




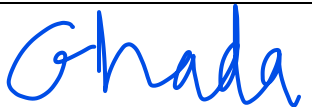

Biomedical Wastes Handler Robot

MCG 5138 – Mobile Robotics

Fall 2023

Dr. Amirhossein Monjazebe, P.Eng

Design and Development of a Robotic Sorting System for A Production Line

#	Group member's full last and first name	UOIT student ID number	Signature
1	Mariam Hussien	300389400	Mariam 
2	Ibrahim Elshenhapy	300389386	Ibrahim 
3	Ahmed Asem	300389894	Ahmed 
4	Ghada Salah	300389364	Ghada 
5	Shady Osama	300389887	shady 

Executive summary

In response to the critical need for efficient and safe handling of hazardous materials, our mobile robotics project utilizing TurtleBot and ROS2 Foxy emerges as a cutting-edge solution for emergency scenarios. Dealing with hazardous materials demands a delicate and meticulous approach, and our project is designed to adhere to the highest standards of care. Special policies and procedures are rigorously followed, reflecting a commitment to both environmental stewardship and the safety of response teams.

The primary objective of our project is to automate the intricate task of handling hazardous materials in challenging environments characterized by complexity and uncertainty. By leveraging the capabilities of TurtleBot and the ROS2 Foxy framework, we aim to significantly reduce response times and minimize risks associated with human exposure to potentially dangerous substances.

Key components of our project include advanced navigation, the establishment of keep-out zones, visual servoing for object detection, and strategies for pick-and-place operations. The integration of these components creates a cohesive and sophisticated system capable of addressing the nuanced challenges posed by handling hazardous materials.

In the domain of navigation, our project employs a combination of A* algorithm, Navigation 2, and waypoint planning to optimize path traversal in dynamic environments. This approach ensures that the robot can navigate efficiently and adapt to unforeseen obstacles, critical in emergency response situations.

The establishment of keep-out zones is a paramount aspect of our project, contributing to spatial awareness and ensuring the robot avoids areas deemed hazardous. Utilizing image processing, map conversion techniques, and dynamic cost map updates, our system actively responds to changes in the environment, guaranteeing a real-time and adaptive response to evolving conditions.

Visual servoing, a sophisticated computer vision technique, is incorporated to enable the robot to precisely detect and manipulate hazardous materials. Specifically, the system focuses on detecting blue-colored rectangles, enhancing the accuracy and reliability of object recognition.

To achieve seamless material handling, our project integrates pick-and-place strategies with robotic manipulator control. This combination enables the robot to execute precise movements, ensuring the secure and controlled relocation of hazardous materials.

Nodes integration, a cornerstone of ROS2 Foxy, is implemented through ROS2 topics and actions. This integration facilitates communication and collaboration among the various components of the robotic system, ensuring a synchronized and cohesive operation.

While the project exhibits a robust design and comprehensive approach, challenges such as issues related to the ROS Foxy Fitzroy distribution and the need for real-time cost map updates have been identified. Ongoing efforts are dedicated to addressing these challenges, with a commitment to continuous improvement and refinement of the system.

In conclusion, our mobile robotics project is a testament to the innovative application of technology in addressing critical challenges associated with hazardous materials. By combining state-of-the-art robotics with meticulous adherence to safety protocols, our project represents a significant advancement in the field of emergency response, with potential implications for broader applications in environmental protection and human safety.

Table of Contents

Executive summary 2

Table of Contents	2
Introduction and Background	4
Problem description and Objective	4
1- Map Design:	5
a. Design the map in Gazebo.	5
2- Robot Control.....	5
a) Navigation	5
b) Detect Redlines	6
c) Detect container with visual servoing.....	8
d) Pick and place.....	10
e) Implement Manipulator controllers to control the arm joints on turtlebot3.....	11
f) Create launch files that run all nodes and parameters, enabling a complete project launch.	12
3- Extra Credit	13
Conclusion.....	14
References	16

Introduction and Background

Proper handling of biomedical waste is crucial to minimize the risk of accidents and injuries. Facilities must adhere to regulations, train employees, use suitable containers, and ensure proper labeling and packaging for safe transportation and disposal. [1]. Mobile robot systems offer a solution to the challenges faced in warehouses by providing agile, flexible, and scalable order fulfillment. They improve productivity and throughput while reducing operating risks and costs compared to manual operations. [2]. The mobile robot will do the following:

- Navigate through map till the containers and pick up.
- Place the containers in the desired place.

Problem description and Objective

The handling of hazardous materials presents a significant challenge in emergency response scenarios, where time-sensitive and meticulous actions are required to mitigate risks effectively. Traditional approaches to dealing with hazardous materials often involve manual intervention, exposing human responders to potential dangers and slowing down response times. Furthermore, the complexity of confined spaces and the dynamic nature of emergency environments exacerbate the difficulties in executing precise and safe material handling tasks.

In facilities dealing with hazardous materials, special policies and procedures are mandated to ensure the utmost care and compliance with safety regulations. Despite these precautions, there remains a need for innovative solutions that can automate and streamline the process of handling hazardous materials in emergency situations. The objective is to reduce the reliance on human intervention in potentially dangerous environments and, in turn, enhance the efficiency and safety of emergency response operations.

- Objectives:

The overarching objective of our mobile robotics project is to develop a sophisticated and autonomous system that addresses the challenges associated with the handling of hazardous materials in emergency response scenarios. The specific objectives include:

- Automation of Material Handling:

Develop an automated system capable of handling hazardous materials without direct human intervention. The system should exhibit precision and reliability in material manipulation tasks.

- Reduction of Response Time:

Implement advanced navigation algorithms and strategies to significantly reduce response times. The robot should efficiently navigate through complex and confined spaces, reaching the target location swiftly.

- Minimization of Human Exposure:

Prioritize the safety of emergency responders by minimizing their exposure to hazardous materials. Achieve this through the robot's ability to execute material handling tasks autonomously, thereby reducing the need for human proximity to dangerous substances.

- Spatial Awareness and Avoidance:

Establish an intelligent system with robust spatial awareness capabilities. Utilize keep-out zones and dynamic cost map updates to ensure the robot can adapt to changing environmental conditions, avoiding hazardous areas effectively.

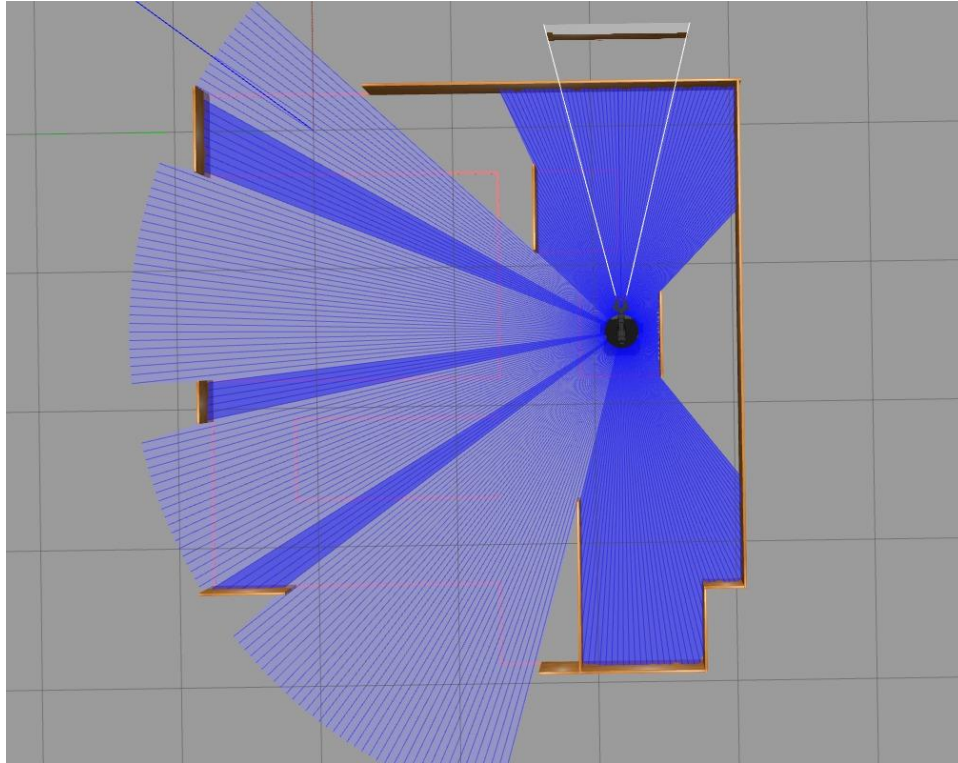
- Computer Vision for Object Detection:

Integrate visual servoing and computer vision techniques to enhance the robot's ability to detect hazardous materials accurately. Focus on the detection of specific visual cues, such as blue-colored rectangles representing hazardous objects.

Detailed Design Documentation

1- Map Design:

- a. Design the map in Gazebo.



2- Robot Control

a) Navigation

Navigation refers to the process of determining and controlling the movement of a vehicle, robot, or individual within an environment, with the goal of reaching a desired destination or completing a specific task. This involves the use of sensors, algorithms, and decision-making mechanisms to plan and execute a path, avoiding obstacles and adapting to the dynamic nature of the surroundings. Navigation is a fundamental aspect of autonomous systems, encompassing various techniques such as path planning, obstacle avoidance, and localization to enable effective and safe movement.

We've implemented the A* algorithm for navigation with the TurtleBot3 to perform a specific task of picking up a can and placing it in a goal pose. Using A* for navigation is a common and effective approach, especially when you need to find an optimal path from a starting pose to a goal pose.

Here's a breakdown of the process:

- A* Path Planning:

A* (A-star) is a popular pathfinding algorithm that finds the shortest path between a starting point and a goal point on a grid or graph. In our case, it's likely used to plan the robot's path from its initial pose to the location of the can and then to the goal pose.

- Localization:

To execute the planned path accurately, TurtleBot3 needs to know its current pose within the environment. This is achieved through localization methods, such as odometry and sensor fusion techniques that combine data from wheel encoders, IMU (Inertial Measurement Unit).

- Obstacle Avoidance:

While navigating, the TurtleBot3 needs to avoid obstacles in its path. A* generates a path, but the robot should adapt to dynamic obstacles or changes in the environment. Obstacle avoidance mechanisms, possibly integrated into the A* implementation or as a separate layer, ensure the robot can navigate around or react to unexpected obstacles.

- Pick and Place Actions:

Once the robot reaches the location of the can (pick-up pose), it needs to execute the pick-up action. This involves mechanisms such as robotic arms with a gripper to grasp the can securely.

- Goal Pose:

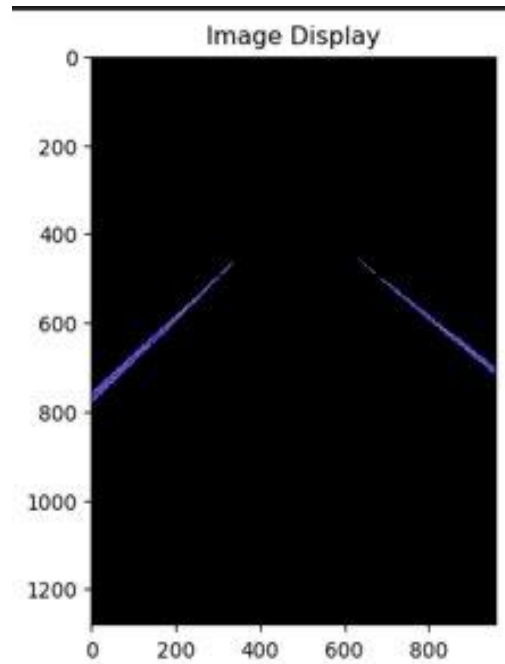
After picking up the can, the robot then follows the A* planned path to the goal pose. At the goal pose, the robot performs the place action, releasing the can in the desired location.

- Task Execution Loop:

The entire process is likely part of a task execution loop where the robot continuously plans paths, executes actions, and monitors its environment to complete the desired task efficiently.

b) Detect Redlines

We're describing a scenario where a robot uses camera feedback to detect specific red lines in its environment and employs a Keep Out Zone algorithm to avoid moving over or too close to these detected red lines. This type of implementation is common in various robotic applications, including robotics in manufacturing, logistics, or other environments where certain areas are restricted.



Here's a general outline of how such a system might work:

- Camera Feedback:

The robot is equipped with a camera or a vision system capable of capturing images or video frames of its surroundings.

- Image Processing:

Image processing algorithms are applied to the camera feed to identify and segment red lines in the images. This can involve color-based segmentation or more advanced computer vision techniques.

- Line Detection:

The processed images are analyzed to detect the presence and location of red lines within the field of view. Various computer vision algorithms, such as edge detection or Hough transform, may be used for this purpose.

- Keep Out Zone Generation:

Based on the detected red lines, Keep Out Zones are generated. These zones represent areas that the robot should avoid or navigate around to prevent crossing the red lines.

- Path Planning with Keep Out Zones:

The Keep Out Zones are integrated into the robot's path planning algorithm. When planning a trajectory from its current location to a target destination, the robot considers the presence of Keep Out Zones and plans a path that avoids these restricted areas.

- Dynamic Updates:

The system may be designed to dynamically update the Keep Out Zones based on real-time camera feedback. For example, if the red lines move or change shape, the robot adapts its Keep Out Zones accordingly.

- Control and Navigation:

The robot's control system uses the planned trajectory to navigate through the environment while avoiding the detected red lines. This involves adjusting the robot's velocity and steering commands to follow the planned path.

- Feedback Loop:

The system operates in a closed-loop fashion, continuously receiving camera feedback, updating Keep Out Zones, and adjusting the robot's trajectory as needed to ensure compliance with the restrictions imposed by the red lines.

c) Detect container with visual servoing

We have a robotic system that uses computer vision (specifically OpenCV, denoted as cv2) to detect a can based on its blue color, and then employs an approach strategy to navigate towards the can until it is reached, at which point the robot picks it up. Here's a breakdown of the process:

- Color Detection using OpenCV:

You are using OpenCV to process the camera feed and identify pixels corresponding to the blue color, allowing you to isolate the can from the background. This may involve techniques such as color thresholding or more advanced color space transformations.

- Object Localization:

Once the blue color is detected, the algorithm likely performs additional steps to identify and localize the can within the camera frame. This may involve contour detection, blob analysis, or other image processing techniques.

- Approach Strategy:

After the can is localized, a strategy is employed to approach the can. This strategy might include determining the distance to the can and adjusting the robot's position or orientation to align itself with the can.

- Closed-Loop Control:

The approach strategy likely involves closed-loop control, where the robot continually adjusts its motion based on real-time feedback from the camera. This allows the robot to adapt to variations in the environment or changes in the can's position.

- Distance Estimation:

The algorithm may estimate the distance to the can based on the visual information. This distance estimation is crucial for controlling the approach and ensuring that the robot reaches a desired proximity to the can.

- **Pick-Up Action:**

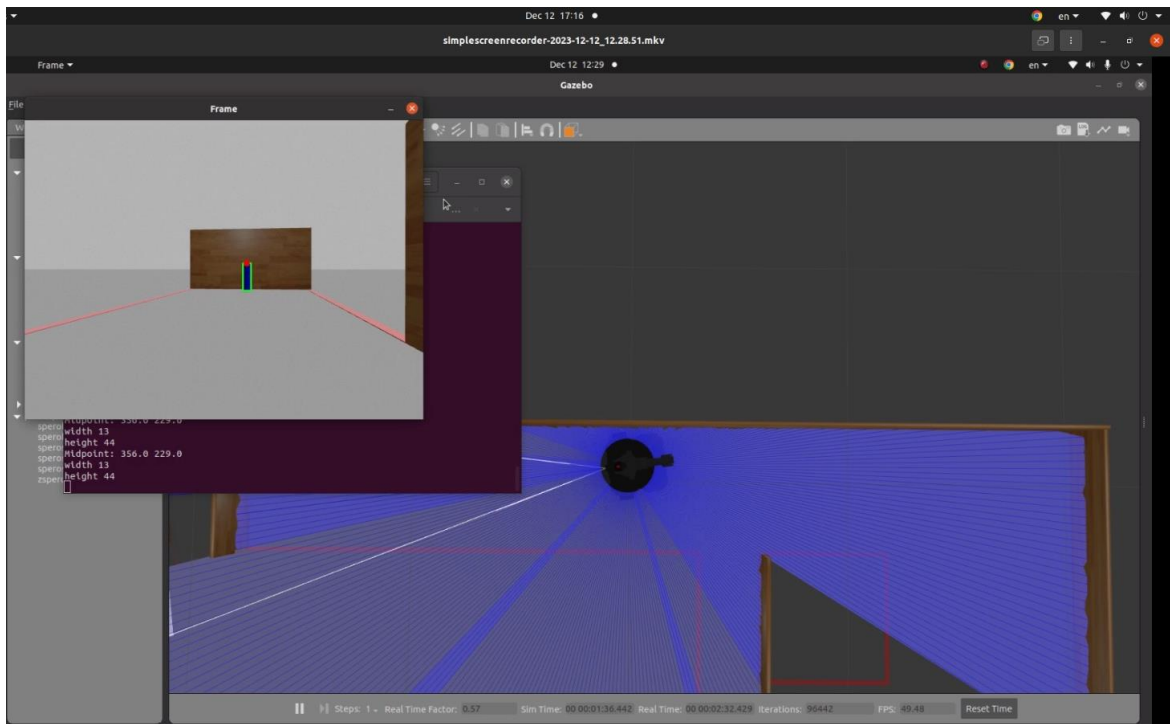
Once the robot has successfully approached the can, it performs the pick-up action. This might involve using a gripper or another mechanism to grasp the can securely.

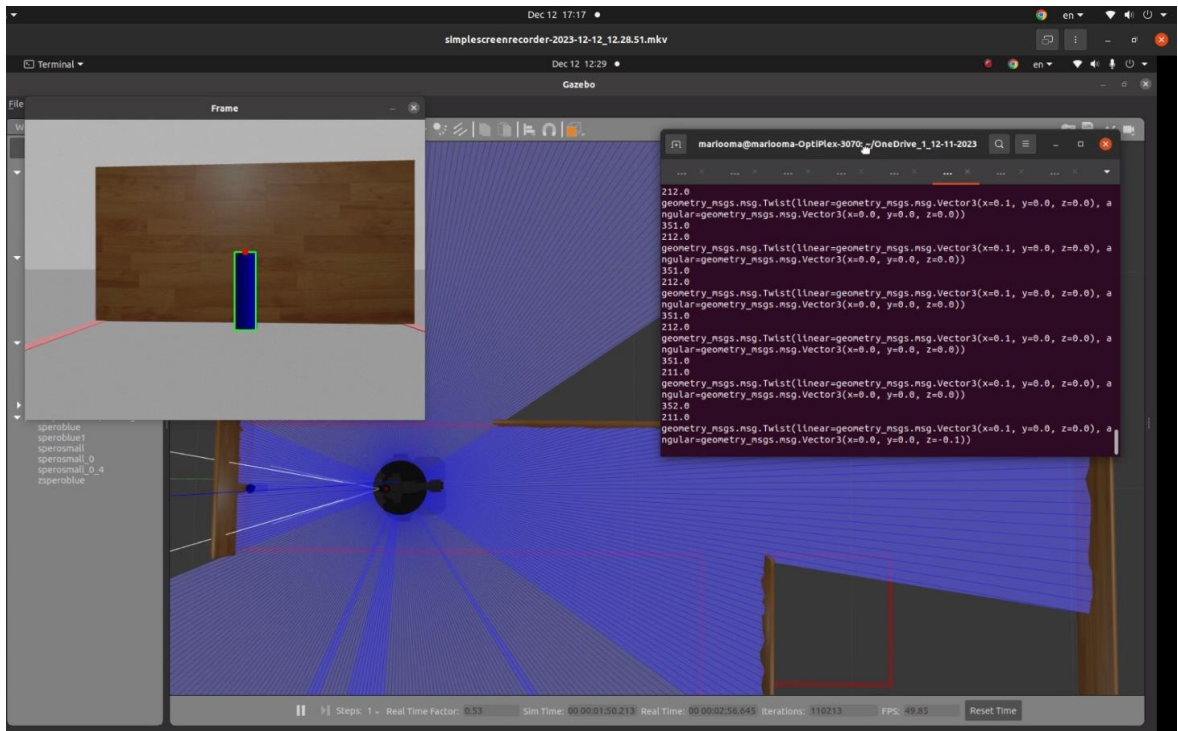
- Integration with Robot Control System:

The entire process, including color detection, object localization, approach, and pick-up, is likely integrated into the overall robot control system. This integration ensures coordination with other robot modules and components.

- Error Handling and Robustness:

The system may incorporate error handling mechanisms and robustness features to handle variations in lighting conditions, potential false positives or negatives in color detection, and other uncertainties.





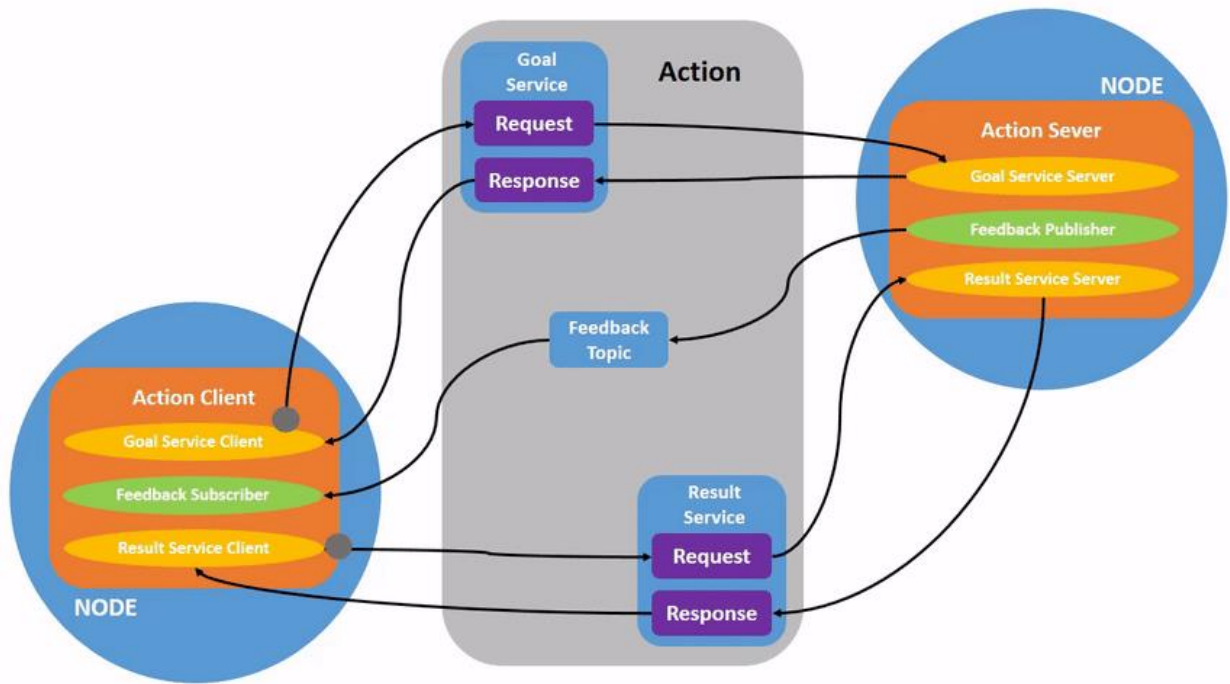
d) Pick and place

In environments where hazardous materials are handled, such as toxic fumes or radioactive materials, safe manipulation and disposal are crucial. To achieve this, a manipulator is employed to pick up and place objects, keeping them away from human contact and minimizing human exposure.

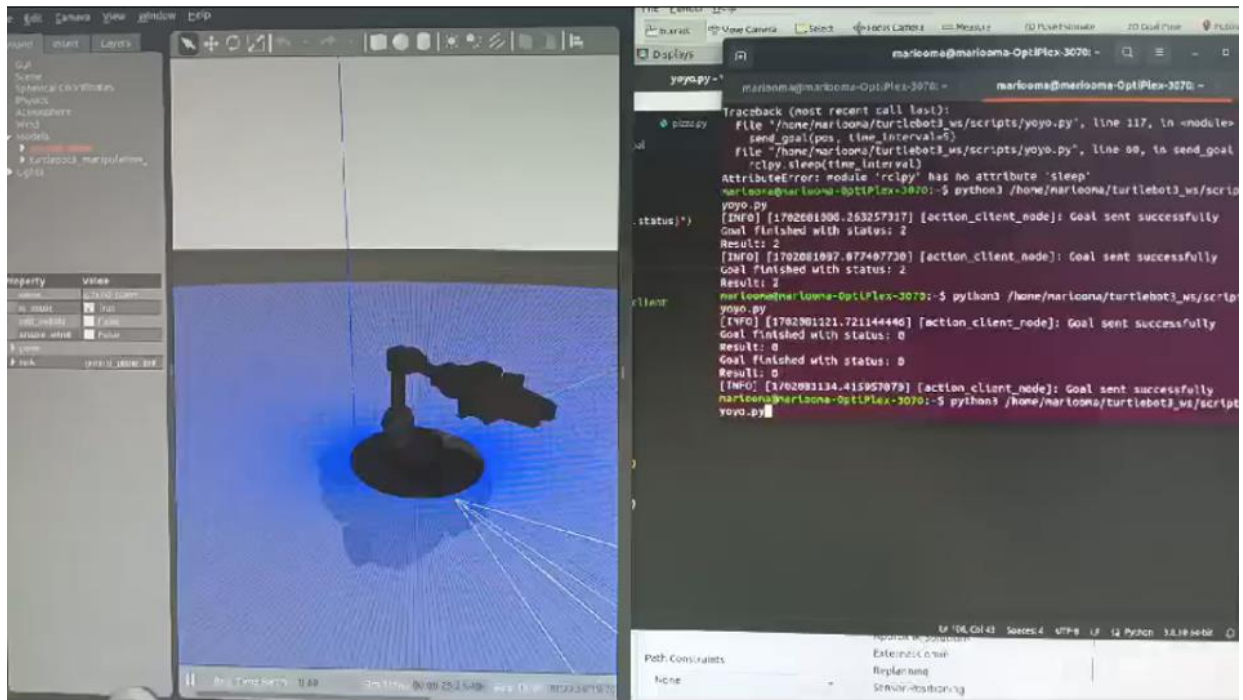
The pick and place task in robotics is a tricky task as it needs calculations of the joint angles for the arm to move to the specified position using inverse kinematics or utilizing the position of the detected object and plan the trajectory using forward kinematics. Our approach was to specify the angles or positions for each joint of the manipulator using inverse kinematics rather than relying on the position of the detected object and planning the trajectory through forward kinematics. This decision is motivated by the desire for trajectory stability, as changes due to noise or environmental variations could potentially interfere with the external camera and disrupt the visual servoing framework, impacting algorithm performance. Additionally, the selected package for specifying positions and moving to them is not supported for ROS2 Foxy.

During the manipulator movement process we had to choose and settle on an approach for operation, we encountered challenges with the availability of packages, particularly the removal of moveit2 and pymoveit packages for ROS2 Foxy. Exploring alternative solutions, we considered rewriting an existing package to support ROS2 Foxy. However, this proved to be a complex and uncertain task, given the numerous dependencies requiring modifications.

Another avenue explored was studying ROS 2 actions, which are communication mechanisms designed for long-running tasks. Actions utilize a client-server model, enabling the sending of goals and receiving a stream of feedback and results. The identified actions for the manipulator were `/arm_controller/follow_joint_trajectory` for the arm and `/gripper_controller/gripper_cmd` for the gripper. Understanding the message format required for these actions, we successfully sent actions through the terminal.



Having determined the necessary actions, types, and message formats, we implemented the control through Python code. We developed a function that takes joint variables, simplifying the execution of the required sequence. This approach allowed us to overcome challenges posed by the unavailability of specific packages for ROS2 Foxy, ensuring effective manipulator control in hazardous material environments.



e) Implement Manipulator controllers to control the arm joints on turtlebot3

In reality, the challenges of implementing turtlebot3 manipulation packages on the physical TurtleBot differed from those encountered in simulation. To enable manipulation, the correct setup and execution of turtlebot3 manipulation packages on the TurtleBot were imperative. However, we faced issues related to errors in packages and dependencies, requiring extensive troubleshooting efforts.

Various solutions were attempted to address the problems, such as uninstalling and reinstalling packages, reformatting the SD card image on the Raspberry Pi, and then burning a fresh image of Ubuntu 20 with ROS2 Foxy following the instructions in the robot's e-manual. Additionally, we verified the package integrity and re-uploaded the OpenCR board sketch. Despite these efforts, resolving the issues was a time-consuming process, involving repeated attempts and troubleshooting iterations.

Once the setup was stabilized, the hard part and subsequent challenge was determining precise joint positions. This involved a trial-and-error approach based on known positions for the joints. Systematically, we moved one joint at a time, recording positions until achieving the optimal configuration for both the picking and home positions.

The picking sequence comprised four steps:

- The initial position (0 position or home position) for all joints.
- Moving just above the can.
- Opening the gripper.
- Manipulator movement to pick up the can, followed by gripper closure, and returning to the home position.

The placing sequence was essentially the reverse of the picking task. This intricate process required careful calibration and adjustment to ensure accurate and successful manipulation in the real-world environment.



f) Create launch files that run all nodes and parameters, enabling a complete project launch.

In the development of our robotic project, the creation of launch files plays a pivotal role in orchestrating the seamless execution of various nodes and parameters. Launch files in ROS (Robot Operating System) provide a convenient way to start multiple nodes simultaneously, set parameters, and configure the robot's behavior. Let's delve into how launch files are crafted to encompass the entire project launch, covering the bring-up of the TurtleBot3, navigation, visual servoing for can detection, and manipulator pick-up and place actions:

- TurtleBot3 Bring-Up:

A launch file is designed to initiate the bring-up of the TurtleBot3, ensuring that all necessary hardware components are connected and initialized. This involves starting nodes responsible for communication with sensors, actuators, and the robot's base.

- **Navigation Algorithm:**

Another launch file is crafted to execute the navigation algorithm. This file launches nodes related to the navigation stack, including the global and local planners, odometry, and sensor fusion components. Parameters are set to define the robot's kinematics, sensor characteristics, and the target point for navigation.

- **Visual Servoing for Can Detection:**

The launch file for visual servoing integrates the computer vision nodes responsible for detecting the can based on its blue color. This file starts the camera node, image processing nodes (utilizing OpenCV), and the visual servoing algorithm. Parameters may include color thresholds and camera calibration information.

- **Manipulator Pick-Up and Place:**

A launch file is created to coordinate the nodes required for the manipulator's pick-up and place actions. This involves starting nodes responsible for kinematics, trajectory planning, and control of the manipulator. Parameters are configured to define the pick-up and place poses, as well as the characteristics of the manipulator.

3- Extra Credit

Understanding the significance of speed in the competition, where faster execution earns bonus points, our team successfully completed all tasks within an impressive time frame of less than 7 minutes—precisely 6 minutes and 51 seconds. There was potential to further decrease this time by increasing the speed of execution, but we opted for a cautious approach to ensure the flawless performance of each task. This decision reflected our commitment to precision and reliability in task execution, even while operating within a time-constrained competition environment.

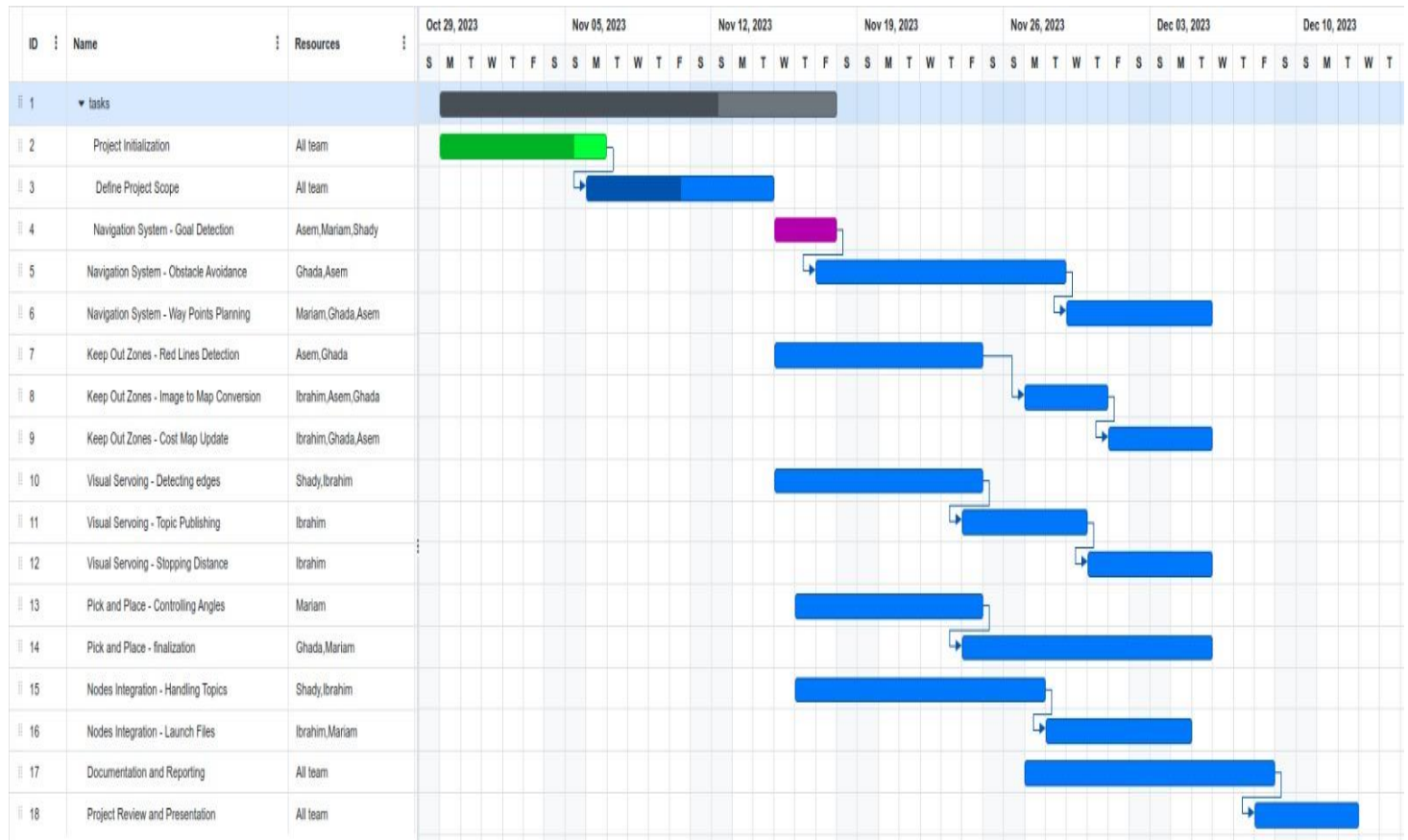
Gantt Chart

The implementation of turtlebot3 manipulation packages represents a dynamic and multifaceted project that intertwines robotics, software development, and real-world application. This endeavor centers around configuring and deploying turtlebot3 manipulation packages on a physical TurtleBot, navigating through challenges in setup, troubleshooting, and calibration. The overarching goal is to create a robust robotic system capable of precise picking and placing tasks in an environment involving hazardous materials.

This Gantt chart serves as a visual roadmap, outlining the project's timeline, task dependencies, and key milestones. As we delve into the intricacies of configuring the turtlebot3 manipulation packages, troubleshooting package errors, and fine-tuning joint positions, the Gantt chart will provide a comprehensive overview of the project's progression. From the initial setup phase to the execution of pick and place sequences, each phase is meticulously planned to ensure a systematic and efficient workflow. The timeframes allocated for each task reflect a balance between speed and precision, acknowledging the competition's bonus points for swift task completion while prioritizing the reliability of each operation.

By visually mapping out the project timeline and task interdependencies, this Gantt chart aims to facilitate project management, enhance coordination among team members, and provide a clear trajectory for achieving project

objectives. As we embark on this venture, the Gantt chart serves as a valuable tool for monitoring progress, identifying potential bottlenecks, and ultimately steering the project towards successful completion.



Conclusion

In conclusion, the autonomous system developed for container movement on the TurtleBot3 platform represents a pioneering endeavor to revolutionize the handling of hazardous materials. The core objective of this project is to alleviate the challenges and risks associated with manual material handling by introducing an advanced, autonomous solution. Through the amalgamation of cutting-edge technologies, including localization, mapping, and robotic control techniques, our system strives to redefine safety, efficiency, and operational protocols within facilities dealing with hazardous materials.

Advancing Safety Protocols: The implementation of an autonomous system is poised to significantly enhance safety protocols in hazardous material handling. By reducing the need for direct human intervention, the risk of accidents and potential exposure to dangerous substances is inherently minimized. This proactive approach aligns with the industry's commitment to prioritizing the health and safety of workers, emergency responders, and the surrounding environment.

Accident Prevention and Risk Mitigation: The project's focus on automation is inherently tied to the prevention of accidents and the mitigation of risks associated with handling hazardous material. By introducing a robot capable of navigating through confined spaces, identifying hazardous materials, and executing precise movements for material manipulation, the potential for accidents stemming from manual errors or unforeseen circumstances is substantially reduced.

Operational Efficiency Enhancement: Beyond safety considerations, the autonomous system is geared towards enhancing operational efficiency within facilities dealing with hazardous materials. The system's ability to autonomously navigate through spaces, establish keep-out zones, and execute pick-and-place actions streamlines material handling processes. This not only accelerates response times in emergency scenarios but also contributes to overall operational efficiency and resource utilization.

Practical Application of Robotics: The project serves as a practical demonstration of the application of robotics in the context of hazardous material handling. By successfully implementing localization and mapping techniques alongside sophisticated robotic control strategies, the system showcases the adaptability and versatility of robotic technologies in real-world scenarios. This demonstration underscores the potential for robotics to redefine operational paradigms across various industries.

Future Implications and Continuous Improvement: Looking ahead, the implications of this project extend beyond its immediate application. The successful development of an autonomous system for handling hazardous material sets the stage for future advancements and innovations in the field of robotics. Continuous improvement and refinement of the system will be pursued, addressing challenges, incorporating feedback, and ensuring that the technology remains at the forefront of safety and efficiency standards.

In essence, the autonomous system for container movement on the TurtleBot3 platform encapsulates a transformative approach to hazardous material handling. As we conclude this project, we anticipate its positive impact on safety practices, operational efficiency, and the broader integration of robotics in addressing complex challenges across industrial sectors. The journey does not end here; instead, it propels us toward a future where autonomous solutions play a pivotal role in creating safer and more efficient workplaces.

Acknowledgements

The successful conception, development, and execution of this mobile robotics project have been made possible through the collaborative efforts and support of various individuals and institutions. The acknowledgment section

serves as a heartfelt expression of gratitude to those who have contributed significantly to the realization of this endeavor.

1. Project Team: We extend our deepest gratitude to the dedicated members of the project team who have worked tirelessly in the conception, design, and implementation phases. Their commitment, expertise, and collaborative spirit have been instrumental in achieving the project's objectives.

2. Institutional Support: Our sincere appreciation goes to DEBI, which has provided the necessary resources, infrastructure, and encouragement for the successful execution of this project. Institutional support has played a pivotal role in fostering an environment conducive to innovation and excellence.

3. Faculty Advisors: We express our gratitude to our esteemed faculty advisors, Dr. Amirhossein Monjazez, P. Eng, for their guidance, mentorship, and valuable insights throughout the project's development. Their expertise and commitment to our academic and research pursuits have been invaluable in shaping the project's trajectory.

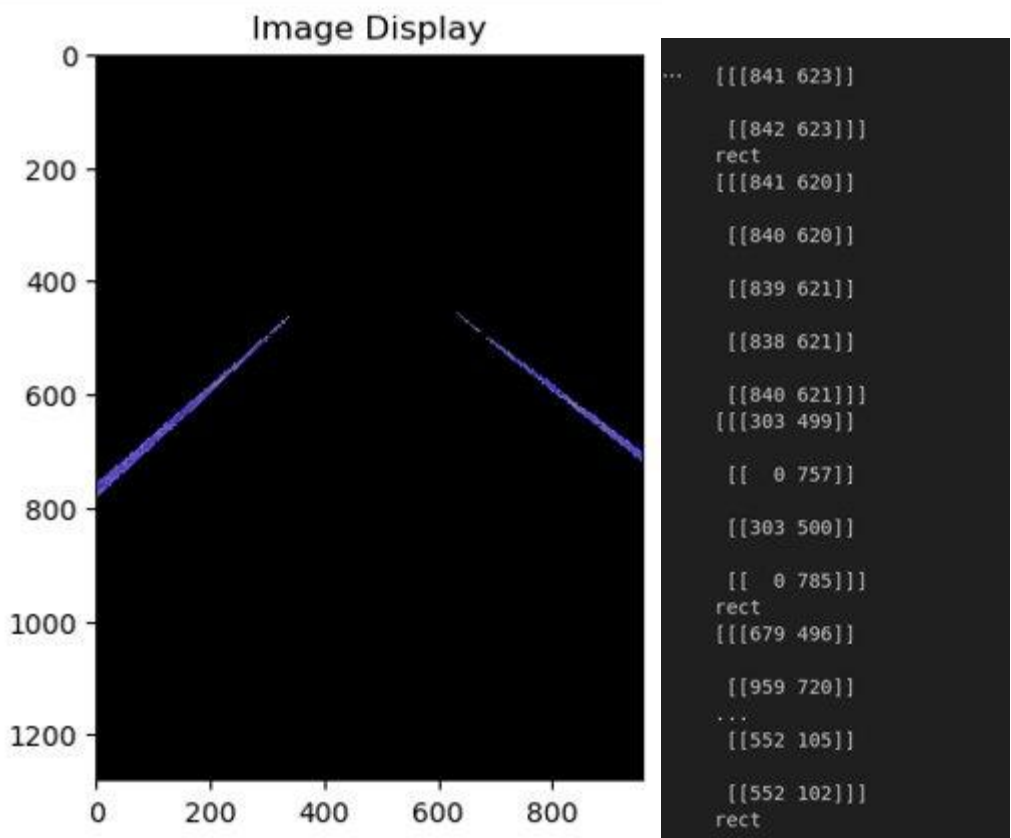
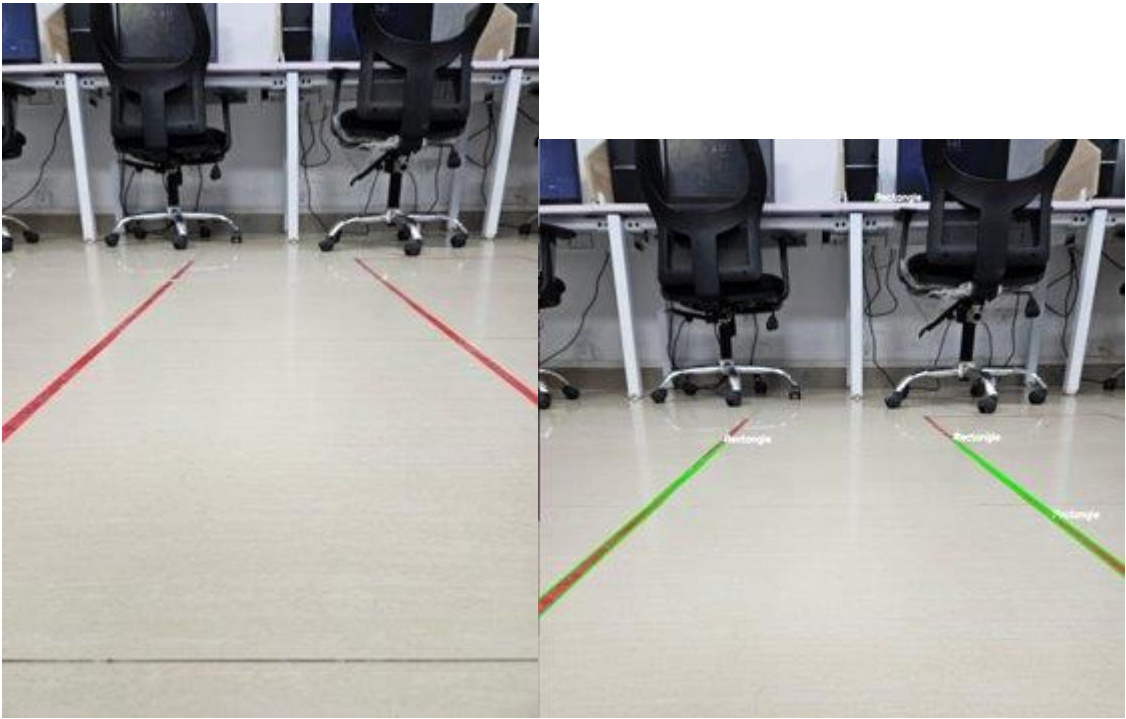
4. Collaborators: A special thanks to our collaborators and partners who have contributed expertise, feedback, and collaborative efforts to enhance the project's scope and impact. The synergy resulting from these collaborations has enriched the project's outcomes.

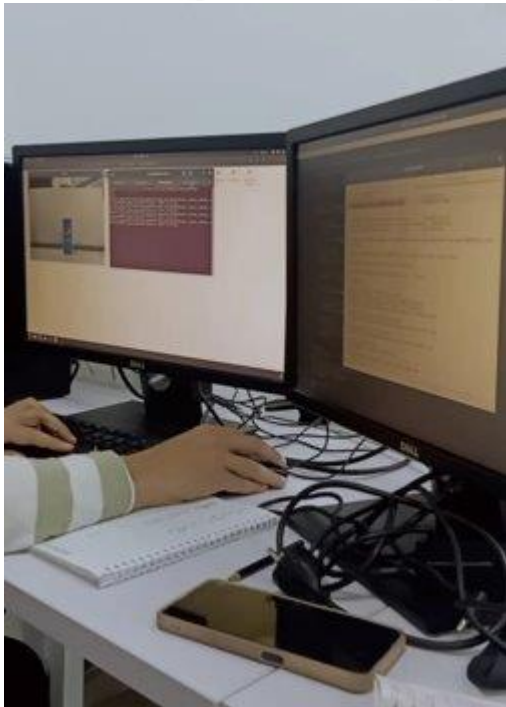
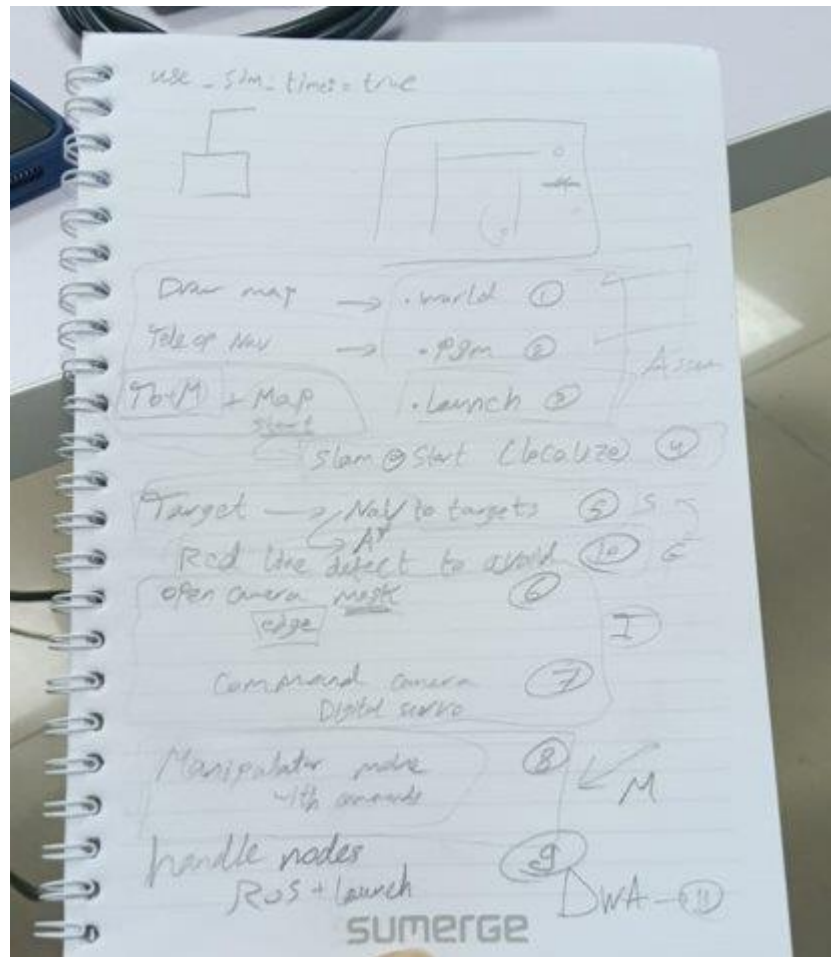
In conclusion, the successful completion of this mobile robotics project is a testament to the collective efforts of a diverse and dedicated community. Each individual and entity mentioned above has played a unique and crucial role in bringing this project to fruition. As we acknowledge their contributions, we also recognize the collaborative spirit that underlies the success of this endeavor.

References

1. <https://wiki.ros.org/Documentation>
2. <https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/>
3. A Concise Introduction to Robot Programming with ROS2 – Taylor and Francis group
4. A. R. C, A. P. M, R. K. C and A. Mohanarathinam, "Clinical Waste Storage System with Contamination Prevention Mechanism using UV and Arduino Microcontroller" 2023 International Conference on Inventive Computation Technologies (ICICT), Lalitpur, Nepal, 2023, pp. 1585-1591, doi: 10.1109/ICICT57646.2023.10134053.
5. A. Yildirim, H. Reefke, and E. Aktas, "Mobile Robot Automation in Warehouses: A Framework for Decision Making and Integration," Palgrave Studies in Logistics and Supply Chain Management, 1st ed., Palgrave Macmillan Cham, 2023. [Online]. Available: <https://doi.org/10.1007/978-3-031-12307-8>
6. A.-T. Nguyen and C.-T. Vu, "Obstacle Avoidance for Autonomous Mobile Robots Based on Mapping Method," in Faculty on Mechanical Engineering, Hanoi University of Industry, Hanoi, Vietnam, 2021. Faculty on Mechanical Engineering, Hanoi University of Industry, Hanoi, Vietnam, 2021. Available: <https://tinyurl.com/5azymbw9>

Appendices





Enter RGB hex code (#):

or

Enter red color (R):

Enter green color (G):

Enter blue color (B):

Hue (H): °

Saturation (S): %

Value (V): %

Color preview: 