably during autumn. Heavy soils, particularly those which crack and allow movement and protection of slugs, are favorable to slug populations. When surface moisture is available, slugs move on the surface at night and shelter under litter etc. during daytime. As surface moisture dries, slugs may move to 20 cm deep in the soil. The black-keeled slug is probably more of a burrowing species than the reticulated slug. Summer rain promotes build-up of numbers for the following autumn. Slugs eat by rasping plant tissue. Slug-feeding on canola is characterized by serrated edges to cotyledons. Reduced tillage and greater stubble retention of conservation farming are allowing more slugs to survive and breed.

#### **2.1.5 *Risk Period***

From emergence of the crop until about 2 months afterwards.

#### **2.1.6 *Damage***

Damage to cotyledons, death of seedlings and reduction of leaf area of establishing plants may occur. Once seedling cotyledons are destroyed, the plants will not recover. Damage often goes undetected as the crop is emerging, and the resultant poor establishment is incorrectly put down to agronomic factors. Damage may be particularly marked along weedy boundaries and may be more common in higher rainfall districts, but extensive damage can occur in drier areas and in the center of crops. Damage from European earwig, false wireworms and even the lucerne flea can be falsely attributed to slugs. Determining the true cause of damage is essential for achieving good control.

#### **2.1.7 *Monitoring***

Pre-sowing slug occurrence can be detected by setting bait traps when the soil surface is moist. Bait traps can be made from either hessian bags or upturned 20 cm diameter flowerpot bases with gaps to allow entry of slugs, under which are placed several bait pellets.

### **2.1.8 Action Level**

If slugs are caught under bait traps, baiting of the crop is recommended prior to or at seeding. (Ed., 2007)

Snails and slugs are most active at night and on cloudy or foggy days. On sunny days, they seek hiding places out of the heat and bright light. Often the only clues to their presence are their silvery trails and plant damage. all places where they can hide during the day. Boards, stones, debris, weedy areas around tree trunks, leafy branches growing close to the ground, and dense ground covers, such as ivy, are ideal sheltering spots.

During cold weather, snails and slugs hibernate in the topsoil. In areas with mild winters, such as southern coastal locations, snails and slugs can be active throughout the year. During hot, dry periods snails estivate (hibernation during hot weather) by sealing themselves off with a parchment-like membrane. They often attach themselves to tree trunks, fences, or walls. (Ed., 2007)

## **2.2 In-Market Competitors**

After a deep market study, we figured out some competitors that present similar functionalities and technologies. However, our product differs from them because it's the only robot designed for home backyards (not for big farms) and collects slugs in a safe way.

### **2.2.1 SlugBot (CHAP)**



Figure 2: SlugBot (CHAP)

The SlugBot project is a collaboration with agricultural robotics start-up Small Robot Company (SRC), cutting-edge AI experts COSMONiO and farming enterprise AV and N Lee.

The technical development of this robot started in 2020, with Phase one focusing on developing the artificial intelligence slug detection capability Phase two will look to deliver slug detection using Small Robot Company’s ‘Tom’ robot, with mobile imaging of slugs and field-surface materials in glasshouse conditions anticipated by early 2021. Phase three will then focus on precision spraying, delivering an in-field slug treatment solution for autumn 2021.(Hutton, n.d.)

### 2.2.2 MSR-bot-Project: University of Kassel



Figure 3: MSR-bot

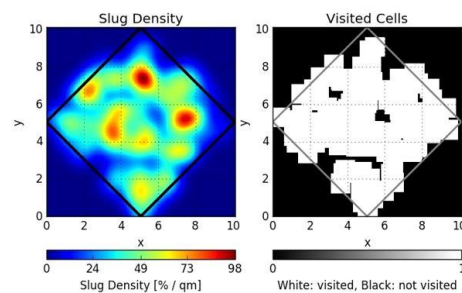


Figure 4: Slug Density Approach

The project started in 2016, and in the first year of the project a vehicle platform and a sensor for slug detection based on a camera system and digital image processing will be developed. In the second year of the project the slug detection sensor will be used in a field test, to estimate the quantitative behavior of slugs in relation to different influencing factors, as weather, soil conditions or the pest

control strategy. Also, a navigation unit will be added to the robot and a robot control will be developed. The robot control shall contain a hotspot focusing method to improve the impact power of the robot. Later the gained knowledge of the quantitative slug behavior will be used to improve the robot control. The robot will be equipped with manipulators to fight the slugs it found. In the third year of the project the developed modules will be assembled to a working robot and the functionality of the system will be proved. It will also be checked if it is possible to use the robot for the abatement of other pests like mice for instance. (Hensel, 2016)

### 2.2.3 *Slugbot "9000"*

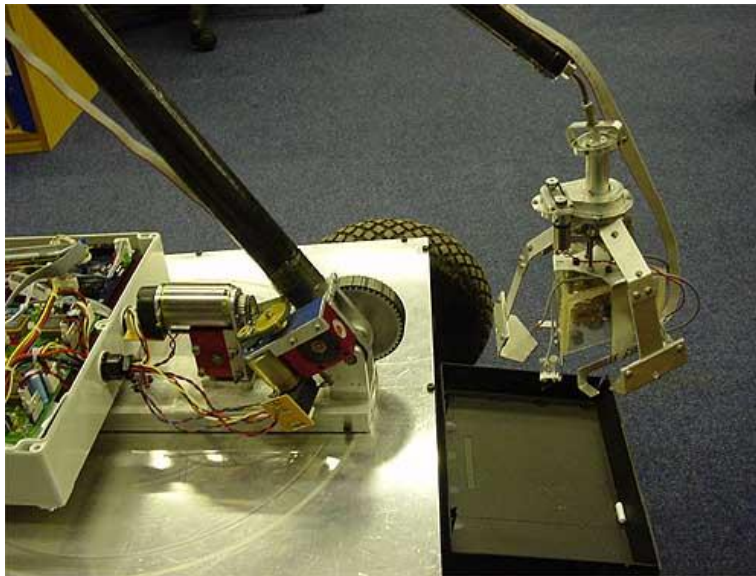


Figure 5: Slugbot " 9000"

Built in 2001 at the University of West England, SlugBot uses a vision sensor and a 360 extending arm to find slugs, grab them, and drop them into an onboard trap. (Wearden, 2002)

Table 1 below, shows a comparison between all the competitors of our Slug Picking Robot

<b>Company Name</b>	CHAP, Small Robot, COSMONiO and AV and N Lee (Hutton, n.d.)	University of Kassel, KommTek and Julius Kühn Institut (Hensel, 2016)	University of west England (Wearden, 2002)
<b>Year</b>	2020	2016	2001
<b>Slug Disposal Method</b>	Special spray to kill slugs	Knife down slugs on-site using nails	Use microbial fuel cells to convert slugs into biomass and generate electricity
<b>Cost</b>	NA	NA	1500\$
<b>Working Space</b>	Big Farms	Big Farms	Big farms
<b>Development Phase</b>	Under Development (Field Testing)	Under Development (Field Testing)	The last info released was in 2013

Table 1: Competitors' Comparison Table

## 2.3 Target Market and Lead Users

### 2.3.1 Target Market

Cropland and Farms (the term agricultural holdings are used sometimes), are areas of land that are made dedicated for agricultural process to produce food and other crops.

The number of farms in the world is approximately 570 million today, with majority of family-owned lands and businesses. Family-operated farms having the size of Medium to Large Croplands are the main target of this business plan

Mainly in Europe, the problem has been reported a lot in UK and Germany among other European countries.

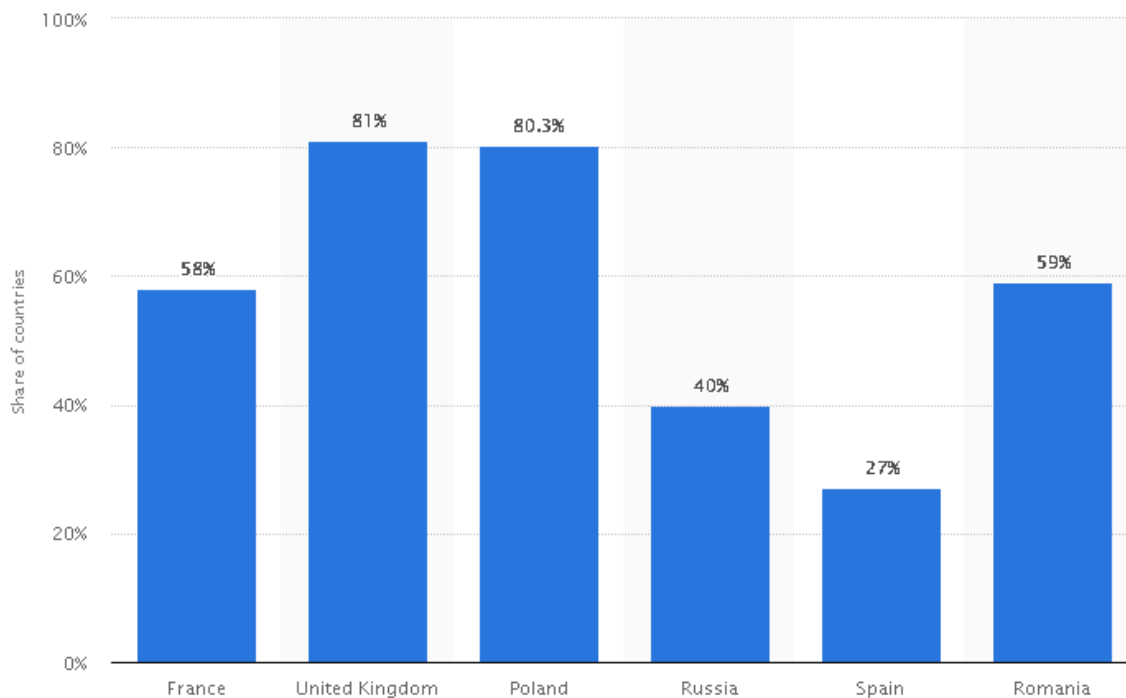


Figure 6: Shares' Percentages in European Countries

### 2.3.2 *Lead Users*

According to the survey we conducted, the lead users in this project are the owners of households with backyards containing plants that are considered slug's favorite, such as:

- Salads (Cabbage, Tomatoes, Pumpkins...etc)
- Strawberries
- Flowers in general
- Sunflowers
- Corn

In addition to that, users who live in high humidity areas, or foggy country sides.

## 2.4 End-Product Requirements

### 2.4.1 *Non-Technical Requirements*

- CE Certified

- IP Grade for rainy, outdoor, muddy/wet environments (The higher the better)
- Environment friendly (Not harm the slugs/plants...)
- Children friendly
- Low noise
- Other requirements (can be collected from customer needs)

#### 2.4.2 Technical Requirements

- The robot should navigate in muddy/wet environments
  - The navigation system should be modular (adapt to different operating areas)
  - The robot should localize itself inside the operating area
  - The robot should navigate/scan the whole operating area
  - The robot should avoid the plants/obstacles
  - The robot should have safety measures
- The robot should detect slugs
  - The slugs should be detected on the ground (as a start)
  - Any camera/sensor type can be used (RGB, Thermal cameras...)
- The robot should collect slugs: (The slugs shouldn't be harmed; the robot should collect several slugs then go back to release them in a specified location)
- The robot design/chassis can be optimized based on the used hardware later

#### 2.5 Specifications

Based on the survey and the required specifications; a set of specifications for the final product and the Proof of Concept prototype were developed as in the following table:

Specifications	Metric (Final Product)	Metric (POC)	Unit	Imp
Operation time (battery)	70	45	Minutes	5
Area of backyard (m <sup>2</sup> )	3 x 4	3 x 4	m <sup>2</sup>	
Weight without slugs	2	6	KG	

L x W x H	15 x 22 x 10	28 x 23 x 31	Inches/cm	
Storage capacity (target number of collected slugs)	10 to 15	10	Slugs	
Drivetrain	Can navigate on muddy and rough terrains	Can navigate on muddy and rough terrains		
Low Noise	60 (daytime) and 45 (nighttime)	-	dB(A)	4
Weather resistance	IP68 - IK10	-	N/A	5
Types of Crops in the targeted backyard	Salads (Cabbage, Tomatoes, Pumpkins...etc.)	-		
Collection (speed/time of collecting it after detection without harming)	From experimentation	-		
Navigation	100% of the field covered	-		
Disposal	0 Killed Slugs	0 Killed Slugs		
Automatic recharging (recharge time)	Goes to charging station when battery is low	-		

Table 2: Specifications

## 2.6 Our Approach

In this report, we studied, designed, and implemented different concepts through different phases. After several concept generation sessions with our advisors, we all agreed to split the project into 4 technical aspects: Navigation, Detection, Drivetrain and Collection. In each aspect, a unique approach was followed by us, after a deep and detailed research.

### 2.6.1 Navigation Approach

Autonomous agricultural robots have undergone rapid advancement in communication, sensors, and computing technologies. To ensure the autonomy of the agricultural robot, the advanced features of Robot Operating System (ROS) will be utilized. (N.Shalal & Low). The navigation systems in agricultural environments consists of designing appropriate systems for mapping, localization, path planning, and obstacle avoidance.

- **Mapping**



Mapping is the process of transforming the physical environment where the robot will navigate into a form of a digital map that the robot can process and understand.

And this can be achieved by several techniques based on different sensors. For example, “Gmapping” is a mapping approach that is based on laser scan data coming from a Lidar and the odometry data to create a map. Another approach is the “RTAB Mapping” which is a visual mapping approach based on stereo and depth cameras such as the one found in the “Microsoft Kinect” sensor. Moreover, a map can be created virtually without reading sensor data using software techniques that generate the map as an image. The generated image represents an occupancy grid. (Swaidan, Baradhi , Mechlawi , & Yahya, 2021)

The occupancy grid is a 2D grid map of pixels, similar to the map shown in figure 7. These pixels represent physical distance according to a specified resolution that defines how many meters each pixel represents (meter/pixel). The pixels in the ROS occupancy grid can have 3 colors: (Addison, 2021)

- **Black:** a black pixel represents an obstacle in the 2D plane. The black space is avoided later on by the path planners in ROS.
- **White:** a white pixel represents free space with no obstacles. The white space is where the planner plans the optimal path later on.
- **Gray:** a gray pixel represents an unknown/undiscovered space. The gray space is usually avoided by the planner. If it was obliged to plan inside a gray space, the planner plans short paths with slow speed to discover the unknown environment and map it safely.

In our case, we'll draw the map of the working environment, using a software called GIMP, to get all the static obstacles as an Occupancy Grid Map.

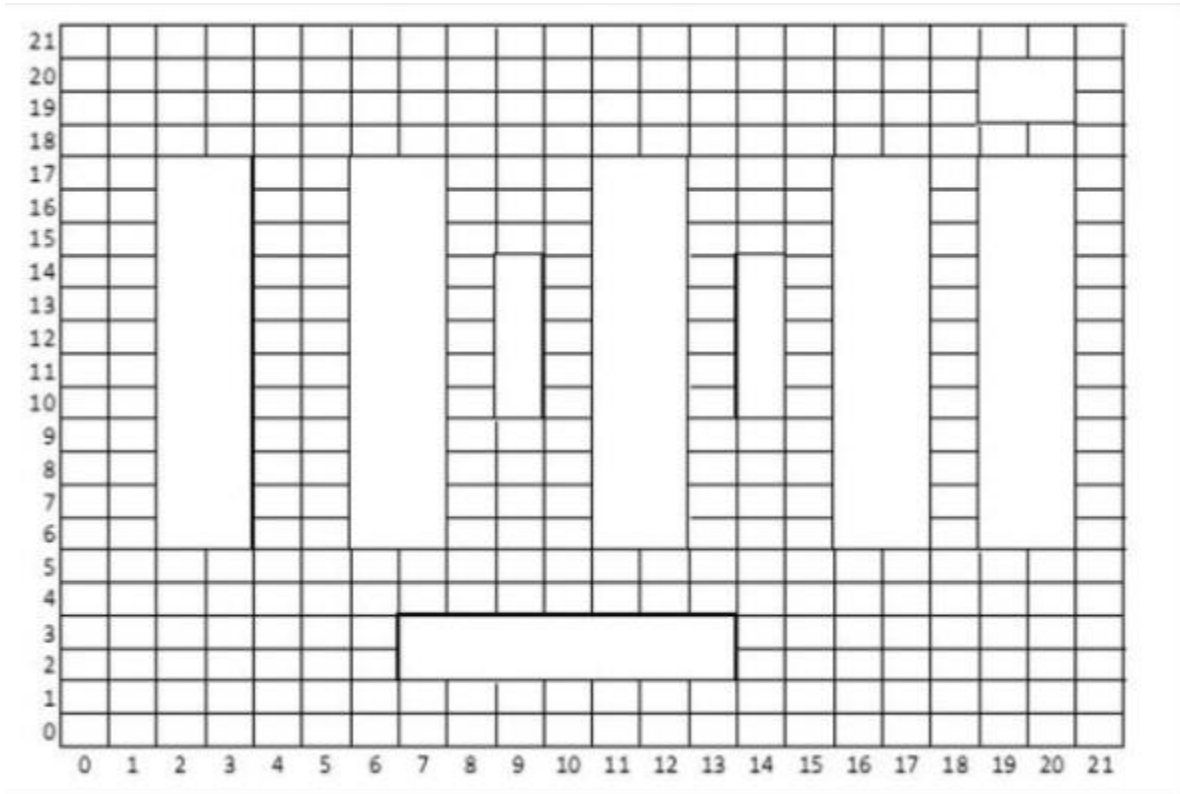


Figure 7: Example of a 1 meter x 1 meter Occupancy Grid Map (Addison, 2021)

- **Localization (GPS+IMU)**

Marvelmind Indoor GPS System is an indoor localization system designed to provide precise and real-time location updates of autonomous robots, vehicles (AGV), and copters (UAV) with a precision of  $\pm 2\text{cm}$ . It is implemented via mobile beacons attached to the object to be tracked. The navigation system consists of a network of stationary beacons having 5 ultrasonic transducers interconnected in a license-free band via a wireless radio interface, one or more mobile beacons mounted on a moving object, and a modem handling the wireless communication between the whole system. The modem can be connected to a PC to configure the network, optimize the data, and visualize it. (Introducing Marvelmind Ultrasonic Indoor Navigation

System, 2020) The device will automatically create a map of stationary beacons without any additional manual data input or any manual distance measurements for simple cases. Once the map is formed, it can be frozen and stored in the memory of the modem. So, the localization can be then done without connecting any PC to the modem and the data of the localization system is tethered wirelessly through the modem to be interfaced with any other system. In our case, the interfacing will be done with ROS through python. The network architecture is summarized in figure 8. (Swaidan, Baradhi , Mechlawi , & Yahya, 2021)

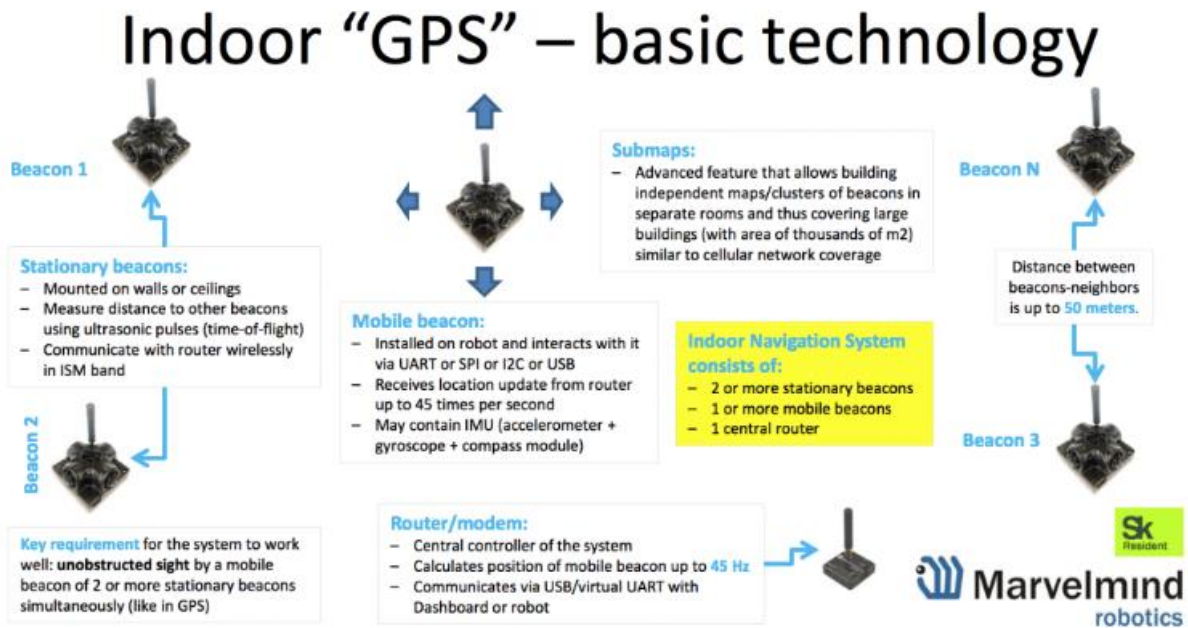


Figure 8: Marvelmind GPS Basic Technology

- **Obstacle Avoidance**

After having a map (occupancy grid) for the navigation environment and precisely knowing the location of the robot inside this map, 2 maps are derived from the main one, a “Global Costmap” and a “Local Costmap”. Both maps have several parameters to be specified and the most important ones are the size and whether it is a static/dynamic map. The global costmap has the same size as the main map of the environment and it is static. This means that the map doesn’t change and never gets updated. It represents the static features in the

navigation environment such as walls. This map is used later on for global path planning. The local costmap is another small map that is a portion from the main map that moves with the robot keeping the robot centered in it. Its size can be chosen according to the robot size. Also, the local costmap is a dynamic map that gets updated by the new obstacles surrounding the robot in it. The obstacle detection can be done using several sensors ranging from simple IR sensors to cameras. Once the obstacles are detected and put inside the local costmap, the path planner, later on, will take them into consideration and commands the car to avoid these obstacles in real-time. (Swaidan, Baradhi , Mechlawi , & Yahya, 2021)

- **Path Planning**

The goal behind path planning is finding a sequence of maneuvers that the robot should execute to move from a starting position to a desired goal position avoiding obstacles along the way. Examples of popular path planning/finding algorithms are the “Dijkstra’s algorithm” and the “A\* algorithm”. (Husarion Docs , n.d.)

Path planning algorithms are based on the occupancy grids (costmaps) discussed in the previous section. The path planner divides the costmap into cells (pixels) and assigns to them a label according to their color whether they are free or occupied. Then, after labeling the cells, the shortest path along the free pixels is selected and sent to the robot to be followed. (Swaidan, Baradhi , Mechlawi , & Yahya, 2021)

In our case, initially the goal will be sent to the robot through Rviz by the user and the robot choose the best path to reach this goal.

### **2.6.2 Detection Approach**

Equipping robots with vision increase their ability of sensing the surrounding environment and extend their visual capabilities to achieve more complex tasks. For example, it helps to detect and local static, as well as dynamic bodies around the robot. This could help in navigation, grasping, recognition problems and much more.

Thus, the main purpose of robot vision is to attach a visual ability to sense, think, understand, and react to the surrounding of the robot.

Vision is done mainly using cameras that send visual data to the processor to process them and extract the needed data. The camera to use may be a RGB or RGB-D camera. RGB (Red - Green - Blue) cameras are cameras that are equipped with a standard CMOS sensor through which the colored images of persons and objects are acquired (Khosrow-Pour, 2017)

A visual camera sensor is an image processor that gathers visible light (400-700nm), transforms it to an electrical signal, and then arranges encode it to return images and video frames." These cameras use light visible wavelengths that ranges from 400 to 700 nanometers, the exact range that the human eye can visualize. Visible cameras capture red, green, and blue light wavelengths (RGB) and output an high quality color representation, and are meant to generate images that mimic human vision. (Infiniti, n.d.)

- **RGB and RGB-D Comparison:**

RGB-D cameras are lately becoming more appealing in robotics applications and dramatically improved their performance as they not only provide visual data, but also depth information in a per-pixel basis. This extra depth info provides the ability to estimate the pose and depth using image-based algorithms and create 3D scene. (Jacob, Menon, & Joseph , 2020)

For our application we want to build a cost-efficient solution with the least possible complexity. RGBD cameras are great for such applications but cost more than RGB. Moreover, the extra depth in the RGBD cameras requires more energy to operate. Thus, RGB camera is most suitable for our proof of concept.

We will be using raspberry pi RGB camera v2 due to its cost, local market availability and ease of adaptability to ROS.

This will be our path to implement the vision system of our robot:

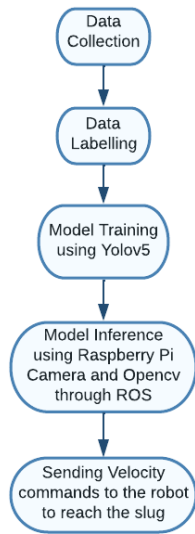


Figure 9: Vision Path

For the model, we will use a pretrained YOLOv5 model and train it on our custom dataset. “YOLOv5 is a family of compound-scaled object detection models trained on the COCO dataset and includes simple functionality for Test Time Augmentation (TTA), model ensembling, hyperparameter evolution, and export to ONNX, CoreML and TFLite. “ (PyTorch, n.d.). It applies a real time object detector and has the feature of returning an annotated image with a boundary box around the needed object to detect, slugs in our case, along with the coordinates of that object. This gives the robot the ability to not only detect slugs, but also to know its location in the input image or frame.

Detailed information about the process can be found in chapter 5.

### 2.6.3 *Drivetrain Approach*

In our case, the robot shall navigate, detect and collect slugs in a rough, agricultural terrain.

To serve this target, our research showed that there are two possible options, either the use of continuous tracks or four wheels.

- **Pros of Continuous Tracks**
  - Ground impact: a robot that moves on rubber tracks has a lower PSI on the ground. That means a less impact on the ground, especially when the robot is heavy (introrobotics , 2013)
  - Weight growth potential: a robot with continuous tracks has a weight spread over the entire surface of the track. This is one of the reasons that a robot with rubber tracks support a heavy load (introrobotics , 2013)
- **Cons of Continuous Tracks**
  - Lower speed: due to more friction and a complex mechanical system, the robots with continuous tracks has lower speed compared with robots on wheels (introrobotics , 2013)
  - Less maneuverability: robots on tracks are less precise in maneuverability and require more power when turning (introrobotics , 2013)
  - Difficult to repair: the continuous tracks are difficult to repair or replace than wheels (introrobotics , 2013)
- **Pros of Four Wheels** (compared to tracks)
  - Lower production costs
  - Speed: Wheels need a lower amount of torque to move on from stationary. (introrobotics , 2013)
  - Maneuverability: the wheels provide high maneuverability. (introrobotics , 2013)
  - Lightweight: continuous tracks are much heavier than wheels, and this is the main reason why wheels are used especially in cases when the mass of the robot is a critical property (introrobotics , 2013)
  - Simplicity: a wheel has less moving parts, which means that there are fewer components that can get damaged (introrobotics , 2013)
- **Cons of Four Wheels**

- Drive over obstacles: depending on the terrain, if the robot needs to pass small or large obstacles. For a wheel to get over a vertical obstacle, it has to be at least twice as tall as the vertical obstacle. (introrobotics , 2013)

Considering all the advantages and disadvantages mentioned above, we chose to design a robot from scratch consisting of 4 wheels to accommodate with the rough, agricultural working environment.

#### 2.6.4 Collection Approach

The initial attributes required a robot to have a collection mechanism, designed to reach out and collect safely the targeted slug. The first choice was a robotic arm with a soft gripper, but studies resulted that the robotic arm isn't the cost-effective solution for the product. Therefore, a brainstorming session took place to decide on a feasible replacement. Many ideas were presented and show-cased with sketches for the stakeholders.

- **Roller with conveyer belt**

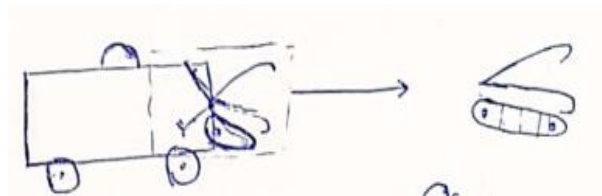


Figure 10: Sketch of Roller with Conveyer Belt

The collection mechanism of this robot, shown in figure 10 has two main parts, the first one is a roller that moves with an inclined motion, when the slug is detected, the robot approaches the slug, and the roller starts. The second part is a conveyer belt that helps the roller pick the slug from the ground to the storage, the main benefit of this mechanism is that it reduces the cost and the complexity of the system, as we can see in table 3 in this mechanism, there is no need for the depth coordinates from the camera so a simple RGB



camera can be used. On the other side, the main drawback of this system is that it collects dust more than the other mechanisms, but this can be controlled by the size of the roller.

- **Suction tube with a gripper**



Figure 11: Sketch of Suction Tube with a Gripper

This collection mechanism, as in figure 11, is separated also into two main parts the first one is a gripper with a vertical linear motion, this gripper is connected to a suction tube. When the robot detects a slug, it approaches it when the slug is positioned under the tube, then the gripper will catch the slug and elevate it to the suction mechanism where the slug will be sucked to the storage. This mechanism will require extra complexity and cost because of the suction mechanism and because of the precise positioning this robot requires an RGBD camera. require a RGBD camera.

- **Two Rollers**

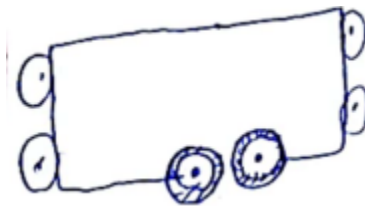


Figure 12: Sketch of Two Rollers

The mechanism in figure 12, is based on a simple concept, two rollers positioned in front of the robot with a small distance between them, when the slug is detected, the rollers will start turning in the opposite direction and when it reaches the slug it will suck it inside the storage. The main advantage of this mechanism is its

simplicity, because, thus its low cost as we can see in table 3 in this mechanism the camera used is an RGB camera and the material used to build this mechanism is low cost. The main drawback of this mechanism is that it has a bigger percentage of damaging slugs than the other mechanisms.

- **Vertical Gripper with a Drawer V1**

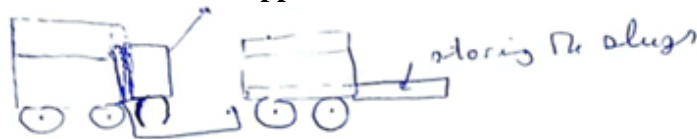


Figure 13: Sketch of Vertical Gripper with a Drawer V1

This mechanism of figure 13, is composed of a linear vertical motion with a gripper, when the slug is detected, the robot is positioned in front of the slug, then the gripper grasps the slug and elevates it, then a drawer will be opened horizontally, and the gripper will open to drop the slug inside the drawer. this mechanism has a high cost because it requires an RGBD camera and usually the price of the industrial grippers is also high, but in parallel, this mechanism has the highest precision.

- **Vertical Gripper with a drawer V2**

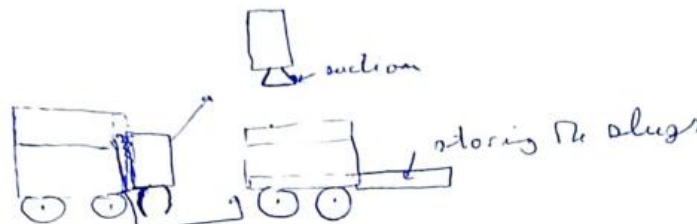


Figure 14: Sketch of Vertical Gripper with a Drawer V2

The mechanism in figure 14 is similar to the mechanism shown in figure 13, but the difference here is that the gripper is now replaced by a suction head.

- **Horizontal Gripper**



Figure 15: Sketch of Horizontal Gripper

This mechanism is similar to the mechanism in figure 14 but the difference here is that the gripper is now oriented horizontally, and the storage is now placed inside the robot, so when the gripper catches the slug it will enter inside the robot to drop the slug.

- **Scara Mechanism**

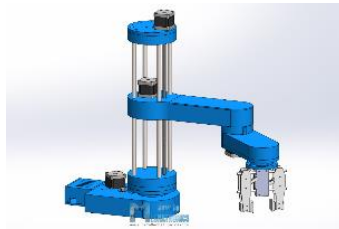


Figure 16: Scara Mechanism

The collection mechanism used in the robot of figure 16 is called the SCARA mechanism. The main benefit of this mechanism is that it will give us more freedom to catch the slugs, but it has a high cost because it will also need an RGBD camera, and it's a kind of a robotic arm so it's also costly.

- **Bulldozer-front storage**



Figure 17: Bulldozer-front storage

In this robot of figure 17, the collection mechanism is a bulldozer, and the storing mechanism is the same drawer used in mechanism 4, the advantages of this mechanism are that it has a medium cost, and it does not depend on the size of the slug, but at the same time the disadvantage of this mechanism is that it is the mechanism with the highest dust collection rate.

- **Bulldozer-back storage**

This mechanism is similar to the mechanism figure 17, but the difference here is that the bulldozer will dispose of the slug in the back of the robot rather than the drawer in front of the robot.

- **Stick Collection**

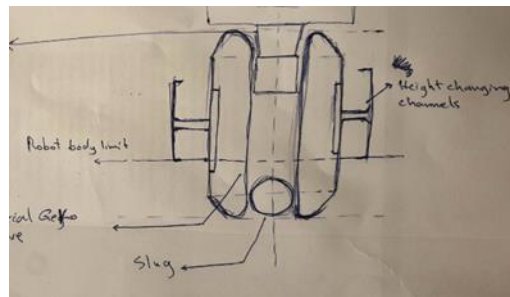


Figure 18: Sketch of Stick Collection Mechanism

In this proposed mechanism, the main idea was to use two rollers covered or made from a Geko like material. Geko is famous for being very sticky and able to pick-up any contacted object. The rollers are connected through a mechanism to the inside of the drivetrain and using motor the entire mechanism can be lifted and lowered. And the slug is removed and directed to the storage with an attached slider in the middle between the rollers. The disadvantage of the mechanism is collecting debris

After studying the brainstormed ideas, it was decided to test the roller solution to decide on its feasibility. The roller mechanism consists of a Roller, a Ramp and a Storage that will be discussed thoroughly in chapter 2.

Mechanism	Number of Motors	Gripper Type	Complexity	Accuracy	Dust	Camera	Slug Size Dependency	Price
Roller with a Conveyer Belt	3	Roller	3	2	2	RGB	1	4
Suction Tube with a Gripper	3	Suction + Clamp	3	2	2	RGBD	3	3
Two Rollers	3	Roller	2	1	2	RGB	3	2
Vertical Gripper with a Drawer V1	5	Clamp	2	3	1	RGB	1	3
Vertical Gripper with a Drawer V2	5	Suction	2	2	1	RGB	1	3
Horizontal Gripper	4	Clamp	2	2	1	RGB	1	3
Scara Mechanism	3	Clamp	4	3	1	RGBD	2	4
Bulldozer-Front Storage			2	1	3	RGB	1	2
Bulldozer-Back Storage			3	1	3	RGB	1	2
Stick Collection	2	Roller	4	2	2	RGB	2	5

Table 3: Sketches' Comparison

# CHAPTER 3

## DESIGN

### 3.1 Drivetrain Design

Before physically assembly our robot, drawing a Computer Aided Design (CAD) Model for it is an essential step. “CAD Design enables designers to see the 3D robotic model from various angles and check if they align with the geometric parameters provided. CAD allows the designer to make corrections with precision.” (CAD/CAM Services , 2022)

The availability of a CAD Model can help in applying different type of mechanical studies such as stress strain study. It can also help in case simulation is needed as it could be converted to URDF file and imported to a simulation software such as gazebo, webots etc.

After putting the guidelines for our robot design, we started sketching it on Solidworks. As a result, we got an accurate model of the robot with the needed dimensions. As a result, when assembly it, we knew beforehand all needed dimensions and components and the assembling was done without any main designing issue.

The CAD design can be found in figures 19,20,21 and 22:

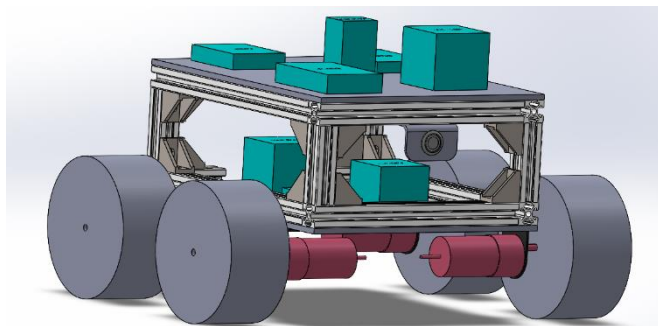


Figure 19: Side-Back View of the Robot CAD's Design

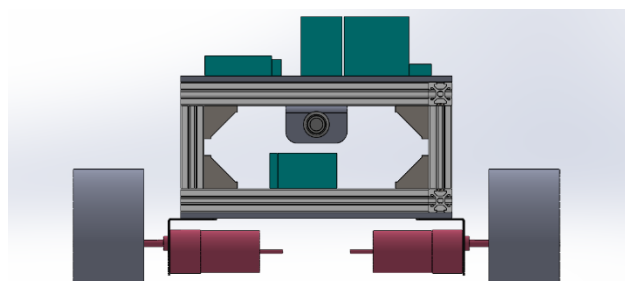


Figure 20: Back View of the Robot CAD's Design

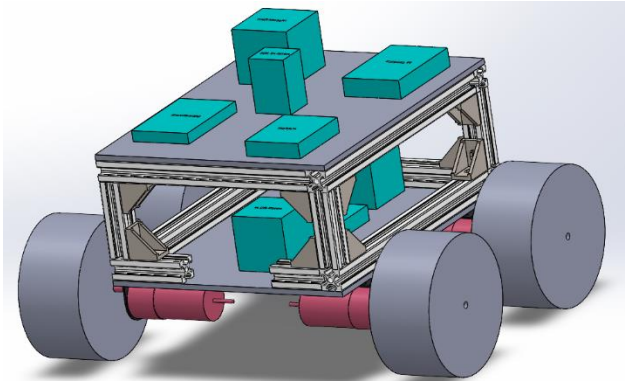


Figure 21: Side-Front View of the Robot's CAD Design

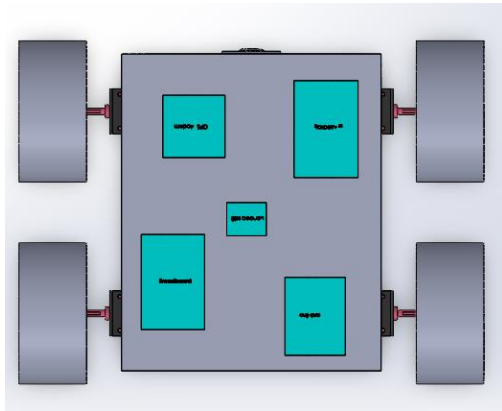


Figure 22: Top View of the Robot's CAD Design

Figures 23 and 24 demonstrate some main robot dimensions which we built our robot based on.

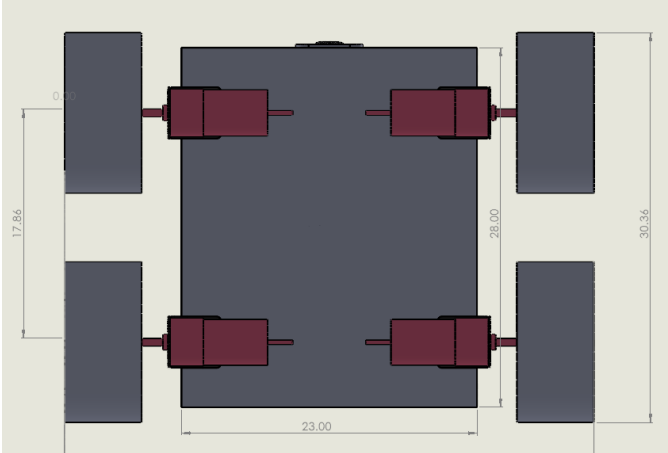


Figure 23: Robot's Lower Part Dimensions

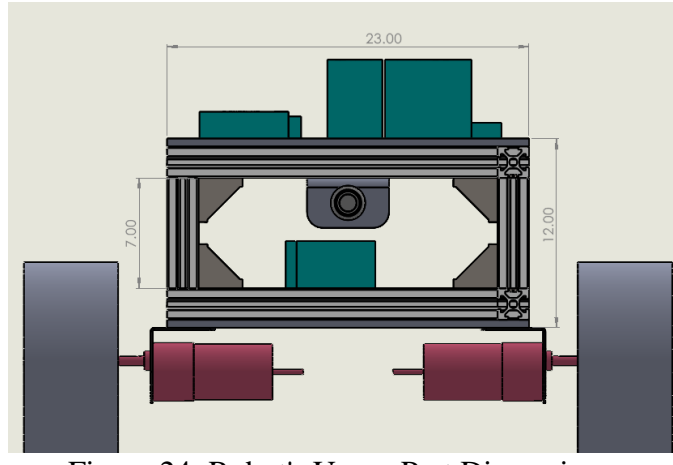


Figure 24: Robot's Upper Part Dimensions

### 3.2 Weight of the Robot

Our robot is composed of different sections that are: Storage- Collection mechanism – Batteries – Chassis – Electronics – Drivetrain. Table 4 identify the total weight of the robot and the weight of the robot with a safety factor of 1.5.

<b>Storage</b>	material	0.082
	slugs	0.0045
<b>collection mechanism</b>	material	0.05
	motors	0.018
<b>batteries</b>	normal	0.3
	power bank	0.141
<b>chassis</b>	(+Brackets)	1.891
<b>Electronics</b>	controllers	0.071
	sensors	0.034
	GPS beacon	0.1
	GPS modem	0.1
	motor drive	0.052
<b>Drivetrain</b>	Motors	0.66
	wheels	0.632
<b>Total</b>	Without SF	4.0155
	With SF	6.02325

Table 4: Total Weight of the Robot (Kg)

To choose which aluminum profile size to use we had 3 options: 2020, 3030 and 4040 (2020 denotes that this aluminum extrusion thickness is 20x20 mm and similarly with 3030 and 4040 profiles).



The most optimal option for us was 2020 but due to the unavailability of the T nuts, in the local market, to connect the profiles, we were left with 2 choices. The criteria to choose between them was the weight. Doing simple calculations of the estimated weights of the robot chassis using the 3030 and 4040 resulted that 3030 was the optimal size as the weight of the chassis using 4040 is more than double the weight of the 2020 profiles. The calculations are shown in table 5.

Thus, we choose the 3030 profiles.

	Up	Down	Height	Total	Weight/m	Weight of the Chassis
2020	940	850	280	2070	0.48	0.9936
3030	900	810	280	1990	0.9	1.791
4040	860	770	280	1910	1.46	2.7886

Table 5: Chassis' Weight of Different Aluminum Profiles

### 3.3 Motor Selection

#### 3.3.1 Motor Selection Criteria

Motor selection is based on the criteria listed below in table 6

Input Requirements	Value	Unit	Description
Robot Weight	From table 4	[Kg]	The weight of the robot is calculated by summing all the weights of the used materials along with the chassis weight. These weights are shown in table 4.
Nominal Robot Velocity	0.75	[m/s]	Our nominal velocity of the robot is chosen to be 0.75 m/s, which is enough for our robot as speed is not an important factor.
Nominal Robot Acceleration	0.25	[m/s <sup>2</sup> ]	The nominal acceleration is 0.25 m/s <sup>2</sup> .
Drive Wheel Diameter	0.125	[m]	We choose wheels with a diameter of 12.5 cm. This would give the robot the ability to drive on rough terrains and provide more height for it to provide wider space for the collection mechanism and avoid hitting some medium-sized stones in the backyard.
Friction Coefficient	0.158	[Dimensionless]	The coefficient of friction between the tire and mud is about 0.158 (Somareddy, 2017)
Motors Efficiency	75	[%]	It is estimated to be 75%
Safety Factor	2.5	[Dimensionless]	“Safety factor is the ratio between a measure of the maximal load not leading to the specified type of failure and a corresponding measure of

			the maximal load that is expected to be applied.” (Gabbay, Thagard, & Woods, 2009)
Output Requirements			
Motor Torque	3.59	[Kgcm]	The torque needed is an essential factor for the selection process. It is calculated using Equation 1
Motor Speed	114.6	[RPM]	The RPM of the motor is another important factor to choose for this process. RPM calculations are found in section

Table 6: Criteria of Motor Selection

### 3.3.2 Speed and Torque Formulas

The formulas of speed and torque are calculated using Matlab Software:

$$Speed [RPM] = 60 * NominalSpeed / (\pi * wheelRadius * 2)$$

Equation 1

$$trustForce [N] = gravity * frictionCoeff * robotWeight$$

Equation 2

$$Torque [Nm] = \frac{wheelRadius * trustForce * SafetyFactor}{4 * efficiency}$$

Equation 3

$$Torque [Kgcm] = Torque * 10.1972$$

Equation 4

### 3.4 Collection Mechanism

After studying the brainstormed ideas, it was decided to test the roller solution to decide on its feasibility. The roller mechanism consists of the following:

- **Roller:** the task of the roller is to push the slug inwards facing the ramp
- **Ramp:** the very next item in the collection mechanism after the roller, the ramp acts as the main entrance to the storage where the slugs are kept after being collected. And provide easy sliding towards the storage
- **Storage:** where the slugs are meant to be kept till the robot finished navigating and stayed in the docking station

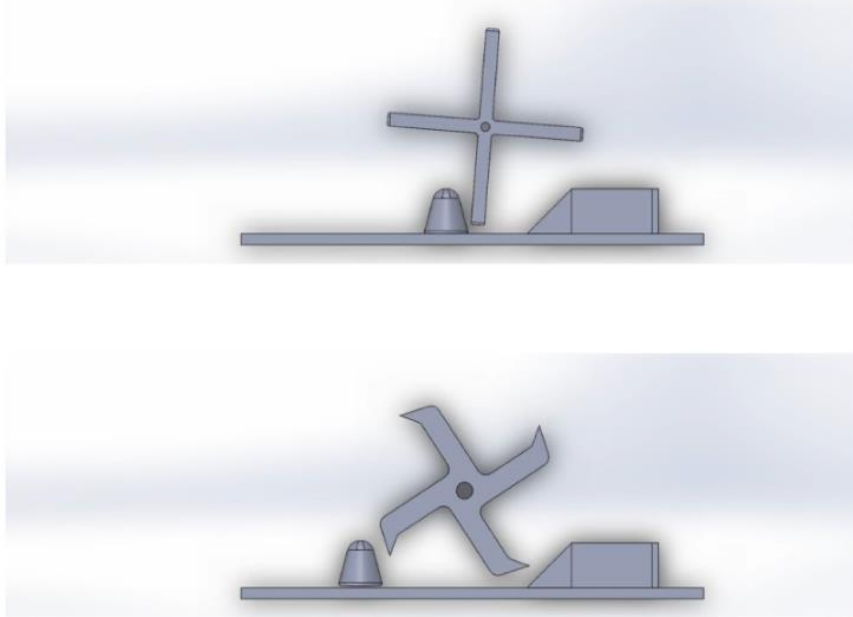


Figure 25: CAD of First Iterations of Collection Mechanism

A test was conducted, taking into consideration various variables and attributes to ensure the test feasibility.

Table 7 summarizes the test results, taking into account that Orientation 1 represents the slug being parallel to the roller blade and Orientation 2 represents slug being perpendicular to the roller blade

Distance	Motor Speed	Slug Size	Slug Orientation	Test #	Result
1.5 cm	35 RPM	M	1	1	✓
				2	✓
				3	✓
			2	4	✓
				5	✓
				6	✓
		S	1	7	X
				8	X
				9	X
			2	10	✓
				11	X
				12	X
	60 RPM	M	1	13	✓
				14	✓

			2	15	✓		
				16	✓		
				17	✓		
				18	✓		
		S	1	19	✓		
				20	✓		
				21	X		
				22	X		
			2	23	✓		
				24	✓		
				35 RPM	1	1	✓
						2	✓
3	✓						
2	4	✓					
	5	✓					
	6	✓					
S	1	7	✓				
		8	✓				
		9	✓				
	2	10	✓				
		11	✓				
		12	✓				
0.8 cm	60 RPM	1	13	✓			
			14	✓			
			15	✓			
		2	16	✓			
			17	✓			
			18	X			
	S	1	19	✓			
			20	✓			
			21	✓			
		2	22	✓			
			23	✓			
			24	✓			

Table 7: Results of Collection Mechanism Testing

Taking into consideration the test results, a comprehensive design of the collection mechanism was initiated to make a proper mechanism that can be attached to the designed

drivetrain. The design undergone many iterations and was improved based on suggested notes and feedback from stakeholders

### **3.4.1 First Iteration**

The first iteration focused on the roller and the ramp.

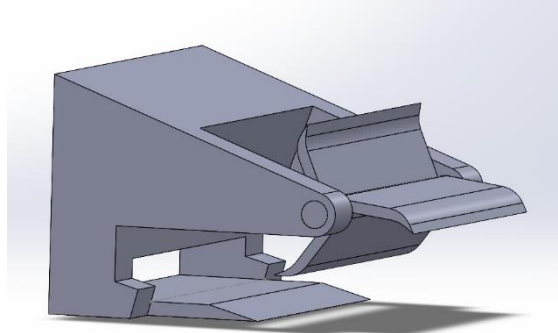


Figure 26: Design of First Iteration

### **3.4.2 Second Iteration**

In the second iteration, the mechanism was enlarged to cover the entire front of the drivetrain of 17cm and the storage was added to the mechanism with one-way gate to prevent the collected slugs from leaving the storage

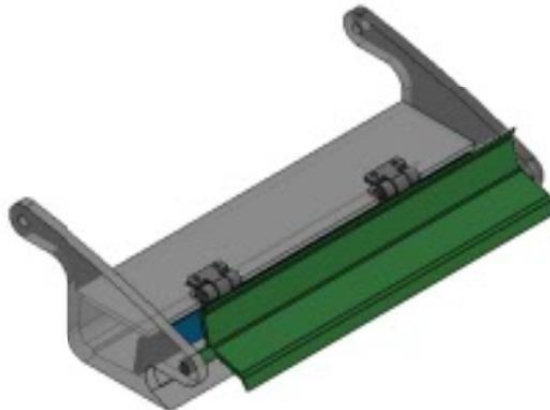


Figure 27: Design of Second Iteration

### **3.4.3 Third Iteration**

In the third and final iteration, the mechanism was reduced in size to be 10cm and support was added between the left and right handles, and the integration points were improved with even reduced space between them.

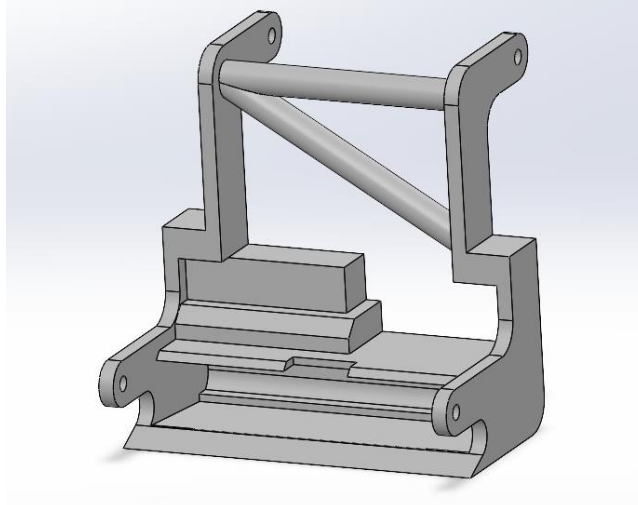


Figure 28: Design of Third Iteration

#### 3.4.4 *Integration with Drivetrain*

The mechanism is attached to the profiles in the front of the drivetrain and the space in between is left to the servo motor that's responsible for lifting the mechanism in angular fashion making the navigation of the robot doable.

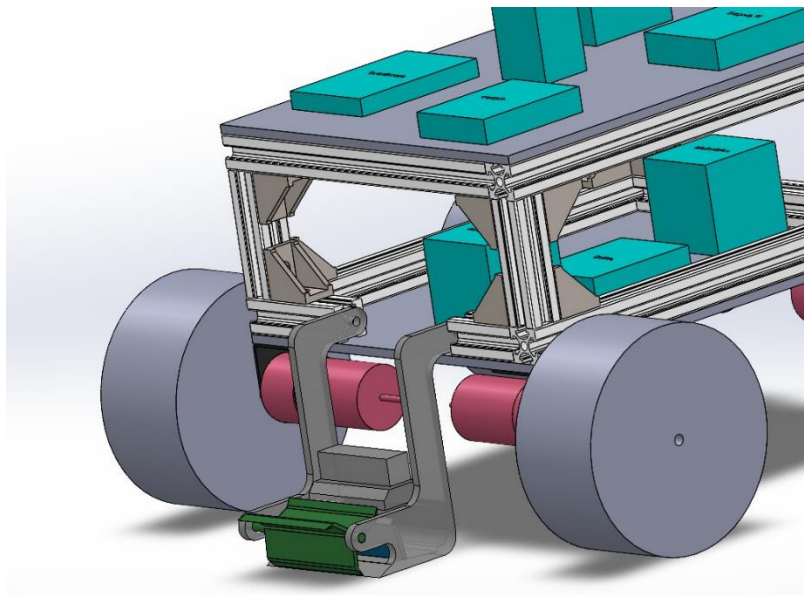


Figure 29: Collection Mechanism and Drivetrain Design

### 3.5 Battery Sizing

The two main power sources in our robot will be a lithium battery and a power bank in the following table we represented the current drawn by each component to determine the working time of our system. We calculated our battery sizing based on two modes:

#### 3.5.1 Motor Selection Calculations in the Worst-Case Scenario

Battery Sizing (50% of the max current)		
Component	Current in A	Explanation
4 DC Motors	4	50% of the stall current (2*4) *50%
Servo Motor(lifting)	0.4	50% of current at max load (0.8 * 50 %)
Dc motor (Roller)	0.097	Continuous Current(A): 97mA
4 Motor Driver	0.06	
MAX	3.41775	Amp SUM * 45 min (one cycle operational time) A/H

Table 8: Battery Sizing

Power Bank Sizing		
Component	Current in A	
Raspberry Pi	2	
Arduino Mega	0.07319	
MAX	1.5548925	Amp SUM * 45 min (one cycle operational time) A/H

Table 9: Power-Bank Sizing

#### 3.5.2 Based on Operational Time

In this method we assume one full cycle and we calculate the consumption of each component by multiplying it by its operational time percentage.

Battery Sizing (50% of the max current)		
Component	Current in A	Explanation
4 DC Motors	3.6	the driving motors operates for 90% of the time
Servo Motor(lifting)	0.04	the lifting mechanism operates for 10% of the time
Dc motor (Roller)	0.03395	the Roller mechanism operates for 35% of the time
4 Motor Driver	0.06	
MAX	2.8004625	Amp SUM * 45 min (one cycle operational time) A/H

Table 10: Battery Sizing - Operational Time

Power bank sizing in this case is similar to that in table 9 in section 3.5.1 because the Arduino mega and the raspberry will operate for 100% of the cycle.

### 3.5.3 *Battery C Rating*

A battery's charge and discharge rates are controlled by battery C Rates. The battery C Rating is the measurement of current in which a battery is charged and discharged at. The capacity of a battery is generally rated and labelled at the 1C Rate (1 C current), this means a fully charged battery with a capacity of 10Ah should be able to provide 10 Amps for one hour. That same 10Ah battery being discharged at a C Rating of 0.5C will provide 5 Amps over two hours, and if discharged at a 2C Rate it will provide 20 Amps for 30 minutes. The C Rating of a battery is important to know as with the majority of batteries the available stored energy depends on the speed of the charge and discharge currents. (PowerSonic, 2022)

Regarding our battery based on the datasheet has a 1C maximum discharge current.

4400mAh battery

$$4400\text{mAh} / 1000 = 4.4 \text{ A}$$

Equation 5

$$1\text{C} * 4.4\text{A} = 4.4 \text{ A available}$$

Equation 6

$$1/1\text{C} = 1 \text{ hour}$$

Equation 7



# CHAPTER 4

## HARDWARE

### 4.1 Drivetrain

Our robot chassis dimensions are 280 x 230 mm

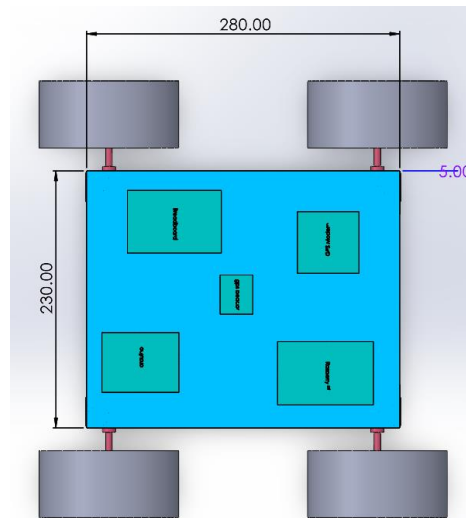


Figure 30: Robot's Width and Length in mm

#### 4.1.1 Choosing Aluminum Profile Size

Profile Size	Upper Part needed Length	Lower Part needed Length	Middle needed Length	Total Length needed	Profile Weight/ Meter [Kg]	Total Weight of the Chassis
2020	940	850	280	2070	0.48	0.9936
3030	900	810	280	1990	0.9	1.791
4040	860	770	280	1910	1.46	2.7886

Table 11: Length and Width of needed Aluminum Profile

At first, we opted for 2020 aluminum profile as it has the smallest thickness and the least weight. But due to the unavailability of suitable T-nuts for these profiles in the local market, we moved to the 3030 profiles which are much lighter than the 4040.

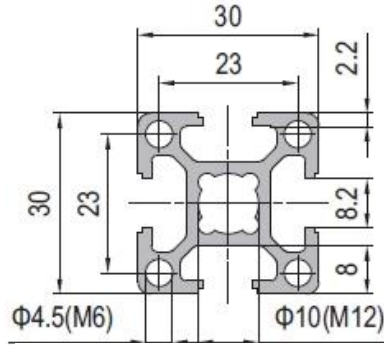


Figure 31: 3030 Aluminum Profile Pattern Used

#### 4.1.2 Cutting Aluminum Profiles

We started by cutting the needed aluminum profiles sizes. Table 11 shows the sizes to be cut along with their quantity.

Profile Length (mm)	25	20	3.5	6.5	7
Quantity	4	3	1	1	4

Table 12: Aluminum Profiles Sizes and Quantities



Figure 32: Aluminum Profiles cutting Machine at Raymond Feghali Co

#### 4.1.3 Chassis Attaching Process

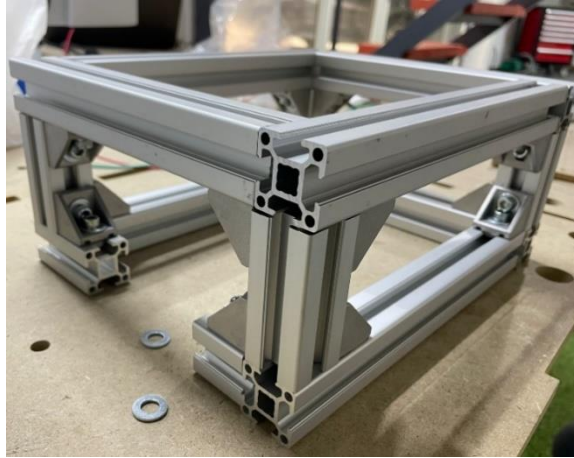


Figure 33: Robot Chassis

The profiles are connected via 3030 L type brackets, 1 between each aluminum pair. Moreover, these L brackets are fixed to the profiles by M5-T nuts, washers and M5 screws.



Figure 34: T-NUT,  
WASHER, AND SCREW  
ASSEMBLY

#### ***4.1.4 Cutting Wooden Plates***

We cut down two 28x23cm wooden plates with a thickness of 7 mm. One is fixed at the bottom of the chassis to screw the motors on, and the other is at the top where some electronics along with the GPS are placed. Each plate is fixed with the chassis using 8 M5 T-nuts, washers with their screws.



Figure 35: Wooden Plate Used

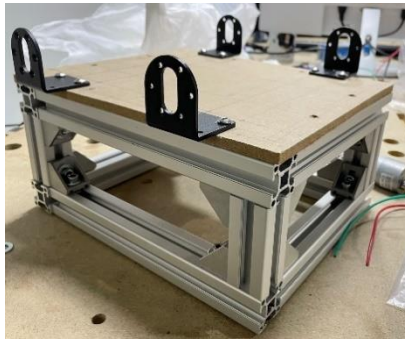


Figure 36: Robot Chassis with the Wooden Plate and Motor Brackets

#### ***4.1.5 Fixing Motors***

Motors are fixed using aluminum bracket as shown in figure 36. These brackets were fixed the same way as the plates. Two combinations of M5 T-nuts, washers and their screws were used for every bracket.

#### ***4.1.6 Adding Wheels***

The chosen  $\phi 12.5$  cm wheels are shown in figure 37.  $\Phi 6$ mm motor coupling, as shown in figure 38, were utilized to attach these wheels with the motors as the wheel hole diameter is 6mm.



Figure 37: Chosen Wheel



Figure 38: Motor Coupling



Figure 39: Robot Chassis with Motors and Wheels

#### ***4.1.7 Components on the Lower Plate***

In the lower plate, we have added the two motor drives and the battery. A Tri-color LED latching push button with 2 SPDT switches where 1 is NO and the other is NC was used. It was screwed on an upper profile.

#### **4.1.8 Components on the Upper Plate**

We added next the upper wooden plate which was fixed like the lower plate. An Arduino, raspberry pi, raspberry pi camera module v1, and the GPS hedgehog beacon were all assembled as shown in figure 40.



Figure 40: Assembled Robot with all Hardware Components

## **4.2 Motors and Motor Drives Selection for the Drivetrain**

### **4.2.1 Motors**

After the motor calculations Chapter 3, the next step is to choose the most suitable motor sizing parameters, based on the local market. One of the problems in the Lebanese market is that the available motor's specifications are very limited, and it is rare to find one with our exact needs. Thus, the motors we chose are oversized.

After searching all available options, we ended up with 2 choices:

- 100 rpm, torque 12kgcm, stall current 2A
- 120 rpm, torque 18kgcm, stall current 7A

We traded off some needed speed to get a motor with much less stall or max current to avoid buying a motor drive with a bigger max current rating of 7A. Thus, due to power consumption reasons, we opted for the 2A motors. After searching online for the datasheet, as the supplier doesn't have one. The motor serial number is 37GB520 and the model is 17320-70. It has the following specs from the datasheet:

Serial No.	Model	Rated Voltage V	No Load		At Maximum Efficiency				Stall	
			Speed rpm	Current mA	Speed rpm	Torque Kgcm	Current A	Output W	Torque Kgcm	Current A
			13490-50	24	180	50	145	2	0.22	3
11750-490	24	12	45	11	10.0	0.1	1.2	40	0.7	
11750-270	12	11	25	9	6	0.1	0.5	30	0.4	
13490-70	12	60	50	48	2	0.15	1	10	0.6	
37GB520	17320-70	12	96	100	80	2	0.32	1.6	12	2
	20260-30	12	280	100	230	1.5	0.6	3.5	8	2.6
	25150-50	6	152	170	125	1.5	0.85	1.9	8	3.5
	17410-90	12	60	50	48	4	0.32	2	18	1.5
	11750-50	24	125	45	100	2	0.15	2.1	10	0.64

Figure 41: 37GB520 Motor Specifications (NOSTOP ELECTRIC CO.)

We will be operating our motor at a torque of 4 Kgcm. The below graph demonstrates that at this point we will get a speed of 66.67 RPM. This is less than what is needed but it won't have a big effect on our robot as speed is essential in our application.

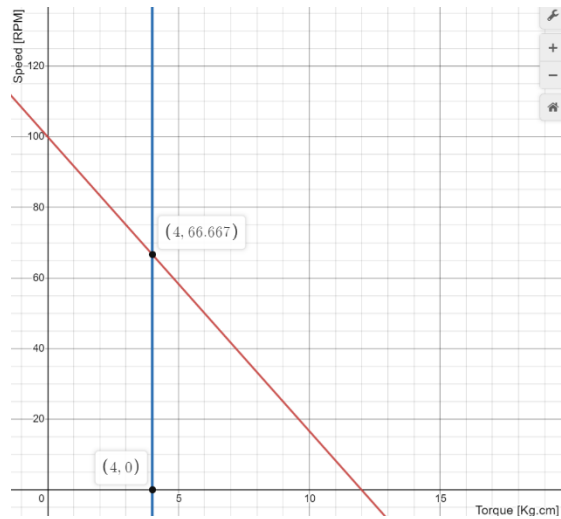


Figure 42: Motor Specifications at our Operating Point using Desmos Graphing Calculator



Figure 43: 37GB520 Chosen Motor

#### 4.2.2 Motor Drives

The motor driver should have a maximum current rating equals or more than the motor's stall current.

Our selected motors have a stall current of 2A. The most suitable motor in the market was the double bridge L298N motor driver that can hold a maximum of 2 A drive current in each bridge. For our robot, we need 4 motors. Thus, we require 2 motor drivers, 1 full bridge or driver for each pair.

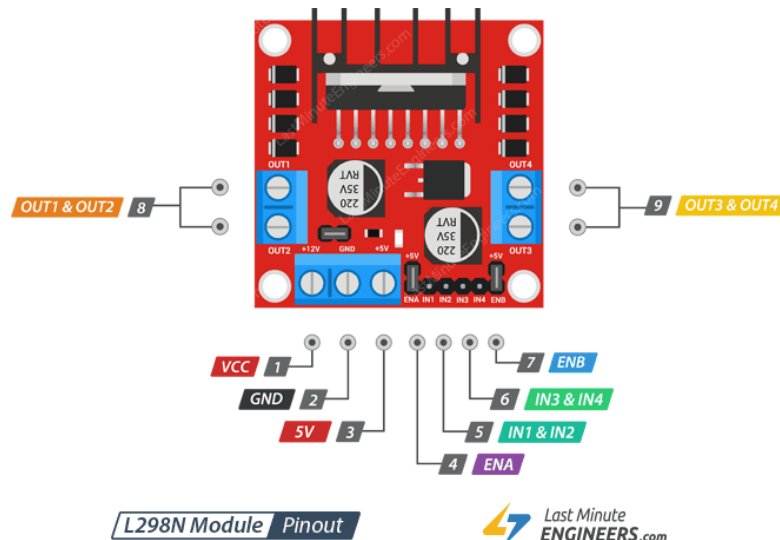


Figure 44: L298N Motor Driver (Last Minute Engineers, 2021)

### 4.3 Sensors

#### 4.3.1 Raspberry Pi Camera Module V2



The raspberry pi camera module v2 is a 8-Megapixel camera compared to the 5-Megapixel of the raspberry pi camera module v1. It is easy to use and can take clear and colorful pictures and videos to use it for different purposes. It can work with all versions of raspberry pi.

There are several available libraries that makes it easy to integrate into different projects that need a visual input. (Raspberry Pi, 2016)

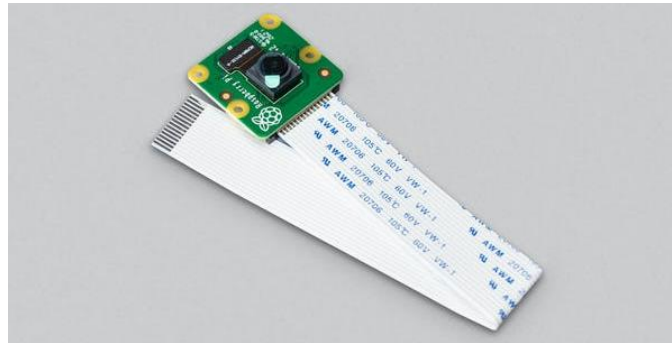


Figure 45: Raspberry Pi Camera Module V2

#### 4.3.2 *Marvelmind Indoor GPS Kit*

For the robot to navigate autonomously, one of its major elements is to ensure the position and orientation of the robot for it to localize itself in the map.

To accomplish the localization step, we chose indoor navigation system based on stationary ultrasonic beacons linked by radio interface in the license-free ISM band. The location of a mobile beacon mounted on a robot (vehicle, copter, or human) is determined using trilateration and the propagation delay of an ultrasonic signal to a series of stationary ultrasonic beacons. (Swaidan, Baradhi , Mechlawi , & Yahya, 2021) Where, Trilateration is a technique for calculating the position of a mobile beacon by measuring distances between three or more known locations. Marvelmind indoor GPS starter set shown in the figure 46 consists of:

- 1 Mobile beacon fixed on the moving object (human, robot, vehicle etc.)
- Stationary beacons where distance between neighbor beacons is not greater than 30m and the mobile beacon is able to reach at least 3 stationary beacons.

- 1 modem that communicates with all stationary and mobile beacons in which it receives position data, it must be able to reach all beacons with a range of 100m. (Marvelmind, 2019)



Figure 46: Marvelmind Indoor GPS Kit

The kit used in our robot consists of a Mini-TX beacons (1 modem, 1 hedgehog beacon, 2 stationary beacons) with the following specifications in table 12:

Specialty and Main-Use	Small TX only beacon
Mode of Operation	TX only
Range	Up to 30m with Super-Beacon
Ultrasonic Frequencies	31/45kHz
Radio Band	915/868MHz
Power/Lipo Battery	USB/250mAh
Environmental Conditions	Indoor/Outdoor t=0 to 40°C
Size and Weight	35x35x26mm and 19g

Table 13: Mini-TX GPS Beacon Specifications (Swaidan, Baradhi , Mechlawi , & Yahya, 2021)

### 4.3.3 MPU-6050

MPU6050 sensor module is complete 6-axis Motion Tracking Device. It combines 3-axis Gyroscope, 3-axis Accelerometer and Digital Motion Processor all in small package. Also, it has additional feature of on-chip Temperature sensor. It has I2C bus interface to communicate with the microcontrollers.



Figure 47: MPU-6050

#### 4.4 Collection Mechanism

##### 4.4.1 *Motor for Rotating the Mechanism*

The motor responsible for the roller rotation is a normal geared 12v DC motor with maximum speed of 60rpm. A simple motor bracket is used to mount the motor on a special made platform on the collection mechanism. As shown in figure 48, a gear and a helping belt are the accessories used to transfer the rotational movement of the motor to the collection roller.



Figure 48: Motor for Rotating the Mechanism

##### 4.4.2 *Motor for Lifting the Mechanism Radially*

The motor responsible for radially lifting the entire collection mechanism is Servo motor MG995 with a mounting plate accessory to help attach it to the collection mechanism as in the figure 49.



Figure 49: Servo Motor for Lifting the Mechanism

#### 4.5 Bill of Materials

Material	Quantity	Description	Cost (\$) / piece	Total cost(\$)	Source
-	-	<b>DRIVE-TRAIN</b>	-	-	-
Aluminum Profile 3030	1.99 m [24x4 + 20x3 + 3.5x1 + 6.5x1 + 7x4] [length(mm) x Quantity]	3030 Cross-section dimensions of 30 mm by 30 mm	12\$/meter	23.88	Raymond Feghali Co.
Aluminum Corner 3030 Brackets	16	Aluminum brackets (Holds 2 profiles together)	1.6	25.6	Raymond Feghali Co.
Wooden Plates	2	28x23 cm	-	-	SPEXAL
Motors	4	12V 100RPM 2A stall current 12kgcm Torque	12	48	BEC

Motor Brackets	4	-	1.5	6	BEC
Wheels	4	12.5 cm	5.5	22	Katranji
Couplers	4	Φ6MM COUPLING	1	4	Katranji
Screws, washers, and t-nuts	48 Combination [8 (For upper plate) + 8 (For lower plate) +16*2 ( For Brackets)]	For 3030 profiles M5 T-Nuts and Screws Number	48*(0.45+0.3)	36	Raymond Feghali Co.
-	-	<b>ELECTRONICS</b>	-	-	-
Arduino	1	Arduino Uno	8	8	University Lab
Raspberry pi 3 model B+	1	3 Model B	54	54	University Lab
RGB Camera	1	Raspicam v2 8 MP	11	11	University Lab
Breadboard	1	-	1.5	1.5	SPEXAL
Motor Drive	2	L298N	2.5	5	BEC
Battery	1	12V 4400mAh 18650 Lithium-ion Battery + 12.6V 1A Charger	24	24	Electroslab
Wires	-	Male-Male, Female-Female, Male-Female types of wires	-	12	University Lab
-	-	<b>COLLECTION</b>	-	-	-
Servo Motor	1	MG995	8	8	University Lab
DC Motor	1	12 V	10.4	10.4	University Lab
Timing Belt	1	2GT (6mm) 110 mm	1.25	1.25	University Lab
Pully	2	Timing Belt Pulley 5mm GT2	2.25	4.5	Dynamic

Table 14: Bill of Materials

## 4.6 Wiring Diagram

Our robot is composed by connecting different electronics component, the wiring process is divided into two main areas: Signal(control)- Power.

- **Power:**

The electronics component in our robot are powered by three power sources. First the power bank is powering the raspberry Pi that is powering the camera to be used, and the GPS are powered by batteries, then the main battery is connected to 4 motor drives that are powering our 6 different motors, on the other side one of these motor drives is powering the Arduino that is powering the sensors we have

- **Signal (Control):**

After powering our component, we moved to the controlling them. First the camera and the GPS is connected to the raspberry pi and the raspberry pi, the sensors and the motor drives are connected to the Arduino, and each for the drive train each two motors are controlled by a motor drive and for the collection mechanism the DC motor connected to the roller and the lifting motor (Servo) are controlled by separate motor drives.

# ELECTRONICS BLOCK DIAGRAM

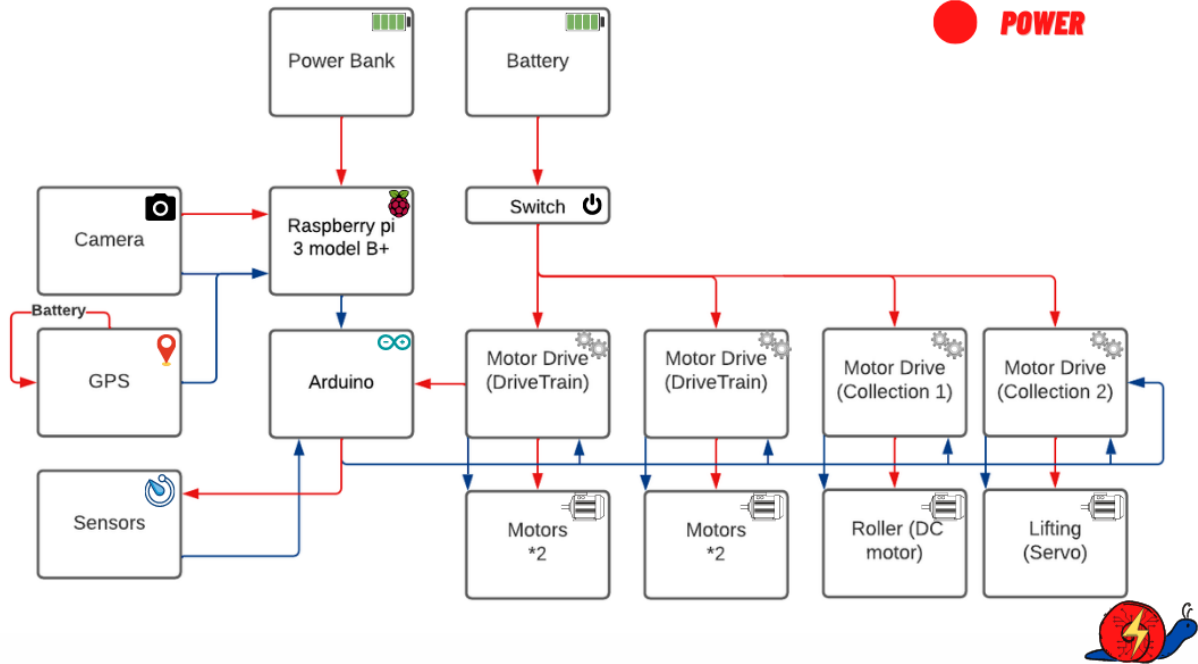


Figure 50: Wiring Diagram

# CHAPTER 5

## IMPLEMENTATION AND RESULTS

The project includes implementation of four sections to obtain a functioning robot that can navigate in a backyard, detect slugs, and collect them safely.

### 5.1 Navigation Implementation

Our target is to implement autonomous navigation on the designed robot. To do so, we have four stages we must accomplish which are Mapping, Localization, Path Planning and Obstacle avoidance.

#### 5.1.1 Mapping

Having a map is a major first step in the navigation process, where we drew a grid map of a backyard using a software (GIMP). After drawing the map, its parameters in the Yaml file should be adjusted as shown in figure 51.

```
image: my_map1.pgm
resolution: 0.01 # 300 pixels == 3 meter (300*.01=3)
origin: [0, 0, 0] #[-15.400000, -13.800000, 0.000000]
negate: 0
occupied_thresh: 0.65
free_thresh: 0.196
```

Figure 51: Map Yaml file

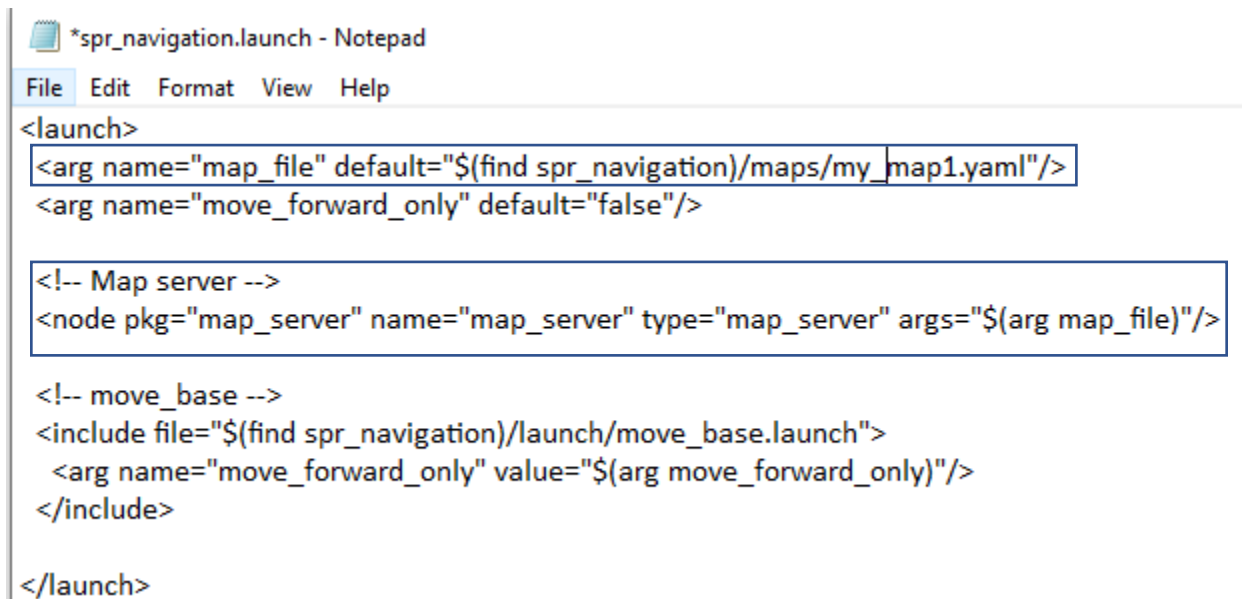
Required fields:

- image: Path to the image file containing the occupancy data; can be absolute, or relative to the location of the YAML file
- resolution: Resolution of the map, meters / pixel
- origin: The 2-D pose of the lower-left pixel in the map, as (x, y, yaw), with yaw as counterclockwise rotation (yaw=0 means no rotation). Many parts of the system currently ignore yaw.
- negate: Whether the white/black free/occupied semantics should be reversed (interpretation of thresholds is unaffected)



- `occupied_thresh`: Pixels with occupancy probability greater than this threshold are considered completely occupied.
- `free_thresh`: Pixels with occupancy probability less than this threshold are considered completely free.

Then we included the map server in the general launch file.



```

*spr_navigation.launch - Notepad
File Edit Format View Help
<launch>
<arg name="map_file" default="$(find spr_navigation)/maps/my_map1.yaml"/>
<arg name="move_forward_only" default="false"/>

<!-- Map server -->
<node pkg="map_server" name="map_server" type="map_server" args="$(arg map_file)"/>

<!-- move_base -->
<include file="$(find spr_navigation)/launch/move_base.launch">
  <arg name="move_forward_only" value="$(arg move_forward_only)"/>
</include>

</launch>

```

Figure 52: Map Launch File

### 5.1.2 Localization (GPS+IMU)

For the robot to localize itself, we used the Marvelmind Indoor GPS Toolkit to get the position and MPU-6050 to extract the rotation. All the technical steps are detailed in Appendix A

### 5.1.3 Path Planning

For the path planning, the robot will start when the user sends a goal through Rviz and the planner will choose the shortest path to navigate through. Refer to Appendix A for more technical information.

## 5.2 Detection Implementation

The process of Model building and deploying consists of 5 main steps. First, we started by creating our dataset, followed by labelling them. In the next step, we trained our model using our custom dataset after which we run inference with the trained weights using raspberry pi

webcam. Finally, we sent velocity commands to the robot to reach the detected slug. Detailed process can be found in Appendix B

### 5.2.1 Dataset Collection

In any machine learning related problem, we first start by searching for available datasets, as the process of data collection is usually a hardship. Unfortunately, after searching for online datasets of slugs, we found only one dataset (<https://www.kaggle.com/tegyntwmffat/slug-detect>) with 989 photos. The problem with it was that the images of the slugs were close-up photos which is not the case for our robot as the camera will be inclined and will have a wider range of view. As a result, we went with the option of building our own dataset from scratch, which was our only option.

To build our own dataset, we first collected a total of 6 slugs around Rafik Hariri University campus which wasn't an easy task as we searched for them during rain. Afterwards, we captured images of slugs in different environments and various conditions such as lighting, surface type etc., around the campus. The total number of images collected was 150 photos.



Figure 53: Four Sample Images from the Dataset

### 5.2.2 Data LabelImg

To train, we used YOLOv5 model, where annotated images are used in the training process. The annotation is the process of selecting the needed object for the model to detect it. This is done by drawing a boundary box or any other shape, depending on the used software options, around our target and making sure that the box is as tight as possible around the object needed to be detected, for better training.

The labeling procedure was performed through “LabelImg” program. After annotation, we resulted with a folder of 150 text files, containing the coordinates of the bounding box for each image and our images.

### 5.2.3 Train Our Custom YOLOv5 Model

We initially select a pretrained model to start training from on our dataset. YOLOv5 provide 5 pretrained model with a tradeoff between precision and size, which are shown below in figure 54.

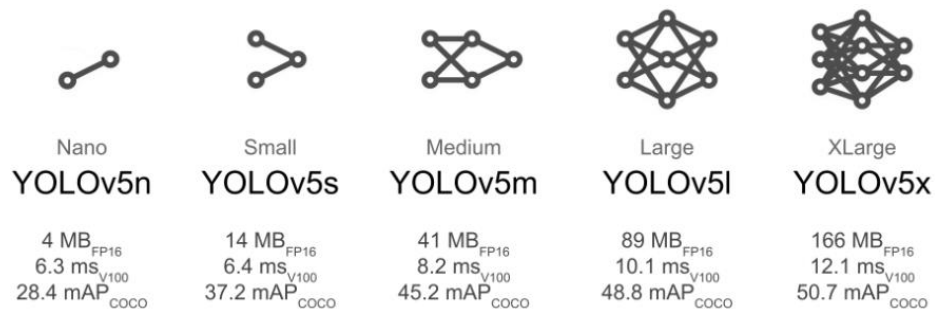


Figure 54: YOLOv5 Model Types

For our application, we used YOLOv5s which is compatible in size and good in precision.

We used Google Colab, which allows for python writing and execution in a browser, to train the model. The Google Colab code loads the YOLOv5s pretrained model and retrains it using our custom model. This is done using the “train.py” file where we pass several arguments: image, batch size, epochs, dataset location, the chosen pretrained model weights, and cache images.

### 5.2.4 Run Inference with Trained Weights

After training the model on our custom dataset to detect slugs, we run inference or test the model. This can be done using the “detect.py” file which

deploys the new model on input and returns an output image with a bounding box around the objects our model predicts as slug.

### **5.2.5 *Sending Commands to the Robot***

After detecting a slug, the robot centers itself first with the centroid of the located slug by turning right or left. We drew a virtual region of interest at the center of the frame, where the robot stays on turning until this slug is inside this box, indicating its alignment with the robot. Finally, the robot stays on moving forward and stops only when the slug becomes inside the boundary of another virtual ROI implying the robot is ready to start the collection process. The process is demonstrated in detail in Appendix B.

All the above steps can be applied under certain constraints in the working environment where the model only work in daylight

Also, the model has some limitations such as:

- Place of detected slugs isn't accurate as we don't have depth factor
- The model detects only the types of slugs that are similar to the ones it was trained on, and it is not generalized to all slug types and colors.

## **5.3 Collection Mechanism Implementation**

The design of the collection mechanism that's discussed in chapters 2 and 3 was implemented using the 3D printing technology. It was printed using Fused Deposition Modeling (FDM). The selected material for printing was filament of PLA (Polylactic Acid). the overall weight of the print was approximately 180g and took around 15 hours to finish.

The mechanism function well taking into consideration an environmental constraint where the terrain that the slug will be located on should be a flat terrain; the mechanism is designed with a very flat manner

## **5.4 System Integration Implementation**

After implementing navigation, detection and collection, each step alone, we integrated all of them together to obtain a fully functioning robot. The output of navigation is a velocity command, also the output of the detection is a velocity command. Using Twist multiplexer

package from ROS, we integrated both command velocities. The main node of this package is twist\_mux, which provides a multiplexer for geometry\_msgs::Twist messages. It takes N input twist topics and outputs the messages from a single one. For selecting the topic they are prioritized based on their priority, the messages timeout and M input lock topics that can inhibit one input twist topic. (NickLamprianidis, 2018) This is illustrated in figure 55 below

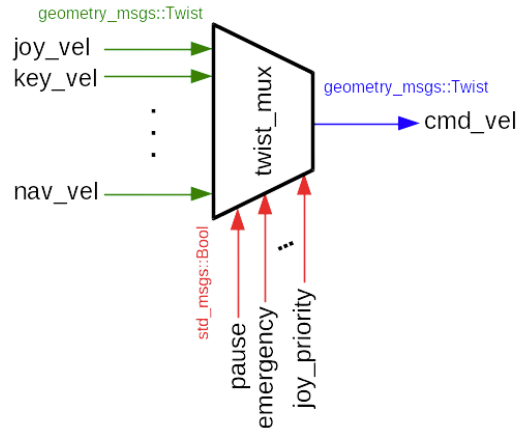


Figure 55: Twist\_MUX  
(NickLamprianidis, 2018)

In other words, we have a velocity command from planner (cmd\_vel\_planner) and velocity command from detection (cmd\_vel\_vision) and both publishes to the original command velocity (cmd\_vel/). The priority in our case, that can be specified in the MUX package parameters, is given to the command velocity from vision (cmd\_vel\_vision). In this way, the robot will start navigating according to the command velocity from the planner but whenever the camera detects a slug, the robot will navigate according to the command velocity from detection(cmd\_vel\_vision).

As shown in figure 56, we edited the Yaml files of the twist\_mux topics where we assigned priority of 1 for the cmd\_vel\_planner and priority of 2 for cmd\_vel\_vision (the priority ranges from 0 to 255, and the higher the more priority is given for the topic).

```

twist_mux_topics.yaml - Notepad
File Edit Format View Help
# Input topics handled/muxed.
# For each topic:
# - name : name identifier to select the topic
# - topic : input topic of geometry_msgs::Twist type
# - timeout : timeout in seconds to start discarding old messages, and use 0.0 speed instead
# - priority: priority in the range [0, 255]; the higher the more priority over other topics

topics:
-
  name : planner
  topic : cmd_vel_planner
  timeout : 0.5
  priority: 1
-
  name : vision
  topic : cmd_vel_vision
  timeout : 0.5
  priority: 2

```

Figure 56: Priority of Topics in MUX Yaml Files

Finally, the following figure, rqt graph, shows all the nodes running at the same time after running the entire system.

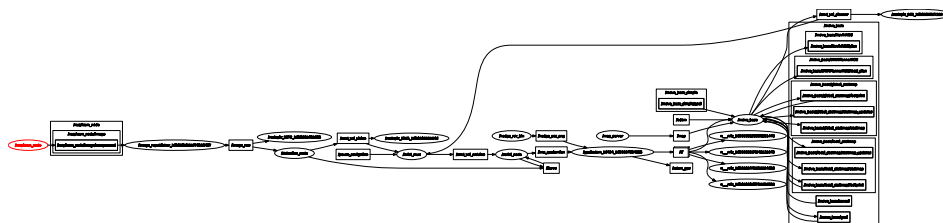


Figure 57: Rqt Graph of all the running Nodes

Also, the working demo can be find in the link below:

[https://studentsrhuedu-my.sharepoint.com/:v:/g/personal/marzoukjb\\_students\\_rhu\\_edu\\_lb/EUSmbK-bjPpLnRI03pP6xJUBKpH3xikr66JbuL2q8QbM2w?e=jy9er1](https://studentsrhuedu-my.sharepoint.com/:v:/g/personal/marzoukjb_students_rhu_edu_lb/EUSmbK-bjPpLnRI03pP6xJUBKpH3xikr66JbuL2q8QbM2w?e=jy9er1)

## CHAPTER 6

### CONCLUSION AND FUTURE WORK

Integrating all the system hardware and software aspects, resulting in a functioning prototype that proves the concept. The robot can navigate autonomously, detect slugs and collect them safely, taking into account some constraints. The following are some improvements that can be applied, in the future, to ensure a better performance in the second iteration:

#### 6.1 Future Work in Navigation

- Using GPS outdoor kit instead of indoor kit.
- Integrating obstacle avoidance in the navigation system using an Infrared sensor.
- Formulating an algorithm, such as waypoints, that enables the robot to navigate without taking commands from the user through Rviz.

#### 6.2 Future Work in Slug Detection

- Using an RGBD camera. For the proof-of-concept phase using an RGB camera and approaching the slug based on slug's location in ROIs was enough. But for further detection improvement and location extraction of the slug more precisely, a depth factor should be introduced. This would also make using other more accurate collection mechanisms such as a gripper more plausible.
- Improving the quality of the Yolov5 model by getting more images and using a higher-level model than the used Yolov5s.
- Using a processor that has GPU such as Jetson Nano. The onboard processing can be done rather than sending the webcam feed to ROS, decompressing the frames and converting them to OpenCV form to be able to use the detection model on it. This can make the overall process much faster.

#### 6.3 Future Work in Slug Collection

- Modifying the roller part of the mechanism to be more Slug friendly. Many flexible materials are considered, to replace the sharp edgy endings of the roller. It can be

designed to be flexible from the get-go or the current roller can be adjusted gradually. A considered material is Polyamide.

- Integrating the entire mechanism more in the drivetrain. In the future work a space can be created to make the collection mechanism inside the drivetrain with a flexible fast mechanism to bring it out and lower to the slug level so the collection sequence can be initiated.
- Making the storage more user friendly where it'll help the end user clear the storage from the collected slugs and easier to clean, whether from the slugs residue or unwanted collected debris.
- Improve the collection mechanism to be more effective in rough terrains whether straight or inclined must be put into work.



# APPENDIX A

## NAVIGATION TUTORIALS

### i. Marvelmind Software Installation

Marvelmind Indoor navigation System of Non-Inverse Architecture is used in localization and path planning in the autonomous navigation. To use it, a specific Dashboard should be installed from the official website of Marvelmind.

- We start by downloading and installing STM32 driver once.

**Download STM32 driver** [Download and install STM32 driver on your computer once](#)

The driver is required on your computer in order for the processors used in beacons and modems to be visible in Windows. Linux already has it. Thus, you don't have to install it

Figure 58: Downloading and Installing the Dashboard Driver

- Download latest SW.

**Download SW/API pack** [Download latest SW pack](#)

The main SW pack required for any Marvelmind hardware. The pack also includes the API

- Latest stable [Marvelmind SW pack](#) (85Mb, 01.Nov.2021). Download the pack, if you have any Marvelmind HW
  - v7.000 – Base SW release (no license purchase required)
  - v7.100 – Base SW release + Optional SW pack (license purchase per beacon required)
  - [Release notes for the SW pack v7.000/v7.100](#) (01.Nov.2021)
- API can be found inside the Marvelmind SW pack as a subfolder. [Download API](#)

Figure 59: Downloading SW Pack

- Copy Dashboard and Dashboard API to directory that you will use for the program. Note that Linux version is provided for two hardware platforms: **x86** (most of laptops based on intel or AMD CPU) and arm (for example, single-board computers like Raspberry PI). In our case, x86 is needed.

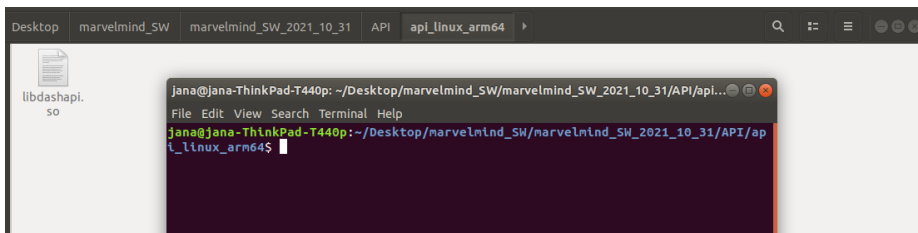


Figure 60: Dashboard API in Directory

- You will need to give rights for the user to access serial port by adding him to dialout group:

Execute in terminal: **sudo adduser \$USER dialout**

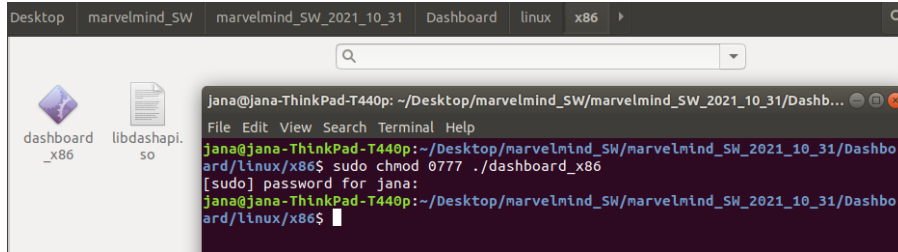


Figure 61: Access Serial Port

- Add to the directory /etc/udev/rules.d file “99-tty.rules” with following content:

#Marvelmind serial port rules

KERNEL=="ttyACM0",GROUP=="dialout",MODE=="666"

## ii. Beacons Setup along with the Dashboard

- Connect modem to pc and choose the modem\_hw49 firmware to download. (From software\_nia folder)

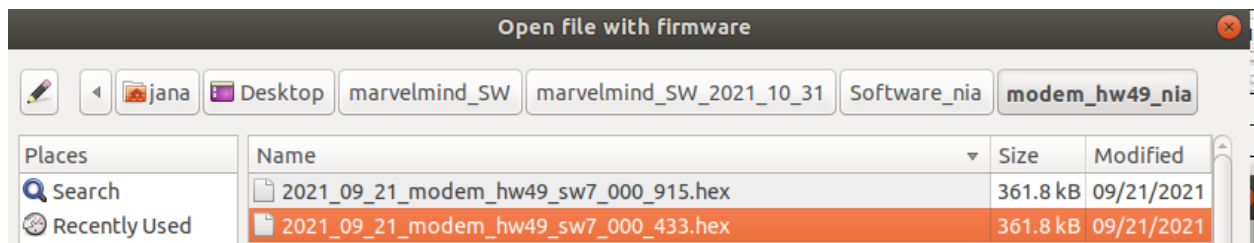


Figure 62: Modem Firmware

- Connect beacons and download their firmware on them. When the firmware is downloaded on a beacon, we directly connect the next one and it automatically downloads the firmware.
- After connecting the beacons, choose one to act as mobile beacon (Hedgehog), and enable the Hedgehog in the Dashboard. after connecting all the beacons, press default, then Write all. Afterwards, number each beacon any number from 1 till 254 in the Dashboard (Device address (1..254)).

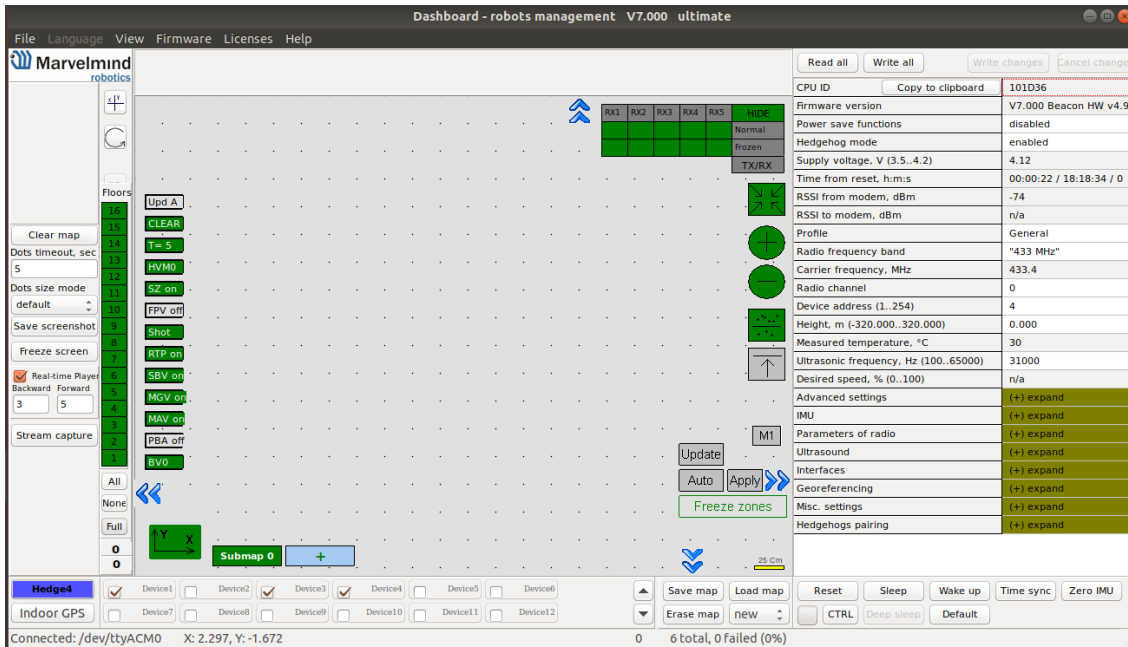


Figure 63: Marvelmind Dashboard after Connecting all the Beacons

- We connect the modem and then we wake up the beacons by pressing Device 1, at the bottom.
- Click on submap then select the beacon that is desired to be the origin. Choose the origin by setting the number of the beacon in "Starting beacon trilateration (0..255), and the origin will be set.

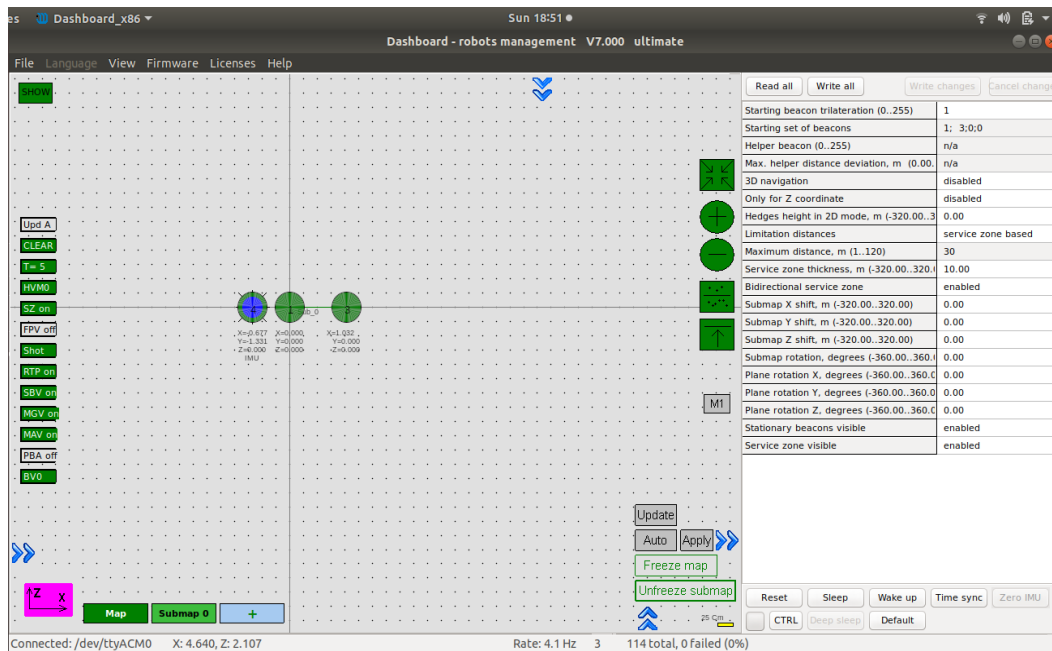


Figure 64: Marvelmind Origin Selection in the Dashboard

- Freeze the map and submap to fix the beacons. Afterwards, only the hedgehog will be able to move on the map.
- Remove the modem from the pc (but it should be connected to a power source).

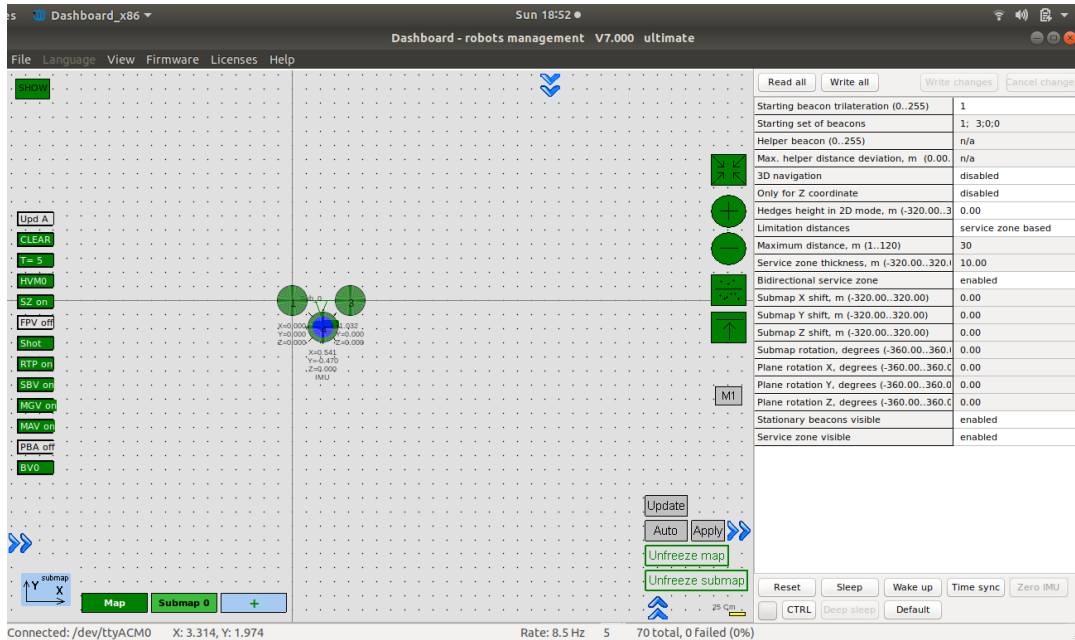


Figure 65: Marvelmind Dashboard of Frozen Map

- IMU of the hedgehog should be calibrated (only the hedgehog's IMU should be calibrated because the other beacons are stationary).

Therefore, connect the hedgehog to the PC and in the View tab of the Dashboard, press Accelerometer to start the IMU calibration, press Reset then press PAUSE ON and tilt the hedgehog around the first axis. This process should be repeated around the 3 axes.

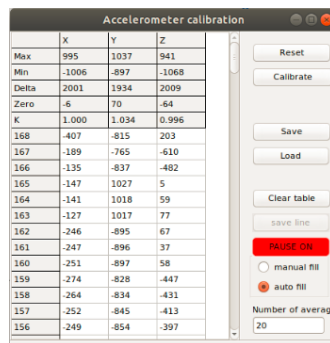


Figure 66: Accelerometer Calibration

- In the dashboard expand IMU and select High performance for the IMU mode, then press on Processed IMU Data.

External device control	No control
PA15 pin function	SPI slave CS
Raw inertial sensors data	disabled
Processed IMU data	enabled
Raw distances data	disabled
Quality and extended location data	disabled
Telemetry stream	disabled

Figure 67: High Performance of the IMU

### iii. ROS Marvelmind Package

Marvelmind supplies ROS package marvelmind\_nav, which is able to communicate with mobile beacon or modem and **provide received location** and other data.

The following link includes the ROS package that is available in source repository.

[https://bitbucket.org/marvelmind\\_robotics/ros\\_marvelmind\\_package](https://bitbucket.org/marvelmind_robotics/ros_marvelmind_package)

- Create a workspace, copy the package to src folder, then catkin\_make, and source the workspace.

```

roscore http://jana-ThinkPad-T440p:11311/
jana@jana-ThinkPad-T440p:~$ cd gps_ws
jana@jana-ThinkPad-T440p:~/gps_ws$ source devel/setup.bash
bash: devel/setup.bash: No such file or directory
jana@jana-ThinkPad-T440p:~/gps_ws$ catkin_make
Base path: /home/jana/gps_ws
Source space: /home/jana/gps_ws/src
Build space: /home/jana/gps_ws/build
Devel space: /home/jana/gps_ws/devel
Install space: /home/jana/gps_ws/install
Creating symlink "/home/jana/gps_ws/src/CMakeLists.txt" pointing to "/opt/ros/melodic/share/catkin/cmake/topLevel.cmake"
####
#### Running command: "cmake /home/jana/gps_ws/src -DCATKIN_DEVEL_PREFIX=/home/jana/gps_ws/devel -DCMAKE_INSTALL_PREFIX=/home/jana/gps_ws/inst
all -G Unix Makefiles" in "/home/jana/gps_ws/build"
####
-- The C compiler identification is GNU 7.5.0
-- The CXX compiler identification is GNU 7.5.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Using CATKIN_DEVEL_PREFIX: /home/jana/gps_ws/devel
-- Using CMAKE_PREFIX_PATH: /opt/ros/melodic
-- This workspace overlays: /opt/ros/melodic
-- Found PythonInterp: /usr/bin/python2 (found suitable version "2.7.15", minimum required is "2")
-- Using PYTHON_EXECUTABLE: /usr/bin/python2
-- Using Debian Python package layout
-- Using empy: /usr/bin/empy
-- Using CATKIN_ENABLE_TESTING: ON
-- Call enable_testing()

```

Figure 68: ROS - Workspace

- Now we are ready to run the software, but before it we need to prepare the Marvelmind system. Therefore, connect the mobile beacon (hedgehog) by the USB cable to the USB port of your ROS machine. Execute command in terminal to find the virtual serial port used by the hedgehog:

```
$ ls /dev/ttyACM*
```

- Run the node 'hedge\_rcv\_bin' for receiving data from hedgehog as shown in figure 69

Note: The parameter of the running program is '/dev/ttyACM0', which is the name of the virtual serial port detected by previous command. (If the port is '/dev/ttyACM0', this parameter can be skipped.)

```

es Terminal Mon 10:01
Jana@Jana-ThinkPad-T440p: ~/gps_ws
File Edit View Search Terminal Tabs Help
roscore http://jana-Thi... x Jana@Jana-ThinkPad-T... x Jana@Jana-ThinkPad-T... x Jana@Jana-ThinkPad-T... x Jana@Jana-ThinkPad-T... x Jana@Jana-ThinkPad-T... x
^Cstopping
Jana@Jana-ThinkPad-T440p:~/gps_ws$ rosrn marvelmind_nav hedge_rcv_bin /dev/ttyACM0
Opened serial port /dev/ttyACM0 with baudrate 9600
[ INFO] [1643616045.799194770]: IMU fusion: Timestamp: 00169596, X=-0.078 Y= -0.554 Z=0.000 q=0.249,0.017,-0.007,-0.968 v=0.039,0.003,-0.00
8 a=-0.321,0.226,0.567
[ INFO] [1643616045.803776030]: Address: 4, timestamp: 169721, 169721, X=-0.078 Y= -0.554 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1643616045.810574407]: IMU fusion: Timestamp: 00169721, X=-0.077 Y= -0.554 Z=0.000 q=0.249,0.021,-0.009,-0.968 v=0.038,0.003,-0.00
9 a=-0.016,0.011,0.597
[ INFO] [1643616045.819663004]: IMU fusion: Timestamp: 00169721, X=-0.077 Y= -0.554 Z=0.000 q=0.249,0.022,-0.005,-0.968 v=0.035,0.003,-0.00
9 a=-0.013,-0.059,0.537
[ INFO] [1643616045.828650927]: IMU fusion: Timestamp: 00169721, X=-0.077 Y= -0.554 Z=0.000 q=0.249,0.019,-0.006,-0.968 v=0.031,0.002,-0.00
9 a=-0.024,-0.013,0.534
[ INFO] [1643616045.838761179]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.554 Z=0.000 q=0.249,0.022,-0.009,-0.968 v=0.026,0.001,-0.00
9 a=-0.033,0.014,0.557
[ INFO] [1643616045.848865509]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.554 Z=0.000 q=0.249,0.019,-0.011,-0.968 v=0.018,0.000,-0.00
9 a=0.026,0.028,0.558
[ INFO] [1643616045.859639762]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.553 Z=0.000 q=0.249,0.021,-0.008,-0.968 v=0.009,0.000,-0.00
8 a=-0.014,-0.030,0.557
[ INFO] [1643616045.868749552]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.553 Z=0.000 q=0.249,0.018,-0.005,-0.968 v=0.001,-0.001,-0.0
07 a=0.029,-0.012,0.576
[ INFO] [1643616045.878799968]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.553 Z=0.000 q=0.248,0.022,-0.005,-0.968 v=-0.005,-0.002,-0.
007 a=-0.061,-0.010,0.535
[ INFO] [1643616045.888747419]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.553 Z=0.000 q=0.248,0.020,-0.009,-0.968 v=-0.010,-0.002,-0.
006 a=-0.059,-0.024,0.557
[ INFO] [1643616045.898839651]: IMU fusion: Timestamp: 00169721, X=-0.076 Y= -0.554 Z=0.000 q=0.248,0.017,-0.006,-0.968 v=-0.014,-0.003,-0.
006 a=0.006,-0.003,0.554
[ INFO] [1643616045.920851945]: IMU fusion: Timestamp: 00169721, X=-0.077 Y= -0.554 Z=-0.001 q=0.248,0.023,-0.000,-0.968 v=-0.017,-0.003,-0
.006 a=-0.055,-0.101,0.514
[ INFO] [1643616045.923657546]: Address: 4, timestamp: 169846, 125, X=-0.077 Y= -0.555 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1643616045.929631278]: IMU fusion: Timestamp: 00169846, X=-0.077 Y= -0.554 Z=-0.001 q=0.248,0.021,-0.004,-0.969 v=-0.017,-0.005,-0
.006 a=-0.018,-0.062,0.495
[ INFO] [1643616048.919178007]: IMU fusion: Timestamp: 00172719, X=-0.080 Y= -0.551 Z=-0.002 q=0.224,0.022,-0.014,-0.974 v=-0.102,0.051,-0.
030 a=-0.009,0.063,0.520
[ INFO] [1643616048.920862543]: Address: 4, timestamp: 172843, 2997, X=-0.076 Y= -0.550 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1643616048.930895152]: IMU fusion: Timestamp: 00172843, X=-0.082 Y= -0.550 Z=-0.002 q=0.224,0.023,-0.009,-0.974 v=-0.103,0.054,-0.
027 a=0.002,-0.032,0.500
[ INFO] [1643616048.940660031]: IMU fusion: Timestamp: 00172843, X=-0.083 Y= -0.549 Z=-0.002 q=0.224,0.019,-0.008,-0.974 v=-0.100,0.055,-0

```

Figure 69: ROS - hedge\_rcv\_bin

Note 1: You might get a message like 'unable to open serial connection' even if the serial port is present. This could mean there is no permissions to access this port. You can get all permissions by command 'sudo chmod 0777 /dev/ttyACM0'. But permissions will be lost after the next reboot.

Note 2: First value in square brackets, of the outcome, is a ROS timestamp, then hedgehog timestamp in milliseconds between position samples, coordinates X,Y,Z in meters, and byte of flags.

Note 3: The node 'hedge\_rcv\_bin' works as ROS publisher, it sends the message with location data to the topics named 'hedge\_pos', 'hedge\_pos\_a' and 'hedge\_pos\_ang'. 'hedge\_pos\_ang' is most new version of the topic; it includes address of mobile beacon and orientation angle of paired beacons.

- The package also contains another node 'subscriber\_test', which is working as ROS subscriber and receiving data from all the topics. This node can be used for test purposes and as basis for user software. Run the 'subscriber\_test' node in separate terminal as shown in figure 70. The running '**subscriber\_test**' node outputs to the terminal the received location data from the topic '/hedge\_pos'

```
jana@jana-ThinkPad-T440p:~/gps_ws$ rosrnrun marvelmind_nav subscriber_test
[ INFO] [1643616100.370083518]: IMU fusion: Timestamp: 00224184, X=-0.022 Y= -0.581 Z=0.001 q=-0.201,0.009,-0.015,-0.979 v=0.063,0.041,0.04
2 a=0.049,-0.026,0.595
[ INFO] [1643616100.383341817]: IMU fusion: Timestamp: 00224184, X=-0.021 Y= -0.580 Z=0.002 q=-0.201,0.015,-0.015,-0.979 v=0.066,0.042,0.04
6 a=-0.035,0.013,0.515
[ INFO] [1643616100.385270881]: Hedgehog data: Address= 4, timestamp= 224309, X=-0.029 Y= -0.585 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1643616100.390189148]: IMU fusion: Timestamp: 00224309, X=-0.021 Y= -0.580 Z=0.002 q=-0.201,0.014,-0.017,-0.979 v=0.064,0.040,0.04
7 a=-0.008,-0.007,0.517
[ INFO] [1643616100.400286655]: IMU fusion: Timestamp: 00224309, X=-0.021 Y= -0.580 Z=0.003 q=-0.201,0.015,-0.022,-0.979 v=0.061,0.041,0.04
7 a=-0.003,0.064,0.578
[ INFO] [1643616100.410156413]: IMU fusion: Timestamp: 00224309, X=-0.020 Y= -0.580 Z=0.003 q=-0.201,0.013,-0.019,-0.979 v=0.056,0.042,0.04
6 a=-0.001,0.021,0.597
[ INFO] [1643616100.419604722]: IMU fusion: Timestamp: 00224309, X=-0.020 Y= -0.579 Z=0.003 q=-0.201,0.009,-0.018,-0.979 v=0.050,0.043,0.04
5 a=0.047,-0.017,0.637
[ INFO] [1643616100.429579715]: IMU fusion: Timestamp: 00224309, X=-0.021 Y= -0.579 Z=0.004 q=-0.201,0.014,-0.017,-0.979 v=0.041,0.043,0.04
2 a=-0.030,0.002,0.677
[ INFO] [1643616100.439190421]: IMU fusion: Timestamp: 00224309, X=-0.021 Y= -0.579 Z=0.004 q=-0.201,0.016,-0.020,-0.979 v=0.029,0.043,0.03
8 a=-0.032,0.066,0.718
[ INFO] [1643616100.449418128]: IMU fusion: Timestamp: 00224309, X=-0.022 Y= -0.580 Z=0.004 q=-0.201,0.013,-0.018,-0.979 v=0.015,0.043,0.03
3 a=-0.017,0.031,0.696
[ INFO] [1643616100.459236446]: IMU fusion: Timestamp: 00224309, X=-0.023 Y= -0.580 Z=0.004 q=-0.201,0.010,-0.016,-0.979 v=0.002,0.042,0.02
8 a=0.017,-0.014,0.615
[ INFO] [1643616100.468798668]: IMU fusion: Timestamp: 00224309, X=-0.024 Y= -0.580 Z=0.003 q=-0.201,0.010,-0.020,-0.979 v=-0.008,0.042,0.0
24 a=0.007,0.021,0.677
[ INFO] [1643616100.476097351]: IMU fusion: Timestamp: 00224309, X=-0.025 Y= -0.580 Z=0.003 q=-0.202,0.010,-0.016,-0.979 v=-0.016,0.042,0.0
22 a=-0.281,0.344,0.667
[ INFO] [1643616100.481200049]: IMU fusion: Timestamp: 00224309, X=-0.025 Y= -0.580 Z=0.003 q=-0.202,0.011,-0.017,-0.979 v=-0.021,0.042,0.0
20 a=0.024,-0.026,0.638
[ INFO] [1643616100.488246068]: IMU fusion: Timestamp: 00224309, X=-0.026 Y= -0.580 Z=0.003 q=-0.202,0.014,-0.018,-0.979 v=-0.024,0.042,0.0
20 a=-0.321,0.304,0.547
[ INFO] [1643616103.48995438]: IMU fusion: Timestamp: 00227307, X=-0.028 Y= -0.571 Z=0.001 q=-0.226,0.018,-0.015,-0.974 v=0.075,0.048,0.00
5 a=-0.063,-0.007,0.577
[ INFO] [1643616103.506652898]: IMU fusion: Timestamp: 00227307, X=-0.027 Y= -0.570 Z=0.001 q=-0.226,0.012,-0.016,-0.974 v=0.082,0.052,0.00
5 a=0.060,0.004,0.536
[ INFO] [1643616103.509091795]: IMU fusion: Timestamp: 00227307, X=-0.026 Y= -0.570 Z=0.001 q=-0.226,0.013,-0.016,-0.974 v=0.082,0.053,0.00
```

Figure 70: ROS - Subscriber\_test

- Also, this node works as publisher and sends data to topic "visualization\_marker". This allows to view the position in the standard ROS software 'rviz'. Run the 'rviz' in separate terminal. The GUI window, shown

in figure 71, should appear. To see the hedgehog, assign 'my\_frame' to the 'Fixed frame' parameter, and the Marker parameter is ticked.

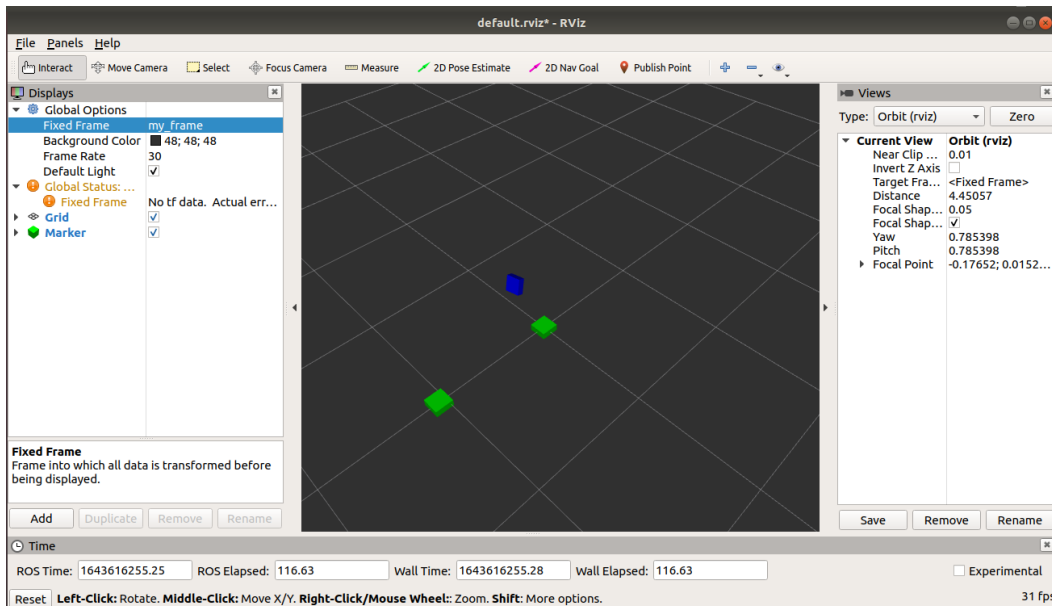


Figure 71: Rviz - Stationary and Mobile Beacons Markers

#### iv. GPS2ODOM Package

After receiving the IMU and the GPS data from the marvelmind\_nav ROS package, we worked on taking these data and writing a script that provides the transformations and odometry data that are later on needed by the navigation package.

- We formed a package 'gps2odom' and named the created python code 'gps\_to\_odom.py'. We run our code while also running the code of marvelmind package that publishes the data. (without running the subscriber test of marvelmind)

```
jana@jana-ThlnkPad-T440p:~/gps_ws$ rosrunc gps2odom gps_to_odom.py
jana@jana-ThlnkPad-T440p:~/gps_ws$ rosrunc gps2odom gps_to_odom.py
```

Figure 72: ROS - Run gps2odom Package

- After running our code, we were able to get 3 reference frames, the map, odom and base\_link. The map frame represents the map's physical origin,



the odom represents the GPS system origin, **and the base\_link** represents the mobile beacon.

Recorded at time: 1643616483.47

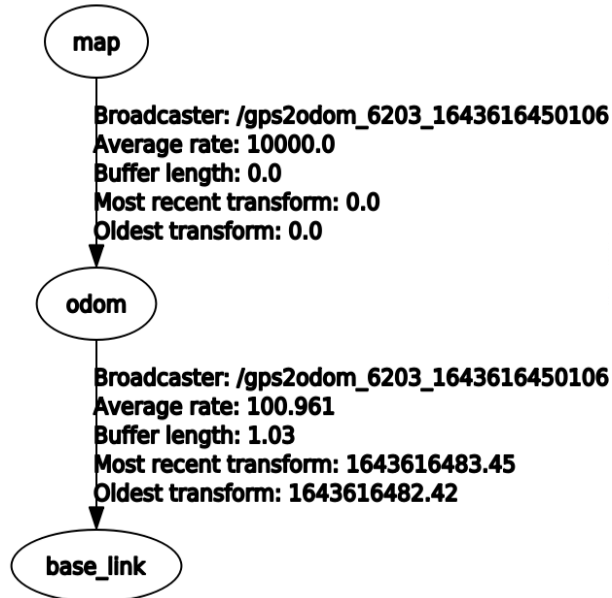


Figure 73: ROS - Frames in rqt tree

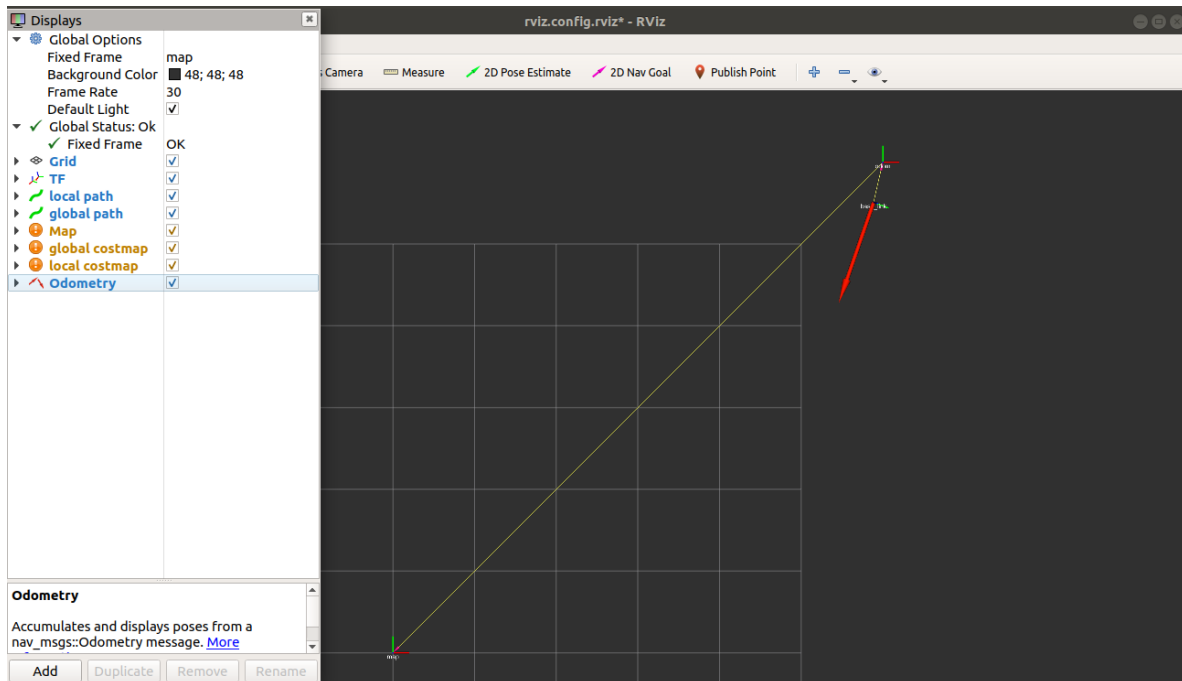
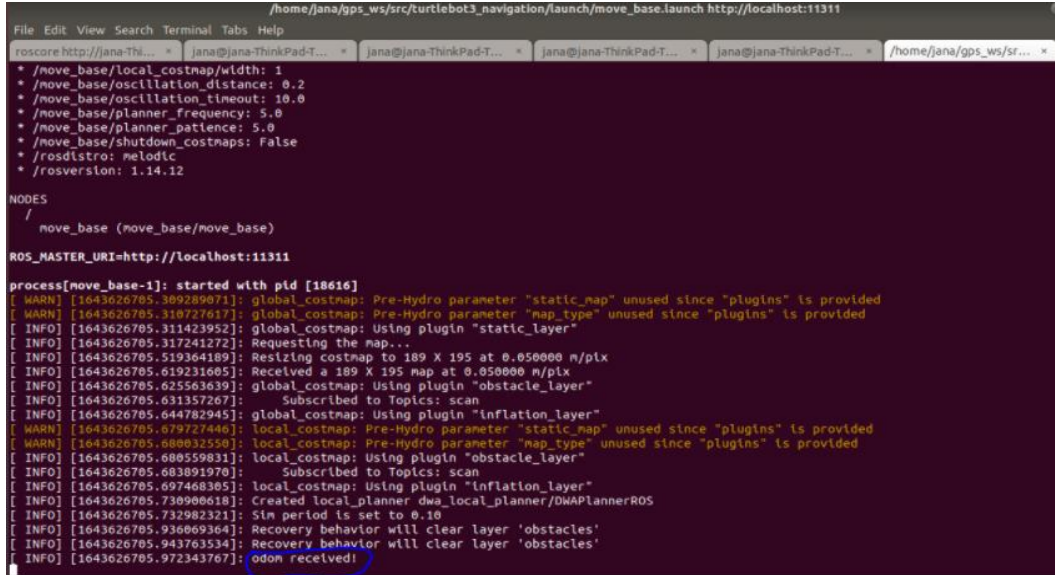


Figure 74: Rviz - Frames TF

## v. Navigation Package

To perform navigation, we downloaded the standard ROS navigation package (can be downloaded from any turtlebot3 demo package). We also got a ready map to test the navigation on it, and edited the parameters in the launch file, and parameters of local and global cost map in the yaml files. Then we launch the move\_base node after running the gps2odom code and marvelmind publisher code.



```
~/home/jana/gps_ws/src/turtlebot3_navigation/launch/move_base.launch http://localhost:11311
File Edit View Search Terminal Tabs Help
roscore http://jana-Thi...  Jana@jana-ThinkPad-T...  Jana@jana-ThinkPad-T...  Jana@jana-ThinkPad-T...  Jana@jana-ThinkPad-T...  /home/jana/gps_ws/sr...
* /move_base/local_costmap/width: 1
* /move_base/oscillation_distance: 0.2
* /move_base/oscillation_timeout: 10.0
* /move_base/planner_frequency: 5.0
* /move_base/planner_patience: 5.0
* /move_base/shutdown_costmaps: False
* /roscpp: melodic
* /rosversion: 1.14.12

NODES
/
  move_base (move_base/move_base)

ROS_MASTER_URI=http://localhost:11311

process[move_base-1]: started with pid [18616]
[ WARN ] [1643626705.389289971]: global_costmap: Pre-Hydro parameter "static_map" unused since "plugins" is provided
[ WARN ] [1643626705.318727617]: global_costmap: Pre-Hydro parameter "map_type" unused since "plugins" is provided
[ INFO ] [1643626705.311423952]: global_costmap: Using plugin "static_layer"
[ INFO ] [1643626705.317241272]: Requesting the map...
[ INFO ] [1643626705.519364189]: Resizing costmap to 189 X 195 at 0.050000 m/px
[ INFO ] [1643626705.619231605]: Received a 189 X 195 map at 0.050000 m/px
[ INFO ] [1643626705.625563639]: global_costmap: Using plugin "obstacle_layer"
[ INFO ] [1643626705.631357267]: Subscribed to Topics: scan
[ INFO ] [1643626705.644782945]: global_costmap: Using plugin "inflation_layer"
[ WARN ] [1643626705.679727446]: local_costmap: Pre-Hydro parameter "static_map" unused since "plugins" is provided
[ WARN ] [1643626705.688832558]: local_costmap: Pre-Hydro parameter "map_type" unused since "plugins" is provided
[ INFO ] [1643626705.688559831]: local_costmap: Using plugin "obstacle_layer"
[ INFO ] [1643626705.683891970]: Subscribed to Topics: scan
[ INFO ] [1643626705.697468305]: local_costmap: Using plugin "inflation_layer"
[ INFO ] [1643626705.730906618]: Created local_planner dwa_local_planner/DWAPlannerROS
[ INFO ] [1643626705.732982321]: Sin period is set to 0.10
[ INFO ] [1643626705.936069364]: Recovery behavior will clear layer 'obstacles'
[ INFO ] [1643626705.943763534]: Recovery behavior will clear layer 'obstacles'
[ INFO ] [1643626705.972343767]: odometry received!
```

Figure 75: ROS - Odometry Received

- Then, we add the TF, Map, global and local costmap , and odometry in rviz. The white area is the local map while the global map is the bigger highlighted one.

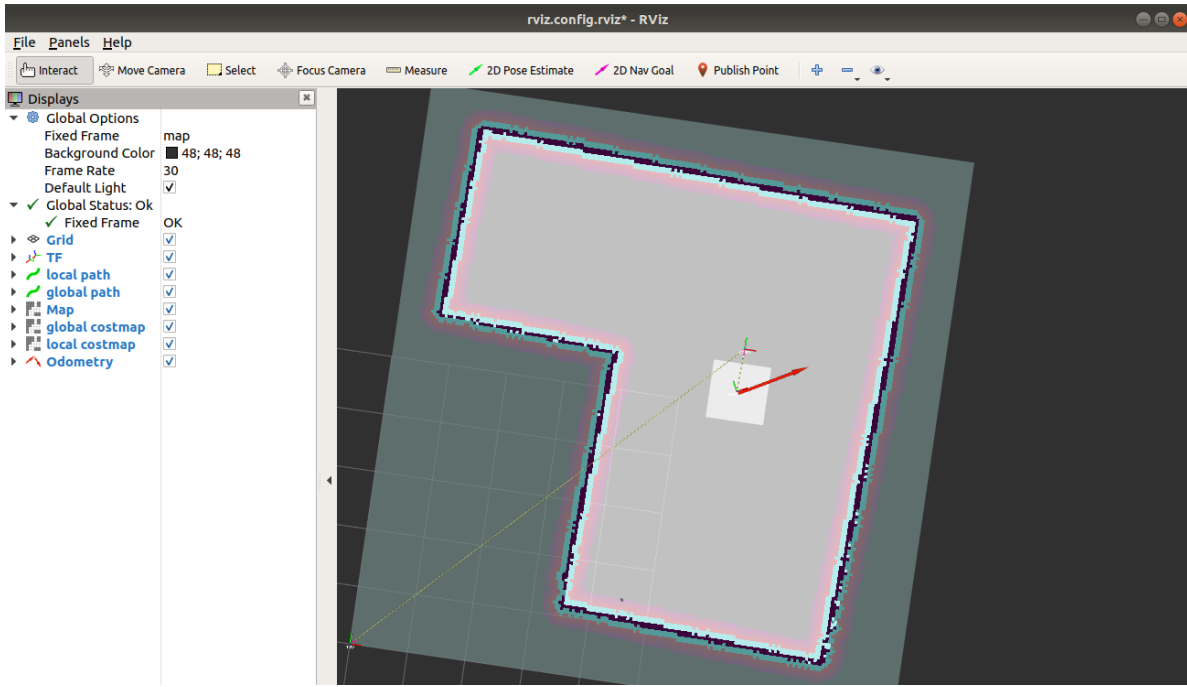


Figure 76: Rviz - Demo Map With Frames

- Then to test the path planning, we gave the mobile beacon (base\_link) a 2D Nav Goal and the cmd\_vel/ topic started receiving motion commands so if we were working on a real robot, it will move. The green trace in figure 76 shows the motion of the mobile beacon.



- Edit the bashrc of the ubuntu, the ROS\_MASTER and ROS\_HOSTNAME should include the WiFi IP address.

```

# Ubuntu: see warnings and errors
#export GCC_COLORS='error=01;31:warning=01;35:note=01;36:caret=01;32:locus=01:quote=01'

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands. Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "${ $? = 0 } && echo terminal || echo error" "$(history|tail -n1|sed -e '\''s/^[0-9]\+\s*//;s/[:&]]\s*alert$//'\''")'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

source /opt/ros/melodic/setup.bash
#source ~/catkin_ws/devel/setup.bash

export ROS_MASTER_URI=http://192.168.2.129:11311
export ROS_HOSTNAME=192.168.2.129

```

Figure 79: Ubuntu bashrc

```

alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands. Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "${ $? = 0 } && echo terminal || echo error" "$(history|tail -n1|sed -e '\''s/^[0-9]\+\s*//;s/[:&]]\s*alert$//'\''")'

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

source /opt/ros/melodic/setup.bash
source /opt/ros/melodic/setup.bash
source ~/catkin_ws/devel/setup.bash
source ~/catkin_ws/devel/setup.bash
source /opt/ros/melodic/setup.bash
#export ROS_MASTER_URI=http://10.0.6.253:11311
#export ROS_HOSTNAME=10.0.6.190

export ROS_MASTER_URI=http://192.168.2.129:11311
export ROS_HOSTNAME=192.168.2.168
export TURTLEBOT3_MODEL=burger

```

Figure 80: Raspberry Pi bashrc

- Edit the bashrc of the raspberry pi as shown in figure 80. We assign the WiFi's IP address to the ROS\_MASTER while the ROS\_HOSTNAME should include the raspberry pi's IP address. Also, we included the turtlebot3

model "burger" in the bashrc to avoid exporting it at every bringup (export TURTLEBOT3\_MODEL=burger)

- As for the beacons, we placed the mobile beacon on the turtlebot3 and the stationary ones in the room in a way to be collinear in the x-y plane, to form an axis. (stationary beacons should be higher from the mobile beacon). Then repeat the configuration on the Dashboard, and calibrate the IMU of the mobile beacon (this step is done each time we change the working environment).

### vii. Turtlebot3 Implementation: Sending Goals from Rviz

- Add a copy of ros\_marvelmind\_package in catkin\_ws in raspberry pi. then apply catkin\_make install on raspberry pi.

```
ros@ros-desktop:~/catkin_ws$ catkin_make install
Base path: /home/ros/catkin_ws
Source space: /home/ros/catkin_ws/src
Build space: /home/ros/catkin_ws/build
Devel space: /home/ros/catkin_ws/devel
Install space: /home/ros/catkin_ws/install
####
#### Running command: "make cmake_check_build_system" in "/home/ros/catkin_ws/build"
####
####
#### Running command: "make install -j4 -l4" in "/home/ros/catkin_ws/build"
####
```

Figure 81: ROS - Catkin\_make Install Command

- Then source the workspace (catkin\_ws in raspberry pi) and run the hedge\_rcv\_bin code

Note: The node 'hedge\_rcv\_bin' works as ROS publisher, it sends the message with location data to the topics named 'hedge\_pos', 'hedge\_pos\_a' and 'hedge\_pos\_ang'

Note 1: always use ls /dev/ttyACM\* to check the port

Note 2: always check that the baudrate of the IMU in the Dashboard is also 9600 else running the code won't output any message.

```

ros@ros-desktop:~/catkin_ws$ source devel/setup.bash
ros@ros-desktop:~/catkin_ws$ rosrn marvelmind_nav hedge_rcv_bin
Opened serial port /dev/ttyACM0 with baudrate 9600
^Cstopping
ros@ros-desktop:~/catkin_ws$ ls /dev/ttyACM*
/dev/ttyACM0 /dev/ttyACM1
ros@ros-desktop:~/catkin_ws$ rosrn marvelmind_nav hedge_rcv_bin /dev/ttyACM1 9600
Opened serial port /dev/ttyACM1 with baudrate 9600
[ INFO] [1645373404.011826973]: Address: 4, timestamp: 706594, 706594, X=0.576 Y= -1.096 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.045836593]: IMU fusion: Timestamp: 00706594, X=0.577 Y= -1.097 Z=0.000 q=0.517,-0.053,0.032,0.854 v=-0.014,-0.003,0.002
a=-0.001,0.008,0.587
[ INFO] [1645373404.066347762]: Address: 4, timestamp: 706719, 125, X=0.576 Y= -1.096 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.103335823]: IMU fusion: Timestamp: 00706719, X=0.573 Y= -1.096 Z=0.000 q=0.517,-0.049,0.034,0.854 v=-0.029,0.000,0.004
a=0.044,-0.004,0.644
[ INFO] [1645373404.211185211]: Address: 4, timestamp: 706844, 125, X=0.576 Y= -1.096 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.250364889]: IMU fusion: Timestamp: 00706844, X=0.573 Y= -1.096 Z=0.000 q=0.519,-0.055,0.028,0.853 v=-0.014,0.000,0.003
a=-0.029,-0.011,0.604
[ INFO] [1645373404.310463600]: IMU fusion: Timestamp: 00706844, X=0.574 Y= -1.096 Z=0.000 q=0.519,-0.057,0.030,0.852 v=-0.014,0.000,0.003
a=-0.133,0.055,0.578
[ INFO] [1645373404.360326050]: Address: 4, timestamp: 706969, 125, X=0.576 Y= -1.096 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.400583594]: IMU fusion: Timestamp: 00706969, X=0.574 Y= -1.096 Z=0.000 q=0.520,-0.050,0.030,0.852 v=-0.012,0.000,0.004
a=0.012,-0.009,0.579
[ INFO] [1645373404.450678284]: Address: 4, timestamp: 707095, 126, X=0.576 Y= -1.096 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.488647283]: IMU fusion: Timestamp: 00707095, X=0.573 Y= -1.096 Z=0.000 q=0.520,-0.057,0.035,0.851 v=-0.016,0.000,0.004
a=-0.018,0.019,0.679
[ INFO] [1645373404.560107569]: IMU fusion: Timestamp: 00707095, X=0.573 Y= -1.095 Z=0.000 q=0.521,-0.058,0.033,0.851 v=-0.015,0.000,0.003
a=-0.105,0.123,0.740
[ INFO] [1645373404.609946165]: Address: 4, timestamp: 707220, 125, X=0.576 Y= -1.095 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.647856675]: IMU fusion: Timestamp: 00707220, X=0.574 Y= -1.094 Z=0.000 q=0.521,-0.047,0.029,0.851 v=-0.001,0.020,0.001
a=0.023,-0.059,0.639
[ INFO] [1645373404.698729283]: Address: 4, timestamp: 707345, 125, X=0.575 Y= -1.095 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.796941212]: IMU fusion: Timestamp: 00707345, X=0.573 Y= -1.093 Z=0.000 q=0.522,-0.052,0.030,0.850 v=-0.022,-0.003,0.000
a=0.000,-0.003,0.624
[ INFO] [1645373404.847194184]: Address: 4, timestamp: 707470, 125, X=0.575 Y= -1.095 Z=0.000 Angle: 0.0 flags=2
[ INFO] [1645373404.886329122]: IMU fusion: Timestamp: 00707470, X=0.572 Y= -1.094 Z=0.000 q=0.523,-0.052,0.027,0.850 v=-0.015,-0.003,0.000

```

Figure 82: ROS - Run Marvelmind Package

- To make sure that the message is published, run the following command:

```

^Cjane@jane-ThinkPad-T440p:~$ rostopic echo /hedge_imu_fusion
timestamp_ms: 710970
x_m: 0.573
y_m: -1.092
z_m: 0.0
qw: 0.5461
qx: -0.0513
qy: 0.032
qz: 0.8354
vx: -0.002
vy: 0.012
vz: 0.0
ax: -0.048
ay: 0.022
az: 0.659
---
timestamp_ms: 711095
x_m: 0.573
y_m: -1.092
z_m: 0.0
qw: 0.5466
qx: -0.0509
qy: 0.0352
qz: 0.835

```

Figure 83: ROS - Messages Output from hedge\_imu\_fusion Node

- Then bringup the turtlebot3 as shown in figure 84

```
ros@ros-desktop:~/catkin_ws
turtlebot3_bringup turtlebot3_msgs
ros@ros-desktop:~/catkin_ws$ roslaunch turtlebot3_bringup turtlebot3_core.launch
... logging to /home/ros/.ros/log/6f184fee-9264-11ec-9c41-5c514fc93079/roslaunch-ros-desktop-4766.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.2.168:45737/

SUMMARY
=====
PARAMETERS
* /roscpp: melodic
* /rosversion: 1.14.6
* /turtlebot3_core/serial_baud: 115200
* /turtlebot3_core/serial_port: /dev/ttyACM0
* /turtlebot3_core/serial_prefix:

NODES
/
  turtlebot3_core (roscpp/serial_node.py)

ROS_MASTER_URI=http://192.168.2.129:11311

process[turtlebot3_core-1]: started with pid [4776]
[INFO] [1645373822.261312]: ROS Serial Python Node
[INFO] [1645373822.383763]: Connecting to /dev/ttyACM0 at 115200 baud
[INFO] [1645373824.530589]: Requesting topics...
[INFO] [1645373824.584800]: Note: publish buffer size is 1024 bytes
[INFO] [1645373824.593997]: Setup publisher on sensor_state [turtlebot3_msgs/SensorState]
[INFO] [1645373824.622820]: Setup publisher on firmware_version [turtlebot3_msgs/VersionInfo]
[INFO] [1645373825.038567]: Setup publisher on imu [sensor_msgs/Imu]
[INFO] [1645373825.101937]: Setup publisher on cmd_vel_rc100 [geometry_msgs/Twist]
[INFO] [1645373825.280038]: Setup publisher on odom [nav_msgs/Odometry]
[INFO] [1645373825.330228]: Setup publisher on joint_states [sensor_msgs/JointState]
[INFO] [1645373825.350474]: Setup publisher on battery_state [sensor_msgs/BatteryState]
```

Figure 84: ROS - Turtlebot3 Bringup

- After bringing up the turtlebot, the topics available include odom from the encoders, and we assigned odom previously to be the topic of odometry from the GPS. In this way, there will be overlapping problem between the odometry data from encoders of the motors and odometry from the gps. To solve this problem, we changed the name of the odometry topic of gps from /odom to /odom\_gps and we changed the frames too from map to map\_gps and from base\_link to base\_link\_gps in the parameters.

```
global_costmap_params.yaml
local_costmap_params.yaml
move_base_params.yaml

src > turtlebot3_navigation > param > ! local_costmap_params.yaml
1 local_costmap:
2   global_frame: map_gps
3   robot_base_frame: base_link_gps
4
5   update_frequency: 10.0
6   publish_frequency: 10.0
7   transform_tolerance: 0.5
8
9   static_map: false
10  rolling_window: true
11  width: 1
12  height: 1
13  resolution: 0.05
```

Figure 85: Parameters of Local Costmap Yaml File



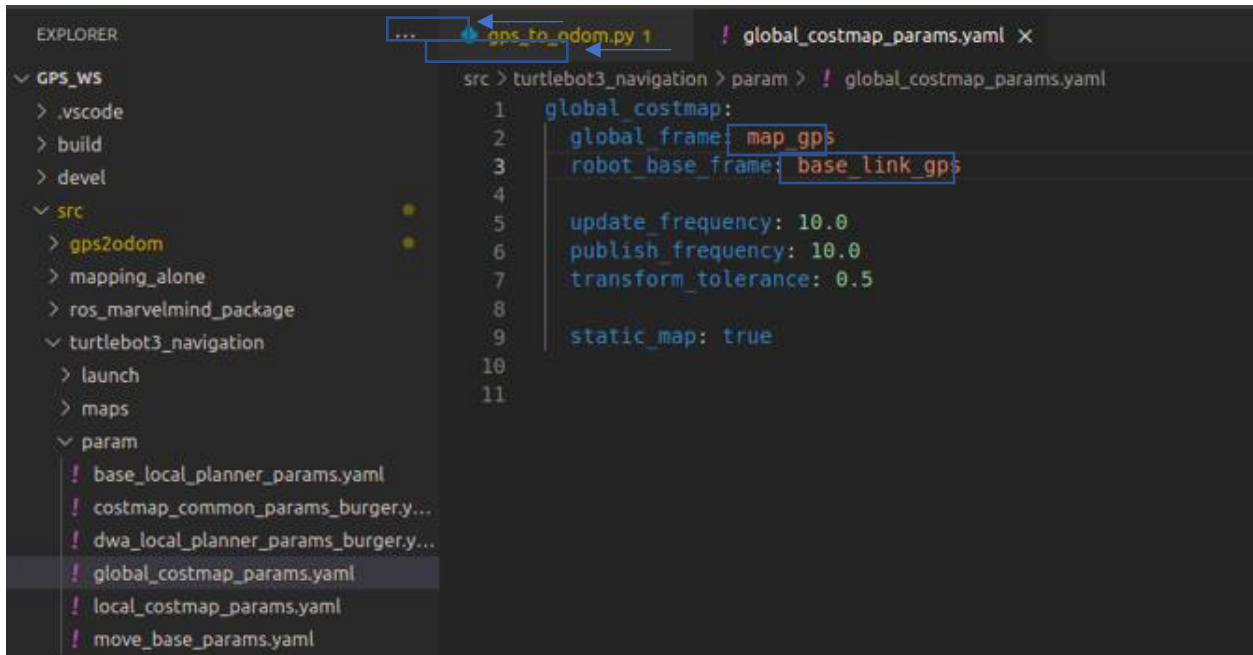


Figure 86: Parameters of Global Costmap Yaml File

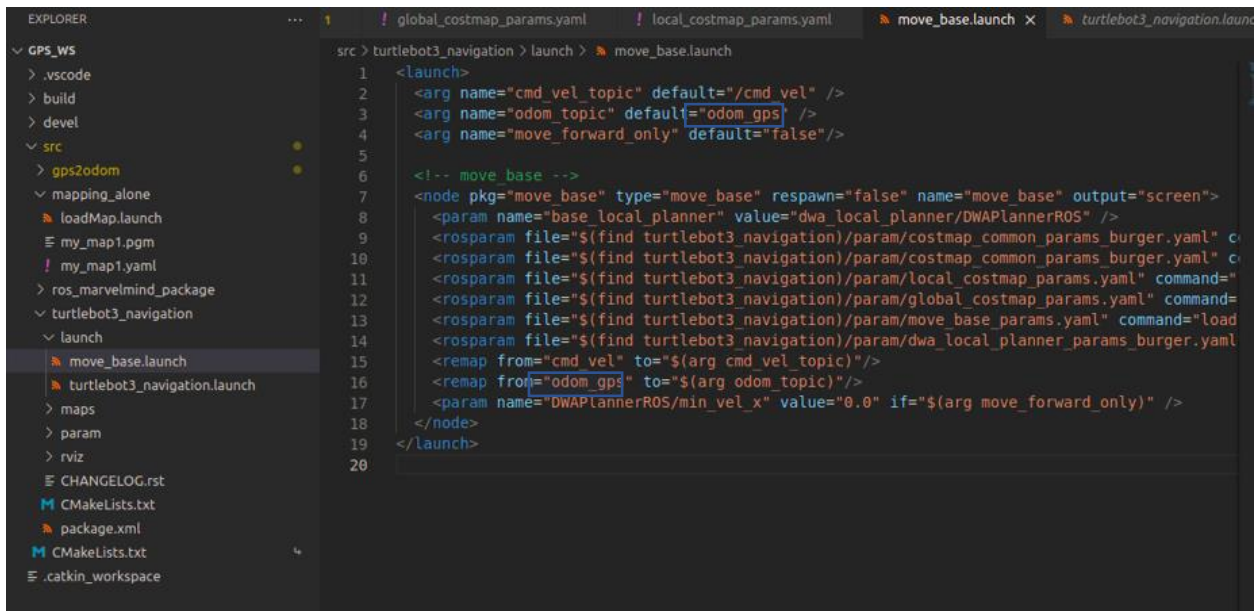


Figure 87: Parameters of move\_base Launch File

- By running the command: `roslaunch rqt_tf_tree rqt_tf_tree`, we get the following `rqt_tf_tree` that validates our work where there is 3 frames from the gps (`map_gps`, `odom_gps` and `base_link_gps`) and another frames from the encoders of the robot's motors. (`odom` and `base_footprint`)

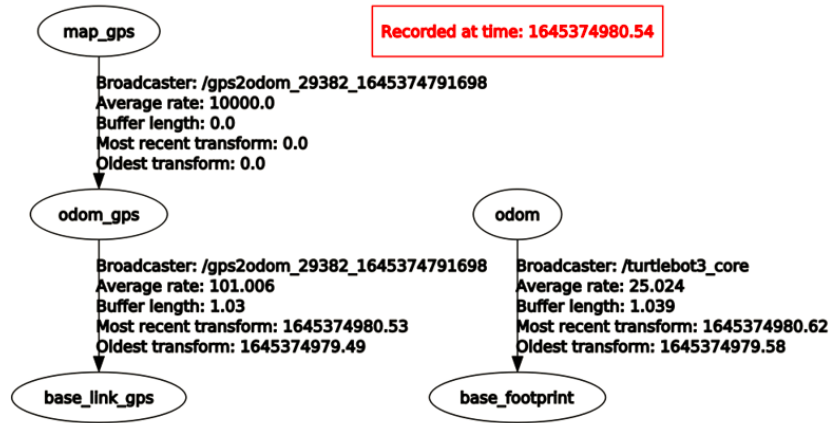


Figure 88: Rqt Tree

- After receiving the IMU and the GPS data from the marvelmind\_nav ROS package, we run the gps\_to\_odom code from gps2odom package that provides the transformations and odometry data that are later on needed by the navigation package.

```
jana@jana-ThinkPad-T440p:~$ rosrund gps2odom gps_to_odom.py
```

Figure 89: ROS - Run GPS2ODOM Python Code

- Then run the turtlebot3\_navigation launch file in the turtlebot3\_navigation package.

```

jana@jana-ThinkPad-T440p:~$ source gps_ws/devel/setup.bash
jana@jana-ThinkPad-T440p:~$ roslaunch turtlebot3_navigation turtlebot3_navigation.launch
... logging to /home/jana/.ros/log/6f184fee-9264-11ec-9c41-5c514fc93079/roslaunch-jana-ThinkPad-T440p-29548.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.

started roslaunch server http://192.168.2.129:46771/

SUMMARY
=====
PARAMETERS
* /move_base/DWAPlannerROS/acc_lim_theta: 3.2
* /move_base/DWAPlannerROS/acc_lim_x: 2.5
* /move_base/DWAPlannerROS/acc_lim_y: 0.0
* /move_base/DWAPlannerROS/controller_frequency: 10.0
* /move_base/DWAPlannerROS/forward_point_distance: 0.0
* /move_base/DWAPlannerROS/goal_distance_bias: 20.0
* /move_base/DWAPlannerROS/latch_xy_goal_tolerance: False
* /move_base/DWAPlannerROS/max_scaling_factor: 0.2
* /move_base/DWAPlannerROS/max_vel_theta: 2.75
* /move_base/DWAPlannerROS/max_vel_trans: 0.22
* /move_base/DWAPlannerROS/max_vel_x: 0.22
* /move_base/DWAPlannerROS/max_vel_y: 0.0
* /move_base/DWAPlannerROS/min_vel_theta: 1.37
* /move_base/DWAPlannerROS/min_vel_trans: 0.11
* /move_base/DWAPlannerROS/min_vel_x: -0.22
* /move_base/DWAPlannerROS/min_vel_y: 0.0
* /move_base/DWAPlannerROS/occdist_scale: 0.02
* /move_base/DWAPlannerROS/oscillation_reset_dist: 0.05
* /move_base/DWAPlannerROS/path_distance_bias: 32.0
* /move_base/DWAPlannerROS/publish_cost_grid_pc: True
* /move_base/DWAPlannerROS/publish_traj_pc: True
* /move_base/DWAPlannerROS/scaling_speed: 0.25
* /move_base/DWAPlannerROS/sin_time: 1.5
  
```

Figure 90: ROS - Launch Turtlebot3 Navigation Launch file

- Afterwards, by giving a navigation goal to the turtlebot3 through Rviz, the turtlebot3 was able to navigate in the map to reach its target. When it reaches the target an indication in the command line will appear as in figure 91.

```

[INFO] [1645375317.016768253]: Got new plan
[INFO] [1645375317.216716882]: Got new plan
[INFO] [1645375317.416714484]: Got new plan
[INFO] [1645375317.616807752]: Got new plan
[INFO] [1645375317.816710529]: Got new plan
[INFO] [1645375318.016787044]: Got new plan
[INFO] [1645375318.218209235]: Got new plan
[INFO] [1645375318.416802847]: Got new plan
[INFO] [1645375318.616772880]: Got new plan
[INFO] [1645375318.816818702]: Got new plan
[INFO] [1645375319.016710860]: Got new plan
[INFO] [1645375319.216688248]: Got new plan
[INFO] [1645375319.416711064]: Got new plan
[INFO] [1645375319.616708039]: Got new plan
[INFO] [1645375319.816713527]: Got new plan
[INFO] [1645375320.016751346]: Got new plan
[INFO] [1645375320.216770590]: Got new plan
[INFO] [1645375320.416693328]: Got new plan
[INFO] [1645375320.616732473]: Got new plan
[INFO] [1645375320.816789052]: Got new plan
[INFO] [1645375321.016703323]: Got new plan
[INFO] [1645375321.216717022]: Got new plan
[INFO] [1645375321.416714615]: Got new plan
[INFO] [1645375321.616680084]: Got new plan
[INFO] [1645375321.816709519]: Got new plan
[INFO] [1645375322.016717771]: Got new plan
[INFO] [1645375322.016824638]: Goal reached

```

Figure 91: ROS - Goal Reached Command Line Indication

- Note: the tb3 was moving uncontrollably fast, so we changed its speed in the `dwa_local_planner_params_burger.yaml`

```

src > turtlebot3_navigation > param > ! dwa_local_planner_params_burger.yaml
1  DWAPLannerROS:
2
3  # Robot Configuration Parameters
4  max_vel_x: 0.05
5  min_vel_x: -0.05
6
7  max_vel_y: 0.0
8  min_vel_y: 0.0
9
10 # The velocity when robot is moving in a straight line
11 max_vel_trans: 0.05
12 min_vel_trans: 0.05
13
14 max_vel_theta: 0.8
15 min_vel_theta: 0.8
16
17 acc_lim_x: 2.5
18 acc_lim_y: 0.0
19 acc_lim_theta: 3.2
20
21 # Goal Tolerance Parameters
22 xy_goal_tolerance: 0.2
23 yaw_goal_tolerance: 0.17
24 latch_xy_goal_tolerance: false
25
26 # Forward Simulation Parameters
27 sim_time: 1.5
28 vx_samples: 20
29 vy_samples: 0
30 vth_samples: 40
31 controller_frequency: 10.0

```

Figure 92: Speed Parameters of `dwa_local_planner_params_burger` Yaml File

### viii. Turtlebot3 Implementation: Sending Goals from Python Code

After applying navigation by sending goals from Rviz, now it will be applied through code.

We run again the turtlebot3\_bringup launch file, turtlebot3\_navigation launch file, gps\_to\_odom python file, and the nav\_goal python file shown in figure 93 (we will send the goals through an action-client server method).

```
def movebase_client():

    # Create an action client called "move_base" with action definition file "MoveBaseAction"
    client = actionlib.SimpleActionClient('move_base', MoveBaseAction)

    # Waits until the action server has started up and started listening for goals.
    client.wait_for_server()

    # Creates a new goal with the MoveBaseGoal constructor
    goal = MoveBaseGoal()
    goal.target_pose.header.frame_id = "map_gps"
    goal.target_pose.header.stamp = rospy.Time.now()
    # Move 0.5 meters forward along the x axis of the "map" coordinate frame
    goal.target_pose.pose.position.x = 4
    goal.target_pose.pose.position.y = 5
    # No rotation of the mobile base frame w.r.t. map frame
    goal.target_pose.pose.orientation.w = 1.0

    # Sends the goal to the action server.
    client.send_goal(goal)
    # Waits for the server to finish performing the action.
    wait = client.wait_for_result()
    # If the result doesn't arrive, assume the Server is not available
    if not wait:
        rospy.logerr("Action server not available!")
        rospy.signal_shutdown("Action server not available!")
    else:
```

Figure 93: Send Goal Through Python File

### ix. Real Robot: Teleop

After building the drivetrain of the robot and adding the Raspberry pi and the Arduino boards, the robot is ready to establish communication between the two boards and apply all the steps listed above. As mentioned earlier, first connect the Raspberry pi on a separate monitor to check its IP address. Then, connect it to ROS by executing through a ROS

terminal: ssh ros@*RaspberryIPAdress*. Afterwards, edit the bashrc file of ubuntu where both the Master and the Host are the main WiFi's IP address and edit the bashrc of the Raspberry pi where the Master is the WiFi's IP address, and the Host is the Raspberry's IP address. Then apply the following steps:

- Apply roscore in ubuntu

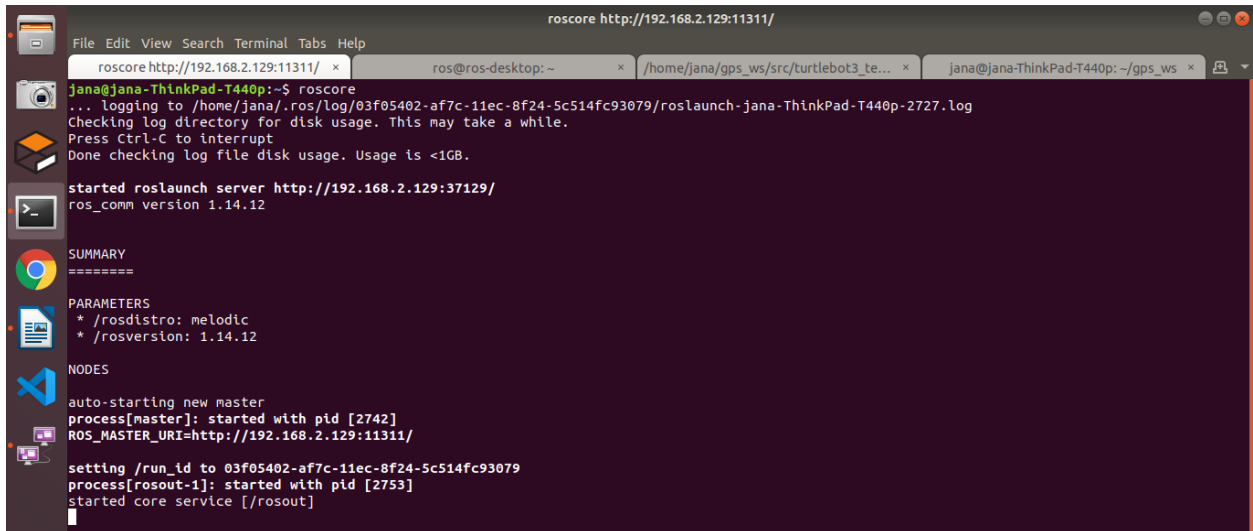


Figure 94: Roscore of Ubuntu

- Source the bashrc of ubuntu and source the workspace

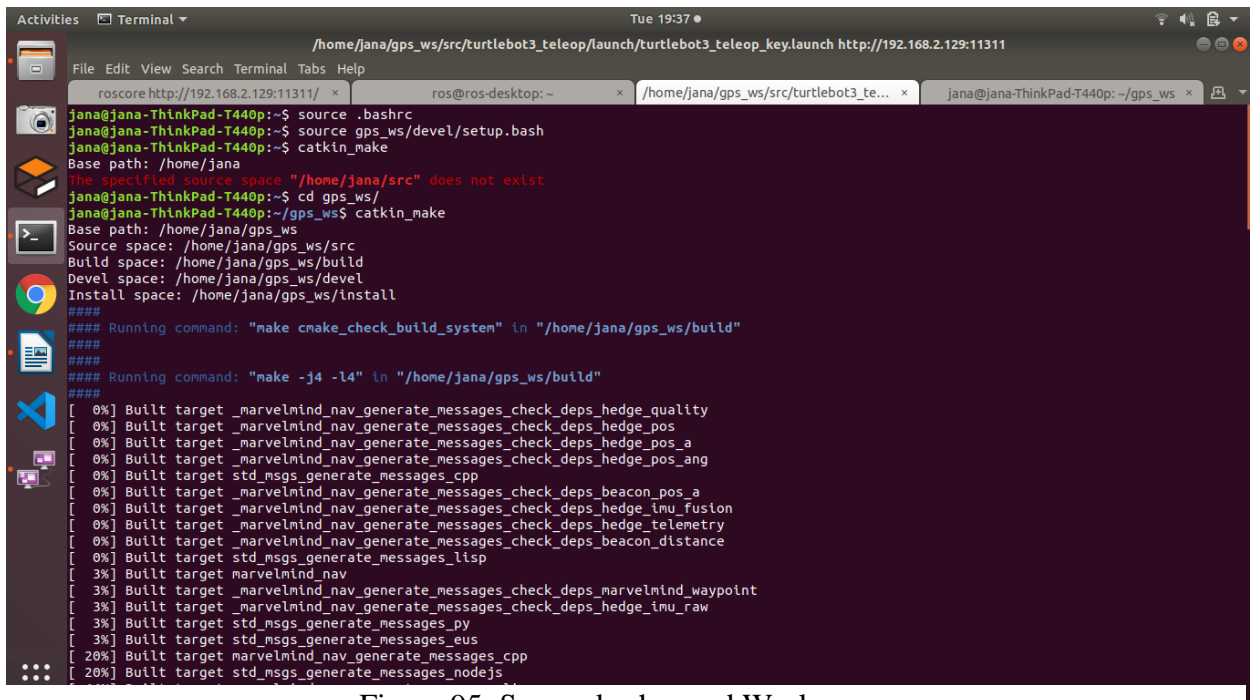


Figure 95: Source bashrc and Workspace

- Apply `sshros@ RaspberryIPAdress`, check the raspberry pi's port that the Arduino is connected to and run the `serial_node` python code to establish connection between the Raspberry pi and Arduino

```

ros@ros-desktop:~$ ls /dev/tty*
/dev/tty0  /dev/tty19  /dev/tty3  /dev/tty40  /dev/tty51  /dev/tty62
/dev/tty1  /dev/tty2  /dev/tty30  /dev/tty41  /dev/tty52  /dev/tty63
/dev/tty10 /dev/tty20  /dev/tty31  /dev/tty42  /dev/tty53  /dev/tty7
/dev/tty11 /dev/tty21  /dev/tty32  /dev/tty43  /dev/tty54  /dev/tty8
/dev/tty12 /dev/tty22  /dev/tty33  /dev/tty44  /dev/tty55  /dev/tty9
/dev/tty13 /dev/tty23  /dev/tty34  /dev/tty45  /dev/tty56  /dev/ttyAMA0
/dev/tty14 /dev/tty24  /dev/tty35  /dev/tty46  /dev/tty57  /dev/ttyprintk
/dev/tty15 /dev/tty25  /dev/tty36  /dev/tty47  /dev/tty58  /dev/ttyS0
/dev/tty16 /dev/tty26  /dev/tty37  /dev/tty48  /dev/tty59  /dev/ttyUSB0
/dev/tty17 /dev/tty27  /dev/tty38  /dev/tty49  /dev/tty6
/dev/tty18 /dev/tty28  /dev/tty39  /dev/tty5  /dev/tty60
/dev/tty19 /dev/tty29  /dev/tty4  /dev/tty50  /dev/tty61
ros@ros-desktop:~$ rosrunc rosserial_python serial_node.py /dev/ttyUSB0
[INFO] [1648571226.144296]: ROS Serial Python Node
[INFO] [1648571227.732150]: Connecting to /dev/ttyUSB0 at 57600 baud
[INFO] [1648571230.255121]: Requesting topics...
[INFO] [1648571231.129413]: Note: subscribe buffer size is 280 bytes
[INFO] [1648571231.206749]: Setup subscriber on cmd_vel [geometry_msgs/Twist]

```

Figure 96: Rosserial

- Then launch the `turtlebot3_teleop_key` launch file. In our case, we downloaded the turtlebot3's teleop file and used it on our robot

```

Activities Terminal Tue 19:37
/home/jana/gps_ws/src/turtlebot3_teleop/launch/turtlebot3_teleop_key.launch http://192.168.2.129:11311
File Edit View Search Terminal Tabs Help
roscore http://192.168.2.129:11311/ x ros@ros-desktop:~ x /home/jana/gps_ws/src/turtlebot3_te... x jana@jana-ThinkPad-T440p:~/gps_ws x
... shutting down processing monitor complete
done
jana@jana-ThinkPad-T440p:~/gps_ws$ roslaunch turtlebot3_teleop turtlebot3_teleop_key.launch
... Logging to /home/jana/.ros/log/03f05402-af7c-11ec-8f24-5c514fc93079/roslaunch-jana-ThinkPad-T440p-3902.log
Checking log directory for disk usage. This may take a while.
Press Ctrl-C to interrupt.
Done checking log file disk usage. Usage is <1GB.
started roslaunch server http://192.168.2.129:44185/

SUMMARY
=====
PARAMETERS
* /roscpp: melodic
* /rosversion: 1.14.12

NODES
/
  turtlebot3_teleop_keyboard (turtlebot3_teleop/turtlebot3_teleop_key)

ROS_MASTER_URI=http://192.168.2.129:11311

process[turtlebot3_teleop_keyboard-1]: started with pid [3917]

Control Your TurtleBot3!
-----
Moving around:
    w
    a s d
    x

w/x : increase/decrease linear velocity (Burger : ~ 0.22, Waffle and Waffle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Waffle Pi : ~ 1.82)

space key, s : force stop

```

Figure 97: Launch the turtlebot3\_teleop

- Finally run the `rqt_graph` to get the graph shown in

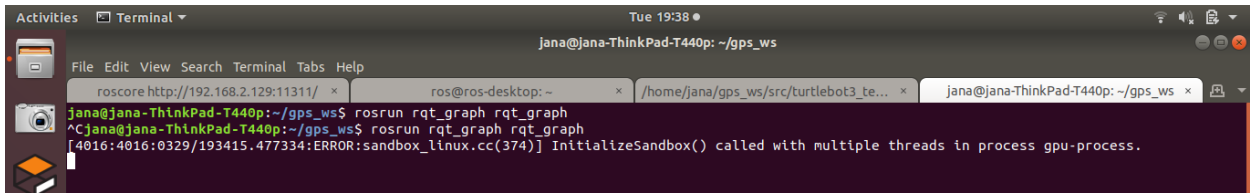


Figure 98: Run rqt\_graph

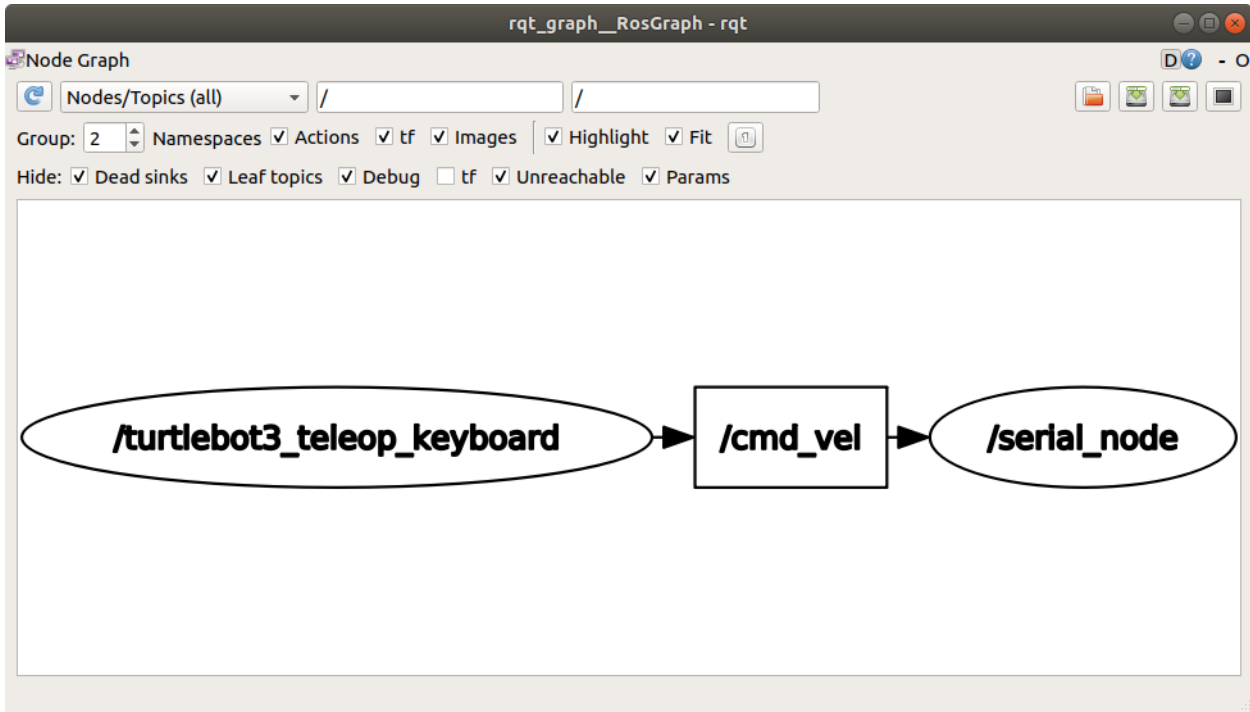


Figure 99: Node Graph

- The difference between the Turtlebot3 and our robot's teleop is mainly the speed because the motors of each robot are totally different. In our case, our robot's maximum speed is 0.5m/s that is mapped to 255 PWM in the Arduino code. However, in the turtlebot3's teleop file includes that every time the linear or angular speed command is pressed, the robot should accelerate by 0.5. When we used the same code, the robot started initially to move uncontrollably fast and whenever we increase its speed nothing change because it already saturated and reached its maximum speed. Therefore, we changed the 0.5 speed step to 0.05 in the teleop launch file. In this way the robot's speed will increase/decrease rationally.

## x. **Real Robot: Navigation through Rviz**

After making sure that the teleop is functioning properly regarding the speed and direction of the motors we shift to applying navigation on the robot by sending the Goal through Rviz. This step was applied previously on the Turtlebot3, but in our robot's case some changes took place in regards to the Raspberry pi and Arduino connections and the output in the rqt tree. Also, after several trails we recognized that the rotation extracted from the IMU of the Marvelmind GPS is not accurate at all, so we included an MPU-5060 to get the rotation of the robot. The code of the MPU-6050 is included in the general Arduino code of the system along with the code in the gps2odom package (in the Quaternions calculations).

The following steps were applied:

- Roscore in ubuntu
- Connect the Raspberry pi to ROS through "ssh" and establish a roserial connection for the connection between the Raspberry pi and Arduino.
- Run the hedge\_rcv\_bin from the marvelmind\_nav package that provide the robot's received location from the Marvelmind beacons
- Run the gps2odom python file from gps2odom package to get the odometry data from the previously received location.
- Then launch the spr\_navigation launch file, that includes launching the move\_base and the map. In other words, when we create our map, we include it's yaml file in this launch file for it to be processed by the map server and at the same time move\_base will be launched.
- Then we open rviz and add the saved configuration from the Turtlebot3 and three transformations will appear, the map, odom\_gps and base\_link\_gps (base\_link\_gps represents the robot on rviz)
- Run the rqt\_tree, and the difference here that we're not getting odom and base\_footprint transformations as we obtained in the Turtlebot3, that's because we don't have encoders in our robot and no odometry received from them.
- Run rqt\_graph to show the nodes.



## APPENDIX B

### DETECTION TUTORIALS

#### i. Installing LabelImg

We install it using the steps written in the documentation found below:

<https://github.com/tzutalin/labelImg>

We used anaconda on windows to install the program.

First, we install anaconda from <https://docs.anaconda.com/anaconda/install/windows/>.

Then, we download the labelImg zip file, as shown in figure 100, and unzip the folder.

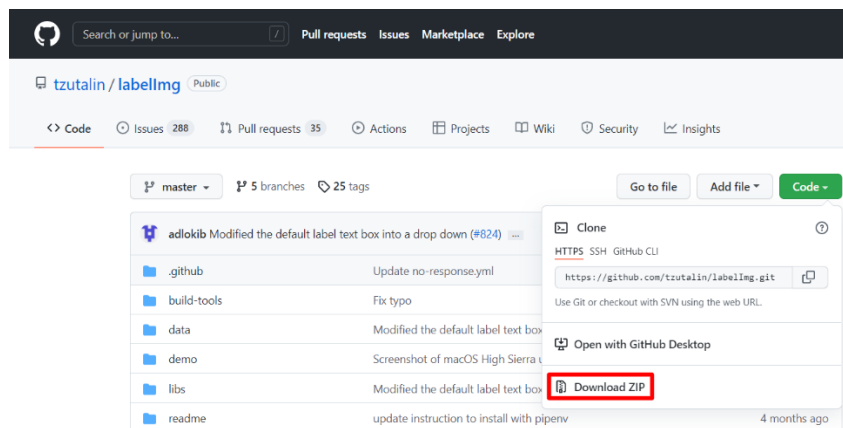


Figure 100: LabelImg Download Zip File

Now we open the Anaconda Prompt and create a virtual environment (Note: we have named our virtual environment, labelImg\_tutorial). “In a nutshell, Python virtual environments help decouple and isolate versions of Python and associated pip packages. This allows end-users to install and manage their own set of packages that are independent of those provided by the system.”

Virtual environment is created using “conda create -n  $[\text{name\_of\_your\_env}]$  python= $[\text{python\_version}]$ ”, as illustrated in figure 101.

```

Anaconda Prompt (anaconda3)

(base) C:\Users\nabil>conda create -n labelingg_tutorial python=3
Collecting package metadata (current_repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\nabil\anaconda3\envs\labelimg_tutorial

  added / updated specs:
    - python=3

The following NEW packages will be INSTALLED:

bzip2                pkgs/main/win-64::bzip2-1.0.8-he774522_0
ca-certificates      pkgs/main/win-64::ca-certificates-2021.10.26-haa95532_4
certifi              pkgs/main/noarch::certifi-2020.6.20-pyhd3eb1b0_3
libffi               pkgs/main/win-64::libffi-3.4.2-h604cdb4_1
openssl              pkgs/main/win-64::openssl-1.1.1m-h2bbff1b_0
pip                  pkgs/main/win-64::pip-21.2.4-py310haa95532_0
python               pkgs/main/win-64::python-3.10.0-h96c0403_3
setuptools           pkgs/main/win-64::setuptools-58.0.4-py310haa95532_0
sqlite               pkgs/main/win-64::sqlite-3.37.0-h2bbff1b_0
tk                   pkgs/main/win-64::tk-8.6.11-h2bbff1b_0
tzdata               pkgs/main/noarch::tzdata-2021e-hda174b7_0
vc                   pkgs/main/win-64::vc-14.2-h21ff451_1
vs2015_runtime       pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
wheel                pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0
wincertstore         pkgs/main/win-64::wincertstore-0.2-py310haa95532_2

```

Figure 101: Creating Virtual Environment

Then activate it using “conda activate \$[name of your env]”.

```

Anaconda Prompt (anaconda3)

setuptools           pkgs/main/win-64::setuptools-58.0.4-py310haa95532_0
sqlite               pkgs/main/win-64::sqlite-3.37.0-h2bbff1b_0
tk                   pkgs/main/win-64::tk-8.6.11-h2bbff1b_0
tzdata               pkgs/main/noarch::tzdata-2021e-hda174b7_0
vc                   pkgs/main/win-64::vc-14.2-h21ff451_1
vs2015_runtime       pkgs/main/win-64::vs2015_runtime-14.27.29016-h5e58377_2
wheel                pkgs/main/noarch::wheel-0.37.1-pyhd3eb1b0_0
wincertstore         pkgs/main/win-64::wincertstore-0.2-py310haa95532_2
xz                   pkgs/main/win-64::xz-5.2.5-h62dcd97_0
zlib                 pkgs/main/win-64::zlib-1.2.11-h8cc25b3_4

Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#   $ conda activate labelingg_tutorial
#
# To deactivate an active environment, use
#
#   $ conda deactivate

(base) C:\Users\nabil>conda activate labelingg_tutorial
(labelimg_tutorial) C:\Users\nabil>

```

Figure 102: Virtual Environment Activation

We then go to the directory where we uncompressed the labelingg folder:

```
Anaconda Prompt (anaconda3)

Proceed ([y]/n)? y

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#   $ conda activate labelimg_tutorial
#
# To deactivate an active environment, use
#
#   $ conda deactivate

(base) C:\Users\nabil>conda activate labelimg_tutorial

(labelimg_tutorial) C:\Users\nabil>C:\Users\nabil\OneDrive\Desktop\labelImg-master
'C:\Users\nabil\OneDrive\Desktop\labelImg-master' is not recognized as an internal or external command,
operable program or batch file.

(labelimg_tutorial) C:\Users\nabil>C:\Users\nabil\OneDrive\Desktop
'C:\Users\nabil\OneDrive\Desktop' is not recognized as an internal or external command,
operable program or batch file.

(labelimg_tutorial) C:\Users\nabil>cd C:\Users\nabil\OneDrive\Desktop\labelImg-master

(labelimg_tutorial) C:\Users\nabil\OneDrive\Desktop\labelImg-master>
```

Figure 103: Navigating Using Ubuntu Terminal

We proceed by installing the below files:

## Windows + Anaconda

Download and install [Anaconda](#) (Python 3+)

Open the Anaconda Prompt and go to the [labelImg](#) directory

```
conda install pyqt=5
conda install -c anaconda lxml
pyrcc5 -o libs/resources.py resources.qrc
python labelImg.py
```

Figure 104: Installation Steps

When typing “python [labelImg.py](#)” in the terminal the labelImg program will appear:

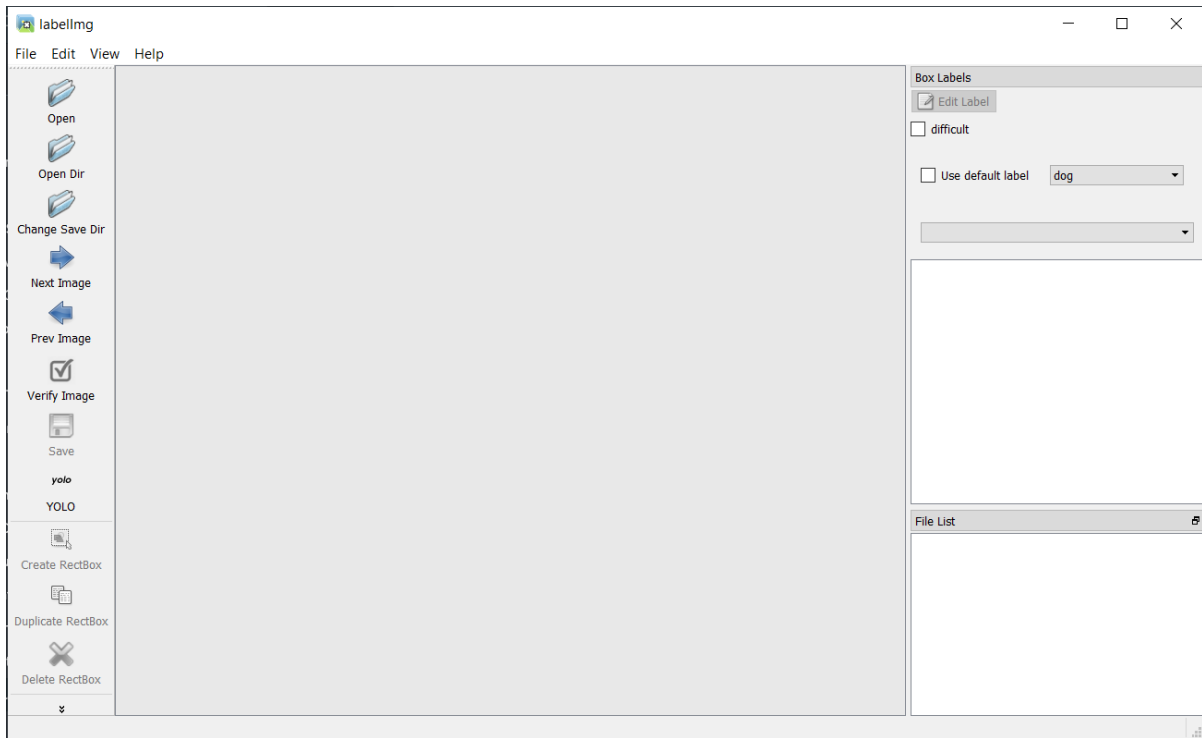


Figure 105: LabelImg Program Interface

First, we select the save directory by choosing a folder where to save directory. Then we open main directory to choose the images we want to annotate.

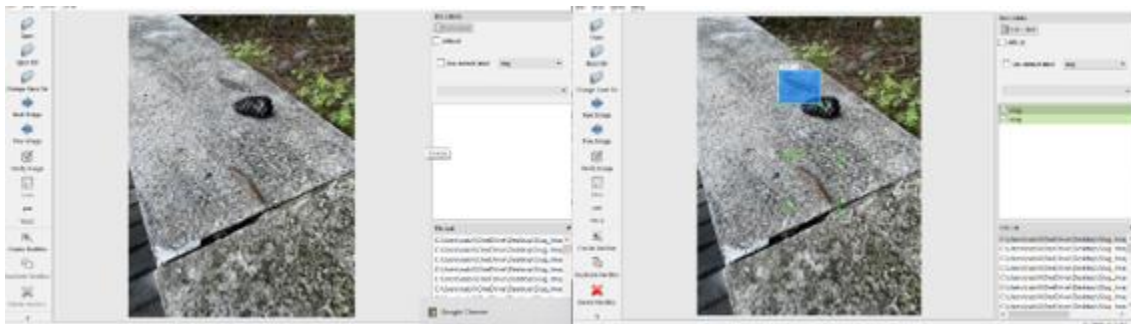


Figure 106: LabelImg Annotation Process

Saving the output in the save directory. After finishing annotating, we ended up with 2 types of files

Normally, .txt files are generated and a “classes.txt” file appears. This txt file indicates the classes labels that Yolov5 could detect them. The numbers of these labels start from [0].

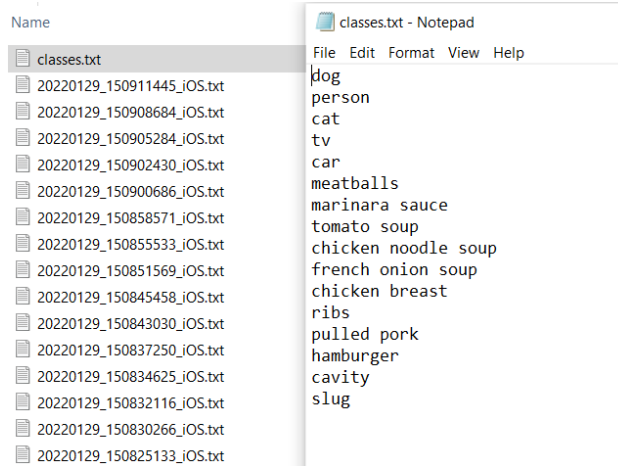


Figure 107: Resulted Files From Annotation Process

The txt files generated after annotating the image contains the slug annotation in the image and image coordinates.

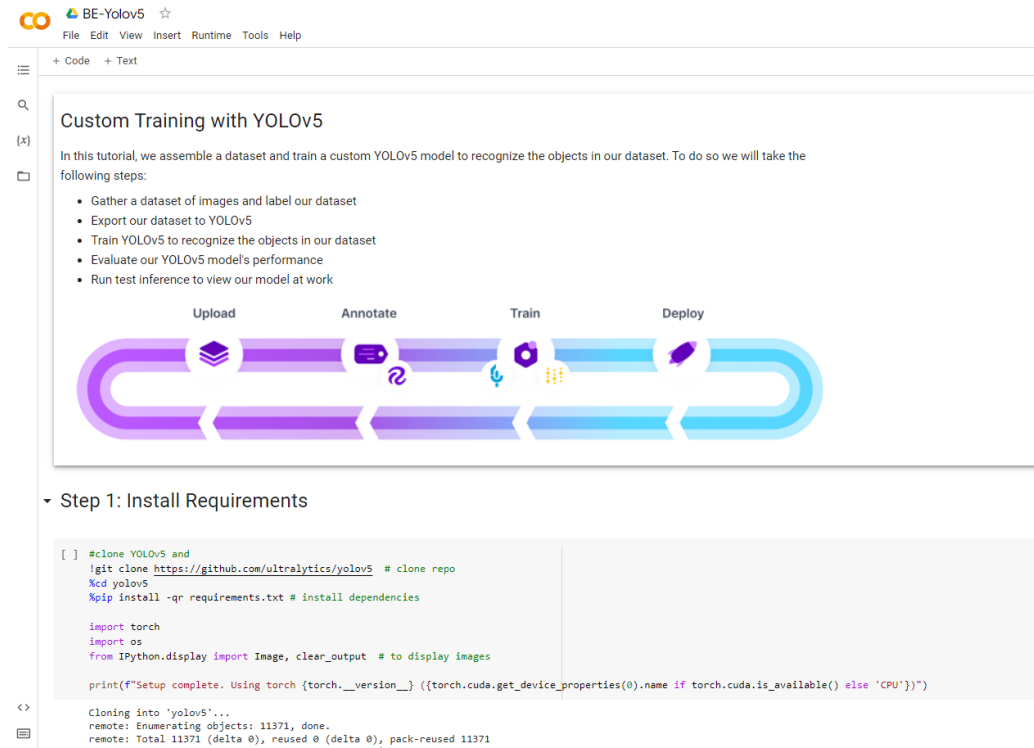
File Name	Modified	Type	Size
classes.txt	31-Jan-22 8:50 PM	Text Document	1 KB
20220129_150911445_iOS.txt	31-Jan-22 8:44 PM	Text Document	1 KB
20220129_150908684_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150905284_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150902430_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150900686_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150858571_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150855533_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150851569_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150845458_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150843030_iOS.txt	31-Jan-22 8:43 PM	Text Document	1 KB
20220129_150837250_iOS.txt	31-Jan-22 8:42 PM	Text Document	1 KB
20220129_150834625_iOS.txt	31-Jan-22 8:42 PM	Text Document	1 KB
20220129_150832116_iOS.txt	31-Jan-22 8:42 PM	Text Document	1 KB
20220129_150830266_iOS.txt	31-Jan-22 8:42 PM	Text Document	1 KB

Figure 108: Bounding Box Coordinates of a Slug and Class Number

Finally, the folder contains both the images and the labels.

## ii. Training Model using Custom Dataset on Google Colab

Training is done using google Colab, where it provides virtual environment with the needed imports and clear coding cells.



BE-Yolov5

File Edit View Insert Runtime Tools Help

+ Code + Text

### Custom Training with YOLOv5

In this tutorial, we assemble a dataset and train a custom YOLOv5 model to recognize the objects in our dataset. To do so we will take the following steps:

- Gather a dataset of images and label our dataset
- Export our dataset to YOLOv5
- Train YOLOv5 to recognize the objects in our dataset
- Evaluate our YOLOv5 model's performance
- Run test inference to view our model at work

Upload Annotate Train Deploy

Step 1: Install Requirements

```
[ ] #clone YOLOv5 and
!git clone https://github.com/ultralytics/yolov5 # clone repo
%cd yolov5
%pip install -qr requirements.txt # install dependencies

import torch
import os
from IPython.display import Image, clear_output # to display images

print(f"Setup complete. Using torch {torch.__version__} ({torch.cuda.get_device_properties(0).name if torch.cuda.is_available() else 'CPU'})")
```

<>

Cloning into 'yolov5'...

remote: Enumerating objects: 11371, done.

remote: Total 11371 (delta 0), reused 0 (delta 0), pack-reused 11371

Figure 109: YOLOV5 in Google Colab

```
#clone YOLOv5 and
!git clone https://github.com/ultralytics/yolov5 # clone repo
%cd yolov5
%pip install -qr requirements.txt # install dependencies

import torch
import os
from IPython.display import Image, clear_output # to display images

print(f"Setup complete. Using torch {torch.__version__} ({torch.cuda.get_device_properties(0).name if torch.cuda.is_available() else 'CPU'})")
```

Figure 110: Cloning the YOLOV5 Repository and Importing Necessary Libraries

We also import images from google drive by mounting the drive and unrar the winrar folder containing the images.

Here, we are able to pass a number of arguments:

- **img:** define input image size
- **batch:** determine batch size
- **epochs:** define the number of training epochs. (Note: often, 3000+ are common here!)
- **data:** Our dataset location is saved in the `dataset.location`
- **weights:** specify a path to weights to start transfer learning from. Here we choose the generic COCO pretrained checkpoint.
- **cache:** cache images for faster training

```
[ ] |python train.py --img 416 --batch 32 --epochs 130 --data custom_dataset.yaml --weights yolov5s.pt --cache
```

Figure 111: Train.py File with Input Arguments

We trained the model using default image size 416 which could be changed. Important note is that the images that the model will apply detection on must be same size with the image size training was done on. For example, in our case, we trained the model on 416 image size. Thus, images that we want to apply inference on, must be converted to 416 before entering the model.

Our custom trained model at the end gets exported in “.pt” format and is ready to use.

After training our model and being satisfied with training results, we go for inference or deploying the model

Note: after training, we can use Tensorboard to visualize the evaluation metrics graphs from which we could extract some important data.

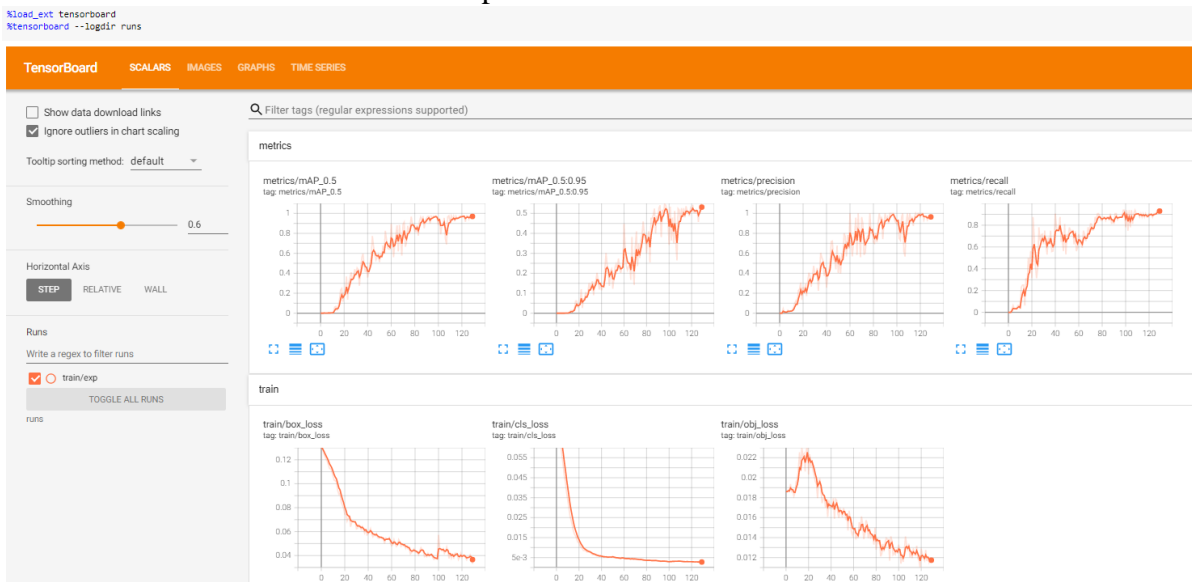


Figure 112: Tensorboard Graphs

### iii. Inference

Inference is the process of applying the trained model on images or videos. To do so, we can use the “detect.py” file that is included in the Yolov5 repository.

Below are some of the arguments that should be passed. First, we specify the weights file or the output of the training process (train.py file). Second, we need to specify the source of the input, for example source 0 is for the webcam and if we need to inference a photo, we put it in our directory and specify its location.

```
Usage - sources:
$ python path/to/detect.py --weights yolov5s.pt --source 0          # webcam
                                                                    img.jpg          # image
                                                                    vid.mp4         # video
                                                                    path/          # directory
                                                                    path/*.jpg     # glob
                                                                    'https://youtu.be/Zgi9g1ksQhc' # YouTube
                                                                    'rtsp://example.com/media.mp4' # RTSP, RTMP, HTTP stream

Usage - formats:
$ python path/to/detect.py --weights yolov5s.pt                  # PyTorch
                                                                    yolov5s.torchscript  # TorchScript
                                                                    yolov5s.onnx         # ONNX Runtime or OpenCV DNN with --dnn
                                                                    yolov5s.xml         # OpenVINO
                                                                    yolov5s.engine      # TensorRT
                                                                    yolov5s.mlmodel     # CoreML (macOS-only)
                                                                    yolov5s_saved_model # TensorFlow SavedModel
                                                                    yolov5s.pb          # TensorFlow GraphDef
                                                                    yolov5s.tflite      # TensorFlow Lite
                                                                    yolov5s_edgetpu.tflite # TensorFlow Edge TPU

"""
```

Figure 113: Detect.py Arguments for Weights and Image Source

Below are the arguments we used. Best.pt is our weights file of the model after training, our image size we used to train, confidence metric (show only objects with a prediction confidence higher than the specified percentage) and in the source we specified the video path in our directory to apply the detection on.

```
!python detect.py --weights best.pt --img 416 --conf 0.5 --source '/content/videos/videos'
```

Figure 114: Detect.py Arguments Passed in the Code Cells

In the Colab part, we finally tried to get the coordinates of the boundary boxes. This was done by the below code:





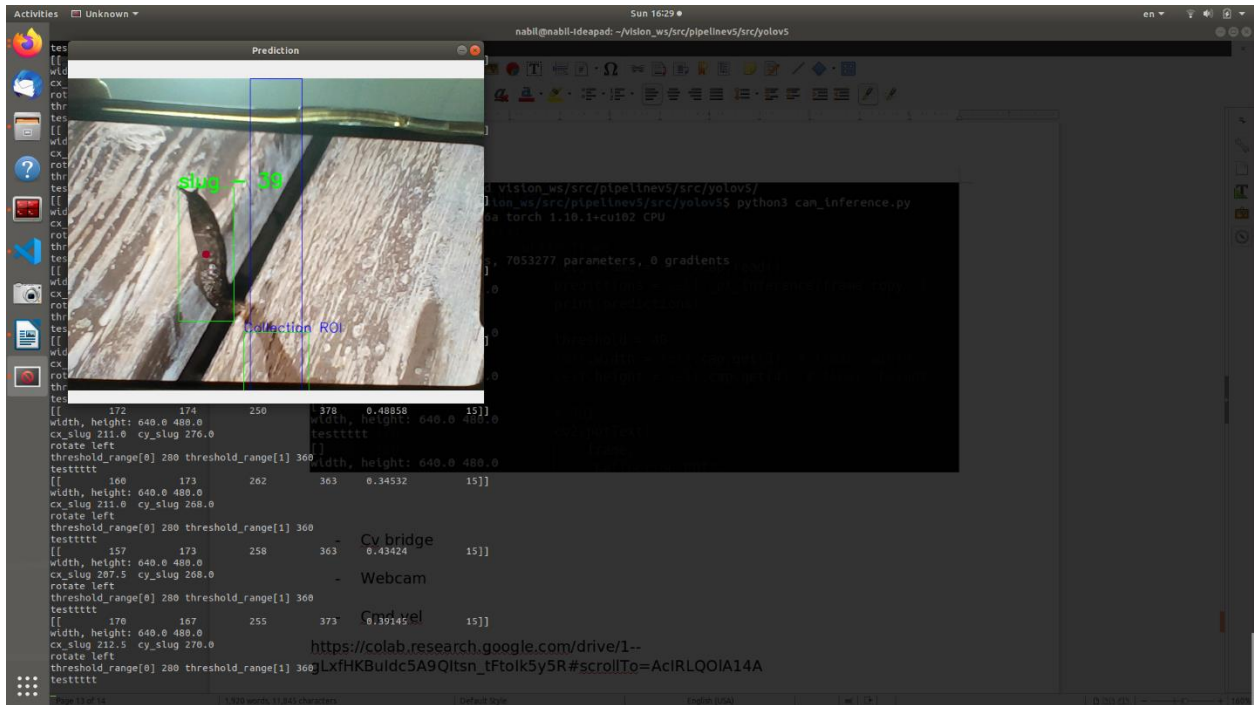


Figure 117: Inference Results

We can see that in the terminal, we printed the prediction output from the model which is a 2D array that includes the coordinates, confidence, and the class number.



Figure 118: Detection Result

#### v. Sending Commands to the Robot

After detecting a slug, the robot centers itself first with the centroid of the located slug by turning right or left. We drew a virtual region of interest at the center of the frame, where the robot stays on turning until this slug is inside this box, indicating its alignment with the robot. Finally, the robot stays on moving forward and stops only when the slug becomes

inside the boundary of another virtual ROI implying the robot is ready to start the collection process. The process is demonstrated later in the report.

We can know that the slug is in certain region by calculating the slug's bounding box centroid and comparing it with the x and y coordinates of the ROI. Moreover, the code will initialize a publisher node that sends a Twist msg to /cmd\_vel topic to move the robot as convenient.

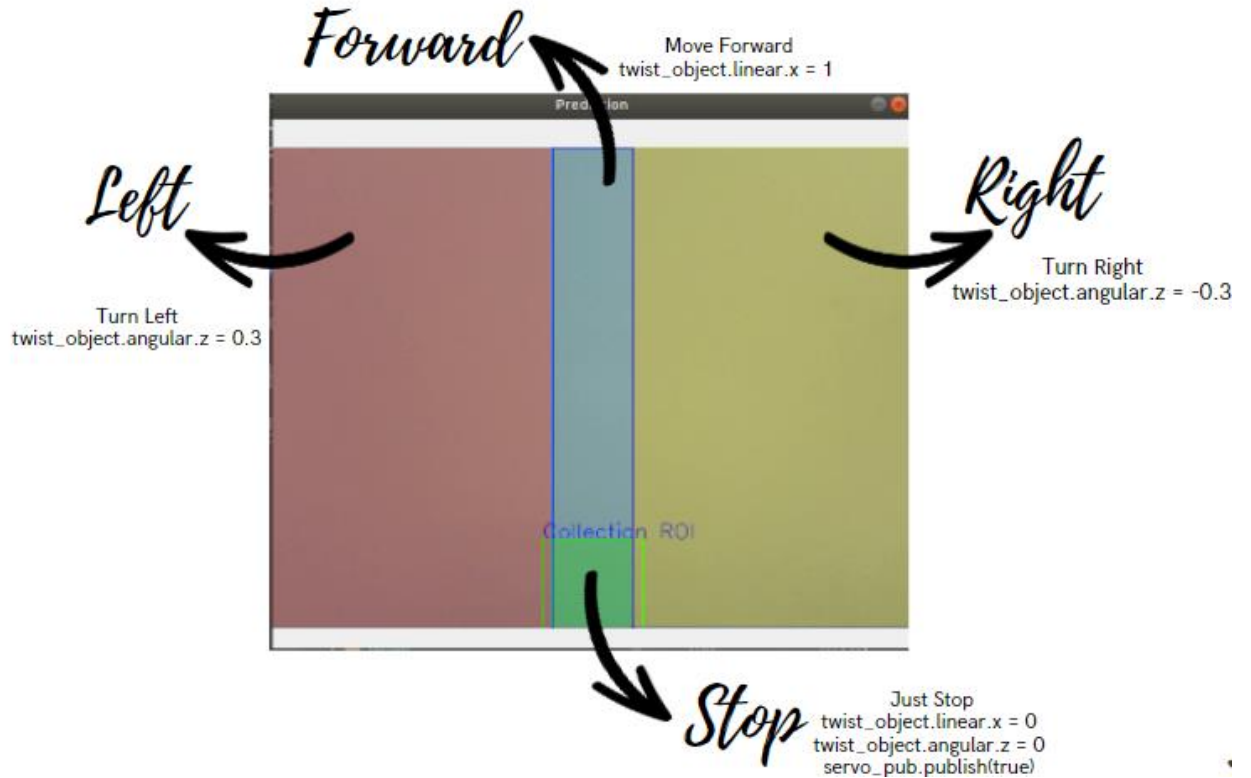


Figure 119: Region of Interest

Region 1 (Left Region):

If the slug is in the left region, then the robot will turn left. The node will send a Twist message that will move the robot in an angular motion around z. (twist.angular.z)

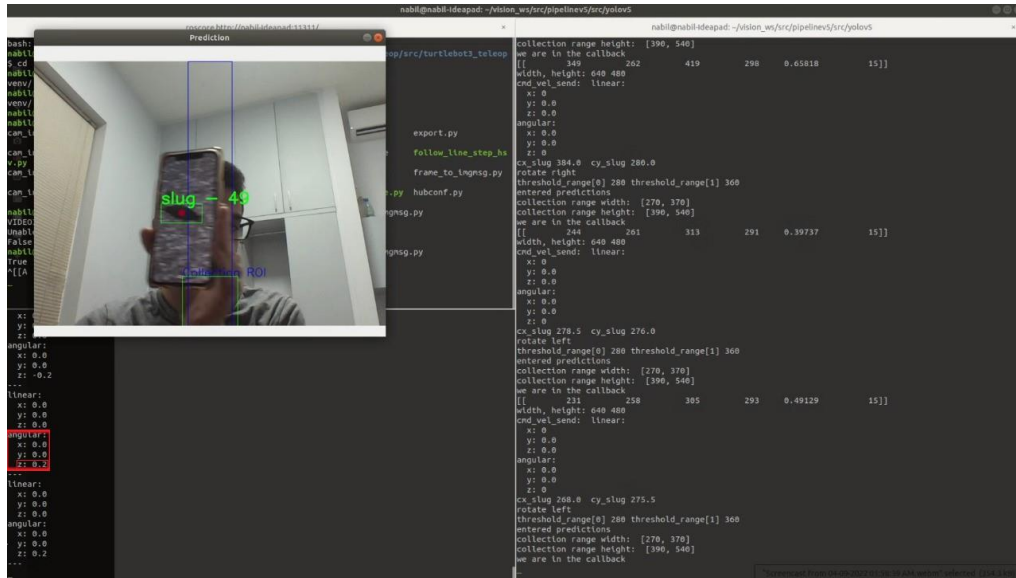


Figure 120: Slug Centroid is in the Left ROI and cmd\_vel Value Turns the Robot to the Left

Region 2 (Right Region):

If the slug is in the right region, then the robot will turn right. The node will similarly send a Twist message that will move the robot in an angular motion around z but the value will be opposite to the left region in sign. (twist.angular.z)

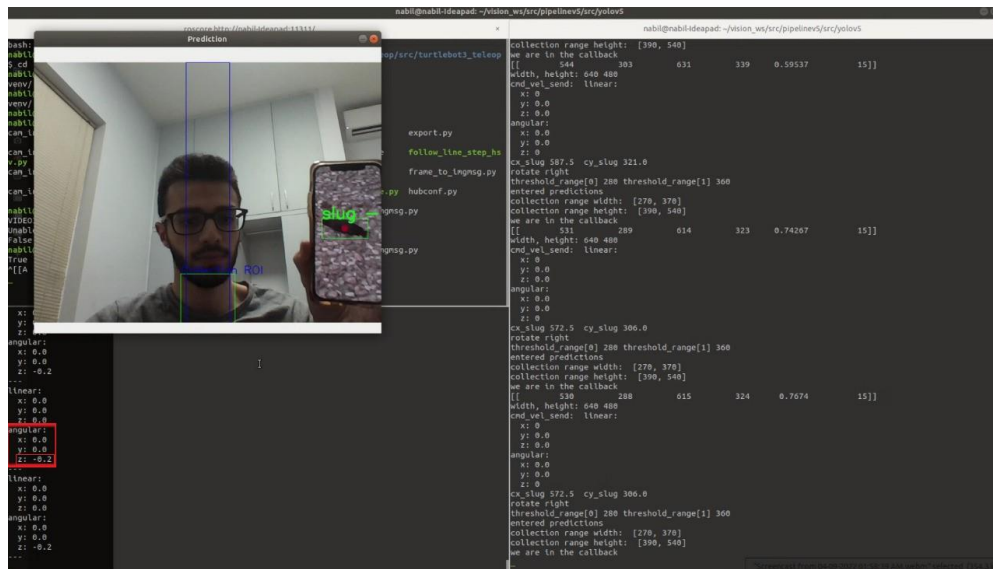


Figure 121: Slug Centroid is in the Right ROI and cmd\_vel Value turns the robot to the right

Region 3 (Forward Region):

If the slug is in the forward region, then the robot will turn forward. The node will send a Twist message that will move the robot in a straight motion along x direction. (twist.linear.x)

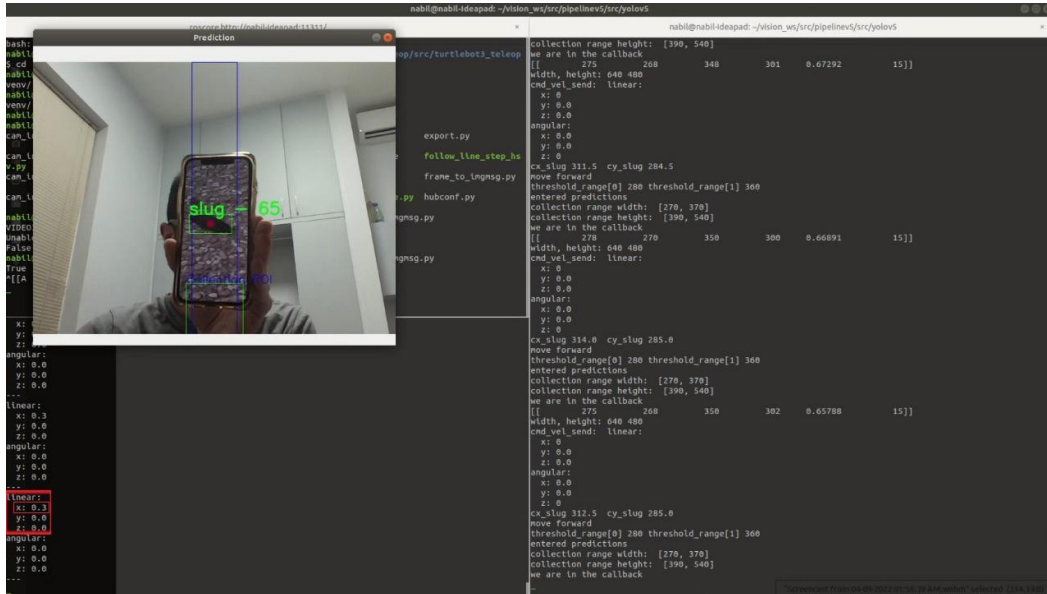


Figure 122: Slug Centroid is in the Forward ROI and cmd\_vel Value Moves Forward the Robot

Region 4 (Collection Region):

If the slug is in the collection region, then the robot will stop, and the collection process will start. The node will send a Twist message that will stop the robot by sending 0 value to the linear and angular values of Twist.

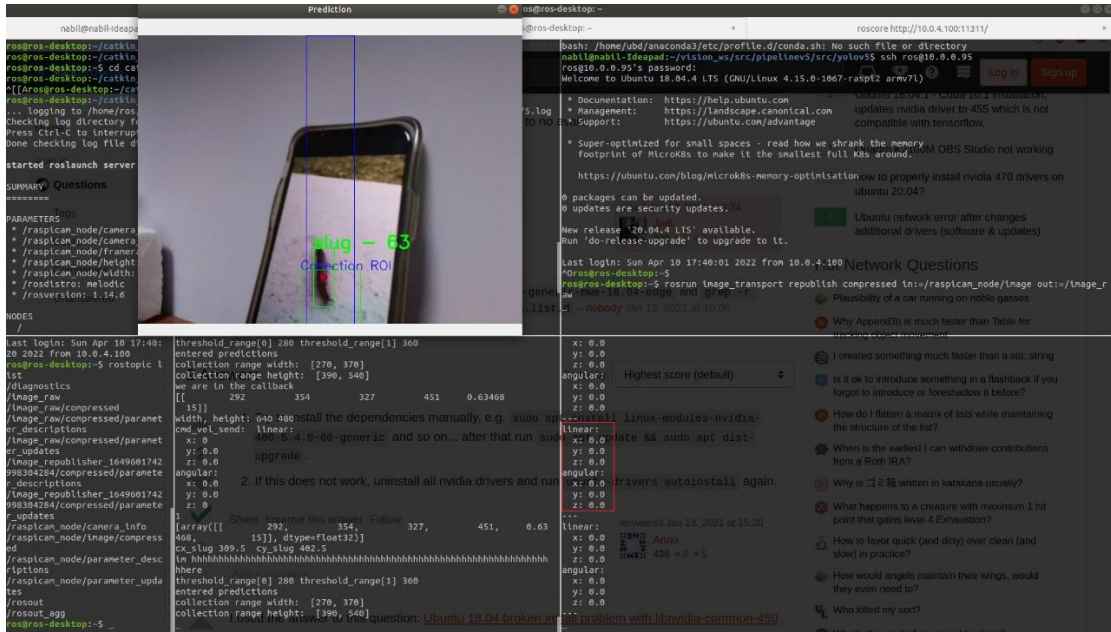


Figure 123: Slug Centroid is in the Collection ROI and cmd\_vel Values Stops the Robot and Starts Collection Process

Note: the file “cam\_inference\_ROS.py” does send Twist message, it just prints to the terminal the directions. The final file “cam\_inference\_ROS\_no.py” does that.

Using this method, the robot will start centering itself with the slug and will move towards it. When the slug falls into the collection ROI, the collection force will begin to collect the slug.

Note: notice that when starting any python file that does inference, we write “python3” before the file name and not “python” because Yolov5 requires a version of python > 3.

## vi. OpenCV and ROS

In the previous example we have the webcam of the laptop directly which is not the case in our robot. We will be using a rpi camera that will send the webcam images from the raspberry pi to the laptop where the inference is done.

So, we have written a simple node that will read from the webcam using OpenCV and send its frames as a ROS message called Image. The file “frame\_to\_imgmsg.py” converts the OpenCV type frame to Image msg and publish it to /image\_raw topic.

Also, we have modified the main code where we added a subscriber that subs to the “/image\_raw” topic and converts it back to OpenCV format to be able to do inference on it. The modified file is named “cam\_inference\_ROS.py”

Note: usually the interface between OpenCV and ROS messages happens using CVBridge library as shown in figure 124.

ROS doesn't support python3 by default and compile the CVBridge library on python 2, default version. Although there are several online solutions for this problem, but nothing has worked. At the end we took the important functions we need from this library and wrote them in our python file which solved the problem.

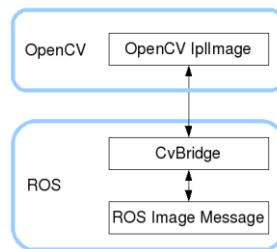


Figure 124: OpenCV-ROS Interface

We used 2 functions. The first one was to convert image read by OpenCV to ROS message called Image. This function is named “imgmsg\_to\_cv2”. The second function does the reverse process of converting this ROS message to OpenCV image type, which is called “cv2\_to\_imgmsg”.

#### vii. Sending Webcam Feed to the Laptop using ROS

In the above example we have got the frames of the webcam of our laptop. In this example we added the raspberry pi camera and initialized it. Webcam feed is processed by deploying the model on it frame by frame. Sending these images from the raspberry pi to the laptop can be done using various methods. One of them is to create a server and send them there. Another approach, which we choose, is to use ROS to send them. Raspiancam\_node is a ROS node that publishes at one time the current frame of the webcam feed. It helps in converting OpenCV type image into ROS message “Image” and vice-versa. On the laptop side, we created a node that subscribes to this node sending the image from the webcam, passes the frame through the model, and returns the output annotated image and with the coordinates of the slug.

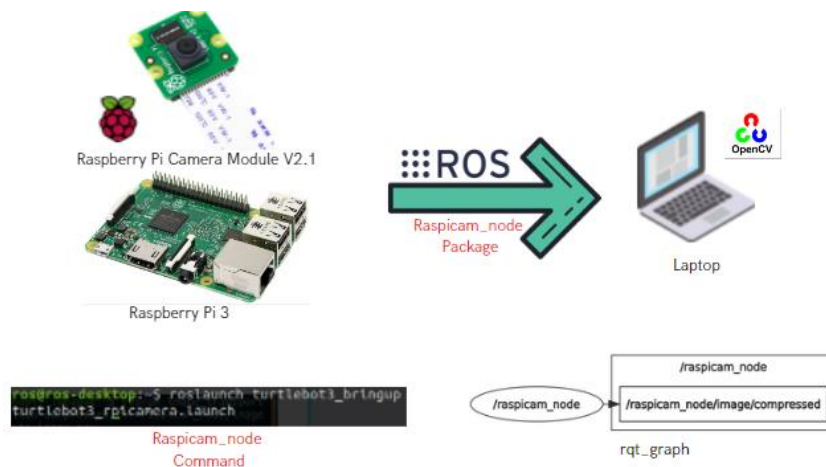


Figure 125: Sending Webcam Feed from rpicamera v2 to the Laptop

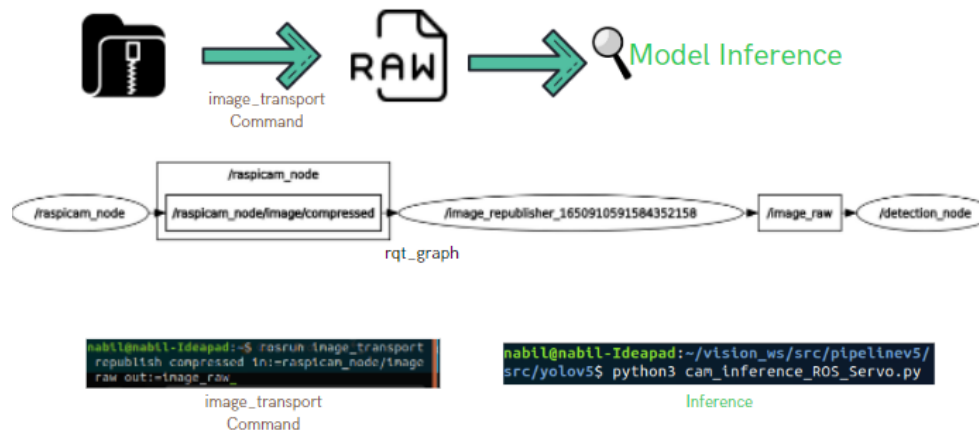


Figure 126: Image Uncompressing and Model Inference Start

First, we connect to the robot using SSH Networking protocol. Then we “cd catkin\_ws” and source it. The next step was to bring up the rpi camera by entering this command to the terminal “roslaunch turtlebot3\_bringup turtlebot3\_rpicamera.launch”.

Note: don’t forget to export the turtlebot3 model before using the above command.





We used the turtlebot3 to detect a slug image on the phone. The turtlebot has maneuvered towards the phone until the slug centroid reach the collection ROI.



Figure 129: Test #1 on Turtlebot

Test #2:

In the second test we have tested the code our robot with the collection mechanism. When the inference code detects the slug in the collection ROI, it sends a Boolean msg to a topic named /Servo. At the other side, in the Arduino, we have written a subscriber that subscribes to this /Servo topic and if it returns true, then the robot will stop and the collection mechanism will open and the roller motor will turn on.

The robot successfully detected and collected the slug while using the “cam\_inference\_ROS\_Servo.py” code

Test #3:

After the success of integrating the vision with the collection mechanism, the next step is to apply this integration while the robot is navigating autonomously.

It was done in a classroom of 12 m<sup>2</sup> area, and it was a successful.

Final Colab: [https://colab.research.google.com/drive/1--](https://colab.research.google.com/drive/1--gLxfHKBUIdc5A9QItsn_tFtoIk5y5R?authuser=1#scrollTo=6ksMD5KvkjZp)

[gLxfHKBUIdc5A9QItsn\\_tFtoIk5y5R?authuser=1#scrollTo=6ksMD5KvkjZp](https://colab.research.google.com/drive/1--gLxfHKBUIdc5A9QItsn_tFtoIk5y5R?authuser=1#scrollTo=6ksMD5KvkjZp)

## APPENDIX C

### SURVEY

Considering the fact that there are not local customers (in Lebanon) and the product's target-market is in Europe, a survey was conducted targeting European citizens. The survey helped in summarizing the needs of the customers along with the requirements of the design.

The following are the questions of the survey along with the resulting data:

- Do you grow plants in your backyard?
  - 88.9% Yes
  - 11.9% No
- If yes, what type of plants do you grow?
  - 37.5% Salads (cabbage, tomatoes, pumpkins...etc.)
  - 25% Strawberries
  - 12.5% Flowers in general
  - 12.5% Raspberry, Apricot
  - 12.5% Herbs, Gourds
- How are the plants distributed in your backyard?
  - 55.6% In boxes along the fence
  - 11.1% In beds along the fence
  - 22.2% Random
  - 11.1% Uniform
- What pests are your crops suffering from?
  - 66.7% Slugs
  - 22.2% Snails
  - 11.1% Gall Mites, Mildew, Squirrels, Fleas, Caterpillars, Aphids
- If you have Slugs problem, how would you describe the number of slugs in your field?
  - 44.4% 2 per meter square
  - 11.1% 5 per meter square
  - 11.1% 8 per meter square
  - 11.1% Less than 2 per meter square
  - 11.1% 1 or 2 in the whole garden

- 11.1% None
- What type of problems are the slugs causing?
  - Eating the plants, sometimes completely, especially if they are young
  - Entering home
  - Crop damage, trail marks
  - Minimal. Small garden in the city, they're not with many and don't bother me very much.
  - They eat our plants
  - Eating vegetables
- How are you addressing the problems caused by the slugs?
  - Slug pellets, manual collection
  - Slug removal powder
  - Nothing in particular
  - I move them to somewhere else.
  - We bought a special product
- How effective are the solution you are implementing to fight the slugs?
  - 28.6% Reduces the damage, but I still have slugs
  - 28.6% Not very good, I still need solutions
  - 28.6% Major improvement
  - 14.3% Slightly better
- Are those solution time-effective?
  - 33.3% Yes
  - 66.7% No
- How costly are those solutions in € per month?
  - 33.3% 0€
  - 11.1% 5€
  - 11.1% 10€
  - 11.1% 20€
  - 11.1% Less than 10€
  - 22.3% NA
- On average what is the area you plant in meters square?

- 55.6% 5-meters square
- 22.2% 2-meters square
- 22.2% 20-meters square
- Is your land horizontal or inclined? if inclined please specify the slope in degree. (You can use your phone to determine the angle)
  - 87.5% low sloping area - 0° to 10°
  - 12.5% Moderate sloping area -10° to 15°
- Would you buy a pest control device?
  - 55.6% Yes
  - 44.4% No
- If yes, would you buy the device even if it harms the pests?
  - 44.4 % Yes
  - 55.6% No

## APPENDIX D

### CE CERTIFICATE

By affixing the CE marking to a product, a manufacturer declares that the product meets all the legal requirements for CE marking and can be sold throughout the European Economic Area (EEA). This also applies to products made in other countries that are sold in the EEA.

Manufacturers play a crucial role in ensuring that products placed on the extended single market of the European Economic Area are safe. They are responsible for checking that their products meet EU safety, health, and environmental protection requirements. It is the manufacturer's responsibility to carry out the conformity assessment, set up the technical file, issue the EU declaration of conformity, and affix the CE marking to a product. Only then can this product be traded on the EEA market

If you are a manufacturer, you have to follow these 6 steps to affix a CE marking to your product:

1. Identify the applicable directive(s) and harmonized standards
2. Verify product specific requirements
3. Identify whether an independent conformity assessment (by a notified body) is necessary
4. Test the product and check its conformity
5. Draw up and keep available the required technical documentation
6. Affix the CE marking and draw up the EU Declaration of Conformity

# APPENDIX E

## STANDARDS

### **Ingress Protection (IP) Rating (IEC Standard 60529)**

The IP system is an internationally recognized method to indicate the degree of protection against the ingress of dust, solid objects and moisture into an enclosure. The letters "IP" are followed by two numerals.

- **First Numeral**

Protection of persons against contact with or approach to live parts and against contact with moving parts, other than smooth rotating shafts and the like, inside the enclosure and protection of the equipment against ingress of solid foreign bodies in accordance with IEC 60598-1:2003

- 0 Not protected
- 1 Protected against solid objects 50 mm in diameter or greater. (A large surface of the body, such as a hand and no protection against deliberate access).
- 2 Protected against solid objects 12 mm in diameter or greater. (Fingers or similar objects not exceeding 80mm in length).
- 3 Protected against solid objects 2.5 mm in diameter or greater. (Tools, wires, etc., of diameter or thickness greater than 2.5 mm).
- 4 Protected against solid objects 1mm in diameter or greater. (Wires or other similar solid material of thickness greater than 1mm in diameter).
- 5 Dust protected. (Dust does not enter in sufficient quantity to interfere with satisfactory operation of equipment).
- 6 Dust tight.

- **Second Numeral**

The second numeral indicates the degree of protection against the ingress moisture as defined in IEC 60598-1:2003.

- 0 Not protected
- 1 Protected against dripping water. (Dripping water (vertically falling drops) shall have no harmful effect).

- 2 Protected against dripping water when tilted up to 15°. (Vertically dripping water shall have no harmful effect when the enclosure is tilted at an angle up to 15° from its normal position).
- 3 Protected against spraying water. (Water falling as a spray at any angle up to 60° from the vertical shall have no harmful effect).
- 4 Protected against splashing water. (Water splashing against the enclosure from any direction shall have no harmful effect).
- 5 Protected against water jets. (Water projected by a nozzle against enclosure from any direction shall have no harmful effects).
- 6 Protected against heavy seas. (Water from heavy seas or projected in powerful water jets shall not enter the enclosure in harmful quantities).
- 7 Protected against the effects of temporary immersion. (Ingress of water in harmful quantity shall not be possible when the enclosure is immersed in water under defined conditions of pressure and time).
- 8 Protected against continuous immersion. (The equipment is suitable for continuous submersion in water under conditions which shall be specified by the manufacturer).



## **Impact Protection (IK) Rating**

Degrees of protection provided by enclosures for electrical equipment against external mechanical impacts in accordance with IEC 62262:2002 and IEC 60068-2-75:1997.

- IK00 Not protected
- IK01 Protected against 0.14 joules impact. (Equivalent to impact of 0.25 kg mass dropped from 56 mm above impacted surface).
- IK02 Protected against 0.2 joules impact. (Equivalent to impact of 0.25 kg mass dropped from 80 mm above impacted surface).
- IK03 Protected against 0.35 joules impact. (Equivalent to impact of 0.25 kg mass dropped from 140 mm above impacted surface).
- IK04 Protected against 0.5 joules impact. (Equivalent to impact of 0.25 kg mass dropped from 200 mm above impacted surface).
- IK05 Protected against 0.7 joules impact. (Equivalent to impact of 0.25 kg mass dropped from 280 mm above impacted surface).
- IK06 Protected against 1 joules impact. (Equivalent to impact of 0.25 kg mass dropped from 400 mm above impacted surface).
- IK07 Protected against 2 joules impact. (Equivalent to impact of 0.5 kg mass dropped from 400 mm above impacted surface).
- IK08 Protected against 5 joules impact. (Equivalent to impact of 1.7 kg mass dropped from 300 mm above impacted surface).
- IK09 Protected against 10 joules impact. (Equivalent to impact of 5 kg mass dropped from 200 mm above impacted surface).
- IK10 Protected against 20 joules impact. (Equivalent to impact of 5 kg mass dropped from 400 mm above impacted surface).

**ABET KPIs**

	<i>How was it addressed in your SLP?</i>	<i>Where was it addressed in your SLP?</i>
<b>1. An ability to identify, formulate, and solve complex engineering problems by applying principles of engineering, science, and mathematics</b>		
1.1 An ability to apply knowledge of mathematics	<i>Battery sizing</i>	<i>Chapter 3 section 3.5</i>
1.2 An ability to apply knowledge of Science	<i>Calculation of speed and torque in motor selection</i>	<i>Chapter 3 section 3.3.2</i>
1.3 An ability to apply knowledge of Engineering	<i>Mechatronics System Design</i> <i>Robot Operating Systems</i> <i>Arduino Programming</i>	<i>SLP Report</i>
<b>2. An ability to apply engineering design to produce solutions that meet specified needs with consideration of public health, safety, and welfare, as well as global, cultural, social, environmental, and economic factors</b>		
2.1 Design a system/component of a system or a process to meet specific needs while respecting safety, health and welfare of the public and adhering to cultural, social, environmental and economic factors.	<i>Applying safety factor in the design criteria to choose the safest and most affordable components</i>	<i>Chapter 3</i>

<p>2.2 Modify a system/component of a system or a process to meet specific needs while respecting safety, health and welfare of the public and adhering to cultural, social, environmental and economic factors.</p>	<p><i>Collection mechanism several iterations in design and implementation</i></p>	<p><i>Chapter 3 section 3.4</i></p> <p><i>Chapter 4 section 4.4</i></p>
<p><b>3. An ability to communicate effectively with a range of audiences</b></p>		
<p>3.1 Ability to write a well-structured formal report/technical document that addresses an audience with diverse educational-background</p>	<p><i>Senior Report</i></p>	<p><i>SLP report (Wednesday, April 27, 2022)</i></p>
<p>3.2 Ability to deliver a well-structured formal presentation that addresses an audience with diverse educational-background<sup>1</sup></p>	<p><i>Demo Presentation</i></p>	<p><i>SLP report (Wednesday, April 27, 2022)</i></p>
<p><b>4. An ability to recognize ethical and professional responsibilities in engineering situations and make informed judgments, which must consider the impact of engineering solutions in global, economic, environmental, and societal contexts</b></p>		
<p>4.1 Identify global, economic, environmental, and societal impact of implementing engineering solutions using applicable engineering code of ethics to differentiate between ethical/unethical behaviors</p>	<p><i>Design a safe collection mechanism that is safe for humans and don't harm slugs through collection</i></p>	<p><i>Chapter 4 section 4.4</i></p>
<p>4.2 Identify global, economic, environmental, and societal impact of implementing engineering solutions using applicable engineering standards and codes to differentiate between professional/unprofessional behaviors</p>	<p><i>Isolation of wiring of the collection mechanism and other sensors</i></p> <p><i>Punctuality in attending meetings.</i></p>	<p><i>Chapter 4</i></p> <p><i>Appendix-F</i></p>

<b>5. An ability to function effectively on a team whose members together provide leadership, create a collaborative and inclusive environment, establish goals, plan tasks, and meet objectives</b>		
5.1 Ability to plan and organize team tasks collectively to meet established goals	<i>All agreed upon actions are properly documented in minutes and work flow is properly managed by the Gantt chart on ClickUp</i>	<i>Appendix-F</i>
5.2 Ability to carry out tasks assigned by the team to attain set objectives.	<i>Clear and efficient task assigning process was followed. In addition, team members abided by the agreed upon time plan to ensure efficient accomplishment of all tasks in due time</i>	<i>Appendix-F</i>
<b>6. An ability to develop and conduct appropriate experimentation, analyze and interpret data, and use engineering judgment to draw conclusions</b>		
6.1 An ability to design experiments.	<i>Design iterations of collection mechanism</i>	<i>Page xx or sec. xx</i>
6.2 An ability to conduct experiments.	<i>Implementation trials of the</i>	<i>Chapter 5 Appendix-A</i>

	<i>collection mechanism</i>  <i>Testing GPS Kit and Camera to ensure accuracy</i>	<i>Appendix-B</i>
6.3 An ability to draw apt evidence-based conclusions by analyzing and interpreting data	<i>Marvelmind GPS Beacon</i>  <i>MPU 60-50</i>  <i>Pi Camera</i>	<i>Appendix A</i>  <i>Appendix B</i>
<b>7. An ability to acquire and apply new knowledge as needed, using appropriate learning strategies.</b>		
7.1 Identify necessary skills and tools of contemporary engineering practice to solve a problem at hand.	<i>Design and implementation of the robot and its collection mechanism from scratch</i>	<i>Chapter 4</i>
7.2 Apply self-learned skills and tools of contemporary engineering practice to solve a problem at hand.	<i>Our Navigation Approach</i>  <i>Our Detection Approach</i>	<i>Chapter 2 Sections 2.6.1 and 2.6.2</i>  <i>Appendix A</i>  <i>Appendix B</i>

# APPENDIX F

## MEETING MINUTES

### MINUTES OF MECA/MECH 595A MEETING (1)

#### COLLEGE OF ENGINEERING – MME Department – RHU

#### Group I

#### ON OCTOBER 8<sup>th</sup>, 2021 AT 05:00 PM

---

**Present:** Dr.Hasan Hariri (Advisor), Dr.Jad Nasreddine (Advisor), Afif swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student) , Abdul Rahim El Mohamed (student), Yahya Al Jamal (Student)

**Absent:** Ziad Chelala (Sponsor), Khalil Raei (Sponsor), Ramzi Halabi (Adbvisor)

---

The meeting came to order at 05:00 pm.

#### ***1. Discussions and Updates***

---

Discussed general issues about the conceptual project idea. Afif introduced the management software (Clickup), and disussed briefly the technical and non-technical requirements.

#### ***2. Advisor Comments and Recommendations***

---

Dr.Hariri and Afif agreed that the project will be cutdown into subsystems, and our target at this stage is the proof of concept and the following are the requirements discussed:

- 1- Robot should move on mud, detect, collect, and store slugs without harming them
- 2- Five suggested subsystems:
  - Design
  - Computer Vision (for slug detection)
  - Navigation
  - Product Design (Market research, costumer needs, international standards...)

#### ***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 15/10/2021 :

- 1- General Information about slugs
- 2- Market research (competitors and publications)
- 3- Drivetrain choices

The meeting was adjourned at 05:45PM

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (2)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**  
**Group I**  
**ON OCTOBER 15<sup>th</sup>, 2021 AT 05:00 PM**

---

**Present:** Dr.Hasan Hariri (Advisor), Dr.Jad Nasreddine (Advisor), Ziad Chelala (Sponsor), Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student) , Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Khalil Raei (Sponsor), Ramzi Halabi (Advisor)

---

The meeting came to order at 05:00 pm.

***1. Discussions and Updates***

---

Ziad and Afif informed the team with all the technical and non-technical requirements. Students presented information about competitors, computer vision and drivetrain.

***2. Advisor Comments and Recommendations***

---

Concerning the competitors, further details are needed regarding the price, area covered by the robot and if its industrial or not. As for the computer vision part, more data should be collected and Ghassan from Spexal will join the next scheduled meeting to advise the team. Regarding the drivetrain, after eliminating multiple possibilities, all agreed on skid steering with either four wheels or continuous tracks. It is recommended to work on the collecting mechanism to decide the drivetrain.

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 27/10/2021:

- 1- Slug collection mechanism
- 2- Computer vision
- 3- More details about competitors

The meeting was adjourned at 06:35PM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (3)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**  
**Group I**  
**ON NOVEMBER 3<sup>rd</sup>, 2021 AT 11:00 AM**

---

**Present:** Dr.Hasan Hariri (Advisor), Dr.Jad Nasreddine (Advisor), Khalil Rai (Sponsor), Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student) , Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Ziad Chelala (Sponsor), Ramzi Halabi (Advisor)

---

The meeting came to order at 11:00 am.

### ***1. Discussions and Updates***

---

Mr. Afif met with the team on Sunday, 31 October 2021 and assigned tasks on the Gantt chart. Yahya posted the survey on Clickup to get Spexal's feedback (Dr Hariri approved the survey).

### ***2. Advisor Comments and Recommendations***

---

Concerning the navigation, Mr. Khalil suggested three different approaches:

- 1- Autonomous Robot
- 2- External performer server (costumer's laptop/phone)
- 3- Integration between performer server and performer robot.

Concerning Computer Vision, team will proceed in learning about computer vision, and Mr. Khalil suggested organizing a workshop with Mr. Ghassan (Computer vision expert from Spexal team) to compensate for any technical gap.

### ***3. Expected Deliverables for Next Meeting***

---

Dr Hariri assigned the following tasks to work on due for the next scheduled meeting on 10/11/2021:

- 1- Edit and finalize the Gantt Chart
- 2- Edit the survey according to Spexal's feedback
- 3- Conduct a detailed comparison between the Navigation approaches.
- 4- Conduct research regarding user experience with existing autonomous robot.
- 5- Proceed in learning Computer Vision

The meeting was adjourned at 11:50AM.

Minutes taken by: Jana Marzouk



**MINUTES OF MECA/MECH 595A MEETING (4)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**  
**Group I**  
**ON JANUARY 4<sup>th</sup>, 2021 AT 10:00 AM**

---

**Present:** Mr.Ghassan Mouhaidly (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Dr.Hasan Hariri (Advisor), Dr.Jad Nasreddine (Advisor), Mr.Afif Swaidan (Sponsor), Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Dr.Ramzi Halabi (Adbvisor)

---

The meeting came to order at 10:00 am.

***1. Discussions and Updates***

---

Team discussed with Mr. Ghassan the detection part of the project to agree on the best approach.

***2. Advisor Comments and Recommendations***

---

The following topics were discussed:

1- Data set:

Team presented the collected photos with Mr. Ghassan. He considers that the collected photos are not efficient enough because the slugs are zoomed in and won't be useful in detection. Therefore, team should collect a dataset of 150-200 pictures that are most far, where slugs are not so close, to apply object detection.

2- Label:

Mr. Ghassan recommended the following resources for labelling the slugs in the photos.

- labelme
- cocoannotator
- labellmg

3- Training:

Team recommended classification for the training, but Mr. Ghassan sees that classification will only show us if it is a slug or not without providing its location. Thus, he recommends applying object detection that will classify and localize the slug.

We agreed that Object Detection will be done using yoloV5

4- Camera selection:

Concerning the camera that will be used, Kinect 2 is a time-of-flight and it won't be efficient in light.

Another good option is D435.

As for RGB camera only, it depends on the collection mechanism.

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next meeting:

- 1- Dataset collection
- 2- Dataset Labelling

The meeting was adjourned at 10:40 AM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (5)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**

**Group I**

**ON JANUARY 5<sup>th</sup>, 2022 AT 11:00 AM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Mr.Ghassan Mouhaidly (Sponsor), Dr.Jad Nasreddine (Advisor), Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Dr.Ramzi Halabi (Advisor)

---

The meeting came to order at 11:00 am.

***1. Discussions and Updates***

---

Team shared with Dr.Hariri and Mr.Afif design sketches for the collection mechanism. Also, team shared the updates on computer vision workshop

***2. Advisor Comments and Recommendations***

---

Concerning the Collection mechanism, team presented the possible designs. Dr.Hariri and Mr.Afif recommended preparing an excel sheet that includes for each design: number of motors, type of gripper, type of end effector, complexity (range of +-), accuracy, dust (range of +-), RGB or RGBD, price, if mechanism depends on the size of the slug or not.

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 12/01/2022 :

- 1- Sign off sheet for the collection mechanism design sketches
- 2- Updates on slugs finding
- 3- Minimum and maximum size of slugs

The meeting was adjourned at 12:10 AM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (6)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**  
**Group I**  
**ON JANUARY 18<sup>th</sup>, 2022 AT 01:25 PM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Ziad Chelala (Sponsor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Mr.Ghassan Mouhaidly (Sponsor), Mr.Khalil El Rai (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor)

---

The meeting came to order at 01:25 pm.

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the navigation workshop and the progress in detection. Also team discussed with Dr.Hariri and Spexal the collection mechanism.

***2. Advisor Comments and Recommendations***

---

After a detailed discussion between Dr.Hariri and Spexal, both agreed on the Bulldozer for collection mechanism as a first step and upon testing we decide if we want front or back storage. Team should finalize their decision to proceed in choosing the needed material, and to form the bill of materials the soonest.

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 25/01/2022:

- 1- Proceed in navigation: apply on the turtlebot what we learned in the navigation workshop.
- 2- Proceed in Computer vision: finalize the dataset and labelling.
- 3- Choose collection mechanism.
- 4- Search for drivetrain and collection material

The meeting was adjourned at 02:30 PM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (7)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**

**Group I**

**ON JANUARY 25<sup>th</sup>, 2022 AT 01:25 PM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Mr.Ghassan Mouhaidly (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor)

---

The meeting came to order at 01:25 pm.

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the progress in navigation and detection. Also team discussed with Dr.Hariri and Spexal the collection mechanism.

***2. Advisor Comments and Recommendations***

---

Team shared with Dr.Hariri and Spexal the following concerns regarding the Bulldozer-collection mechanism:

- 1- Regarding the environment, is it completely muddy terrain or is there concrete paths? i.e should the robot collect slugs only from the soil or also from concrete pathways
- 2- After collection, slugs will stick on the bulldozer and won't be able to free them in the storage
- 3- Bulldozer will definitely collect unwanted pebbles/soil, and the storage will be full faster, then the user will be asked to clean the storage more frequently
- 4- Collection slugs using Bulldozer will change the pattern of the soil in the backyard

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 01/02/2022:

- 1- Design, 3D Print, and test the Roller mechanism.
- 2- Spexal response regarding bulldozer mechanism
- 3- Proceed in navigation and its documentation.
- 4- Proceed in computer vision: finalize the dataset and labelling.

The meeting was adjourned at 01:55 PM

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (8)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**

**Group I**

**On March 1<sup>st</sup>, 2022 AT 1:25 PM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student)

**Absent:** Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Mr.Ghassan Mouhaidly (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor), Yahya Al Jamal (Student)

---

The meeting came to order at 01:25 pm.

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the progress in collection, chassis design, and detection.

***2. Advisor Comments and Recommendations***

---

1- Drivetrain:

- Team showed Dr.Hariri and Spexal the cad drawing of the robot, and team asked if the upper aluminum profile can be removed and add an L shape metal with screws instead.( to decrease the weight minimally, the weight of the robot with profiles is 2.8kg while 2.4kg without them so upper profiles are around 15% of total weight)  
Dr.Hariri and Spexal agreed that additional weight will add grip and friction on the soil, so its recommended to keep the upper aluminum profiles if there's enough torque.
- Dr.Hariri raised the issue of dust across the motors and electric components, and all agreed that dust will not be taken into consideration in the Proof of Concept phase but it can be added in the "Future Work" part of the final report.
- Regarding the position of the wheels and motors, advisors recommended that its always better for them to be close and form a square shape to decrease slippage. Therefore, team should edit the design and try in the hardware assembly phase to take this constructive feedback into consideration. Also, Spexal offered help of Mr.Chadi to check the Cad design when the team finalize it.

2- Collection

- Team reported to Dr.Hariri and Spexal that the collection mechanism will consist of roller, ramp and storage as one piece in order to make the storage near the ground and less distance between the ramp and the storage.

- Testing was done as different distances between the ramp and roller, and variable motor speed; the closer the ramp to the roller, the better. Also, even if we raise the robot from ground by flipping the motors the collection mechanism won't be affected.
- Dr.Hariri also asked if the robot have enough space for the storage and collection? Team reported that as a first step we decided the design of the collection mechanism(1Block) and next step is the integration of the mobile platform, collection, and storage.

### ***3- Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 08/03/2022:

- 1- Edit the mobile platform CAD drawing and discuss with Spexal ordering the components
- 2- Integration of the collection mechanism and the storage with the mobile platform
- 3- Block Diagram and Bill of materials after finalizing the components needed
- 4- Battery sizing for the robot to run for 1hr in area of 25x20 m<sup>2</sup>
- 5- Material of inner storage

The meeting was adjourned at 02:15 PM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (9)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**  
**Group I**  
**On March 8<sup>th</sup>, 2022 AT 1:25 PM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student) ), Yahya Al Jamal (Student)

**Absent:** Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Mr.Ghassan Mouhaidly (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor)

---

The meeting came to order at 01:25 pm.

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the progress in collection, chassis design and components selection.

***2. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 08/03/2022:

- 1- Check rpm-torque graph
- 2- Look for gear ratio from datasheet
- 3- Modify power calculations and block diagram
- 4- Finalize BOM and order components
- 5- Battery sizing for the robot to run for 1hr in area of 25x20 m<sup>2</sup>

The meeting was adjourned at 02:10 PM.

Minutes taken by: Jana Marzouk



**MINUTES OF MECA/MECH 595A MEETING (10)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**

**Group I**

**On March 16<sup>th</sup>, 2022 AT 9:30 AM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student) ), Yahya Al Jamal (Student)

**Absent:** Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Mr.Ghassan Mouhaidly (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor)

---

The meeting came to order at 09:30 am.

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the progress in collection, calculations, detection.

***2. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 22/03/2022:

- 1- Start assembling the drivetrain of the robot.
- 2- Finalize the collection mechanism.
- 3- Finalize the calculations according to final collection mechanism and edit the calculations according to 4x4 aluminum profiles instead of 2x2.
- 4- Proceed in detection.
- 5- Integrate detection with navigation on the turtlebot.
- 6- Proceed with BE report.

The meeting was adjourned at 10:20 AM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING (11)**  
**COLLEGE OF ENGINEERING – MME Department – RHU**

**Group I**

**On March 30<sup>th</sup>, 2022 AT 1:25 PM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student), Yahya Al Jamal (Student)

**Absent:** Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Mr.Ghassan Mouhaidly (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor)

---

The meeting came to order at 01:25 pm.

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the progress in collection, and detection.

***2. Advisor Comments and Recommendations***

---

1- Collection:

Team showed Dr.Hariri and Spexal the CAD drawing of the collection mechanism integrated with the drivetrain. All agreed to widen the collection mechanism to 10cm instead of 7cm. Also a metal servo motor is good as it has decent torque for the collection. Moreover, all agreed that trusses must be included in the mechanism to strengthen it.

2- Vision:

Team updated Dr.Hariri regarding the vision part. Currently, team is working on the model improvement along with Mr. Ghassan Moheidly's guidance  
Dr.Hariri recommends applying the integration of vision and navigation on the turtlebot3 before applying it on the real robot.

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 05/04/2022:

- 1- Edit the CAD drawing of the collection mechanism
- 2- Navigation Implementation on the real robot
- 3- Vision Model Finalization
- 4- Check the real working environment

The meeting was adjourned at 02:15 PM.

Minutes taken by: Jana Marzouk

**MINUTES OF MECA/MECH 595A MEETING**  
**COLLEGE OF ENGINEERING – MME Department – RHU**  
**Group I**  
**ON APRIL 12<sup>th</sup>, 2022 AT 11:00 AM**

---

**Present:** Dr.Hasan Hariri (Advisor), Mr.Afif Swaidan (Sponsor), Jana Marzouk (Student), Nabil Miri (Student), Abdul Rahim El Mohamed (Student)

**Absent:** Mr.Ziad Chelala (Sponsor), Mr.Khalil El Rai (Sponsor), Mr.Ghassan Mouhaidly (Sponsor), Dr.Ramzi Halabi (Advisor), Dr.Jad Nasreddine (Advisor), Yahya Al Jamal (Student)

---

***1. Discussions and Updates***

---

Team updated Dr.Hariri regarding the progress in detection and collection mechanism. Also team discussed with Dr.Hariri some concerns regarding the final report.

***2. Advisor Comments and Recommendations***

---

- 1- Detection: Team updated Dr.Hariri and Spexal regarding the latest improvements in the model
- 2- Collection: Spexal gave feedback on the CAD drawing of the collection mechanism, and it will be printed within 24 hours

***3. Expected Deliverables for Next Meeting***

---

Assigned to work on due for the next scheduled meeting on 19/04/2022:

- 5- Collection implementation
- 6- Navigation, detection and collection integration

The meeting was adjourned at 11:35 AM.

Minutes taken by: Jana Marzouk

## References

- Addison. (2021, June 27). *Automatic Addison* . Retrieved from Automatic Addison : <https://automaticaddison.com/what-is-an-occupancy-grid-map/>
- CADCAM Services . (2022). Retrieved from CAD/CAM Services : <https://www.cadcam.org/blog/robotics-design-using-cad/>
- Ed., P. B. (2007). *Pests of Field Crops and Pastures Identification and Control*. CSIRO .
- Gabbay, D. M., Thagard, P., & Woods, J. (2009). Philosophy of Technology and Engineering Sciences. In S. O. Hansson.
- Hensel, O. (2016). *UNIVERSITÄT*. Retrieved from uni-kassel: <https://www.uni-kassel.de/fb11/11-agrar/en/sections/-/facilities/agrartechnik/completed-research-projects/2016-msr-bot>
- Husarion Docs* . (n.d.). Retrieved from Husarion Docs : <https://husarion.com/tutorials/ros-tutorials/7-path-planning/>
- Hutton, S. (n.d.). *CHAP*. Retrieved from chap-solutions: <https://chap-solutions.co.uk/news/slugbot-aims-to-put-an-end-to-slimy-pests/>
- Ininiti*. (n.d.). Retrieved from Ininiti Electro-Optics : <https://www.ininitioptics.com/contact>
- Introducing Marvelmind Ultrasonic Indoor Navigation System*. (2020, December 24). Retrieved from seed studio: <https://www.seedstudio.com/blog/2019/10/17/introducing-marvelmind-ultrasonic-indoor-introrobotics> . (2013). Retrieved from <https://www.intorobotics.com/wheels-vs-continuous-tracks-advantages-disadvantages/>
- Jacob, S., Menon, V., & Joseph , S. (2020). Depth Information Enhancement Using Block Matching and Image Pyramiding Stereo Vision Enabled RGB-D Sensor. *IEEE*.
- Khosrow-Pour, M. (2017). *Encyclopedia of Information Science and Technology, Fourth Edition*. USA.
- Last Minute Engineers*. (2021). Retrieved from <https://lastminuteengineers.com/1298n-dc-stepper-driver-arduino-tutorial/>
- Marvelmind*. (2019). Retrieved from <https://marvelmind.com/>
- N.Shalal, & Low, T. (n.d.). *A REVIEW OF AUTONOMOUS NAVIGATION SYSTEMS IN AGRICULTURAL CORE*.
- NickLamprianidis. (2018, September 13). *OpenRobotics*. Retrieved from roswiki: [http://wiki.ros.org/twist\\_mux](http://wiki.ros.org/twist_mux)
- NOSTOP ELECTRIC CO*. (n.d.). Retrieved from <http://www.nostopmotor.com/upload/P2a2fbf19054349809874c6291d9a5326.pdf>
- PowerSonic*. (2022). Retrieved from <https://www.power-sonic.com/blog/what-is-a-battery-rating/#:~:text=The%20battery%20C%20Rating%20is,10%20Amps%20for%20one%20hour.>
- PyTorch*. (n.d.). Retrieved from PyTorch.
- Raspberry Pi*. (2016). Retrieved from <https://www.raspberrypi.com/products/camera-module-v2/>
- Somareddy, G. (2017, June 5). *Quora*. Retrieved from <https://www.quora.com/What-is-the-coefficient-of-friction-between-mud-and-a-tyre#:~:text=The%20coefficient%20of%20friction%20between%20the%20tire%20and%20mud%20is%20about%200.158%20.&text=The%20coefficients%20of%20friction%20of,to%20decrease%20with%20incr>

Swaidan, A., Baradhi , R., Mechlawi , W., & Yahya, R. (2021). *A ROS Powered Autonomous Vehicle Utilizing Indoor GPS*. Lebanon : Rafik Hariri University .

Wearden, G. (2002). *cnet* . Retrieved from cnet : <https://www.cnet.com/culture/robot-eats-slugs-to-generate-power/>