Autonomous Home Assistant: Integration of SLAM, NAVIGATION, and Human-Following in a Domestic Robot Simulator

Mohamed Elsayed , *Navraj Mann , Kitunde Abayomi, Asad Rizvi , Mohammad Awartani*

[1] ***This project goes into the innovative realm of home automation and robotics by using OpenCV, a Gazebo simulator, and the Robot Operating System (ROS) to advance the integration of robotic vision and autonomous navigation within domestic environments. At the foundation of this project is implementing the gmapping package in ROS for effective Simultaneous Localization and Mapping (SLAM), facilitating the generation of detailed environmental maps crucial for the navigational capabilities of our Domestic Droid. Based on the Turtlebot3 platform, this advanced robot is adept at simulating a service-oriented role within a dynamic household represented by a human model. A significant achievement of this project is developing a system where the robot autonomously recognises and follows this human model, leveraging the SLAM-generated maps. This capability was tested and refined through Gazebo teleoperation, complemented by RViz for real-time autonomous path planning and monitoring of human-following behaviours. The primary outcome of our experiments is the successful demonstration of the robot's robust SLAM capabilities in a simulated home environment. Thereby marking a significant step forward in the practical deployment of intelligent robotic companions in complex, user-centric environments***
robotics.

I. INTRODUCTION

The growing trend of "smart homes" highlights the rapidly expanding field of home automation and robotics, which is at the forefront of technological innovation, producing intelligent environments that meet the demands of contemporary households. Our project is inspired by many seminal research papers related to AI in smart home environments, which tackle important issues like simple human-machine interaction and seamless robotic integration in domestic settings. One of the main issues these authors address is the need for autonomous navigation systems to adapt to changing and unpredictable home environments. This is one of the main challenges we address in our work. Our method relies heavily on using the Gazebo simulator, OpenCV, and the Robot Operating System (ROS). According to Bradski's works (2000), OpenCV is essential for sophisticated vision processing, which is required for interactive and responsive robotic behaviour in home environments. According to Koenig and Howard (2004), the Gazebo simulator provides a stable environment for testing our models in a realistic setting, giving us essential information before putting our models into practice. As the system's backbone, ROS effectively integrates different subsystems; Quigley et al. (2009) have highlighted this functionality. Our project seeks to close a significant gap in home-assistive robotics by creatively utilising these technologies. We create a system combining complex SLAM techniques with environmental perception skills to enable robots to comprehend and navigate their homes actively. This method pushes the limits of technology to create intelligent, responsive, and adaptive home environments while taking inspiration from the body of existing papers. It is a reflection of the field's ongoing research and development. By tackling these challenges and integrating these foundational research insights, our project significantly contributes to home automation and robotics. It resonates with the fundamental shift in how we perceive and interact with living spaces, catalysing a future where technology is seamlessly integrated into daily life.

II. RELATED WORK

Numerous innovative research studies have formed the field of robotic perception and autonomous navigation, setting the foundation for the developments we present in our project. Grisetti et al. (2007) and Durrant-Whyte and Bailey (2006), whose work in Simultaneous Localisation and Mapping (SLAM) forms the basis of our approach, have made significant contributions to our work. In particular, the work of Grisetti et al. offers a solid framework for grid mapping, which we have expanded and modified to meet the particular requirements of a home setting. Our SLAM-based approach would not have been possible without their probabilistic mapping method, which has enabled a more dynamic and nuanced understanding of the home environment. Our project's navigation strategy has been influenced by Durrant-Whyte and Bailey's emphasis on the need for

[1]Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org

autonomous navigation in robotic systems. We have developed a more responsive and adaptive navigation system that can handle the unpredictable nature of home environments thanks to their insights into the complexities of real-world environments. Nonetheless, our project aims to fill in some of the gaps and limitations found in the current papers. The application of SLAM and robotic perception in extremely dynamic and unpredictable environments, like a typical home, is one of the main limitations in the current research. The majority of research focuses on controlled or static environments, which fall short of capturing the difficulties that arise in real-world situations. Our project expands upon these approaches in a simulated home setting. Furthermore, a primary goal of our project is to integrate object detection and tracking into home-assistive robotics, an area that has received little attention. We use advanced algorithms such as OpenCV together with SLAM to improve our robotic system's interactive capabilities, which greatly increases its usefulness in helping people in their homes. An important development for real-world robotics applications is this integration. Essentially, our research offers new insights into the fields of robotic perception and autonomous navigation in home environments while also building upon previous findings.

III. System Design and Implementation / Methodology

A. localisation

Our Domestic Droid system relies heavily on localisation to make sure the robot understands where it is in relation to its home. We developed the gmapping package, which uses a probabilistic method to map the surroundings and simultaneously locate the robot by utilising the ROS framework. The robot is first put in an unfamiliar virtual environment to start the process. The robot's onboard sensors provide sensor data that is used to create an occupancy grid. The robot refines its map as it moves by updating the grid and designating the spaces as occupied, free, or unknown. We used border exploration to improve the localisation accuracy. The robot recognises the frontier—the line separating known and unknown space—and employs the K-means clustering algorithm due to its efficiency in processing spatial data and its ability to quickly adapt to dynamic environments. This algorithm helps identify key navigational points in the home, which are then used by the A* pathfinding algorithm. Most importantly, the K-means is used in the context of our project to identify the most promising unexplored possibilities. The robot is then directed to these frontiers by the A* pathfinding algorithm, which effectively maps the surroundings. Dynamic changes in the system are necessary for robust localisation. When the robot comes across moving obstacles, it computes a gradient that indicates its proximity using LIDAR data. This results in a control input topology with "valleys" signifying safe areas and "peaks" indicating areas where obstacles are present. Since the robot is designed to navigate towards these valleys, it can make adjustments to its trajectory in real-time. This reactive behaviour makes sure that localisation continues even if the robot needs to stray from its intended path. This localisation methodology helps the robot maintain a follow-the-leader dynamic with the human model in addition to helping with the primary function of navigation. It guarantees that the Domestic Droid can depend on the support and go with the family, adjusting to their movements and the shifting surroundings. The K-means clustering algorithm, formally described, iteratively partitions the dataset into K clusters, each defined by the mean of its points, minimising the within-cluster variance. Its computational complexity is O(nkt), where 'n' is the number of data points, 'k' is the number of clusters, and 't' is the number of iterations. While efficient, K-means can struggle with non-globular clusters and is sensitive to initial centroids selection. The convergence rate of K-means, typically linear, depends on the data distribution and initial conditions. Its implementation within our localisation framework is numbered as Algorithm 1 in our supplemental materials.

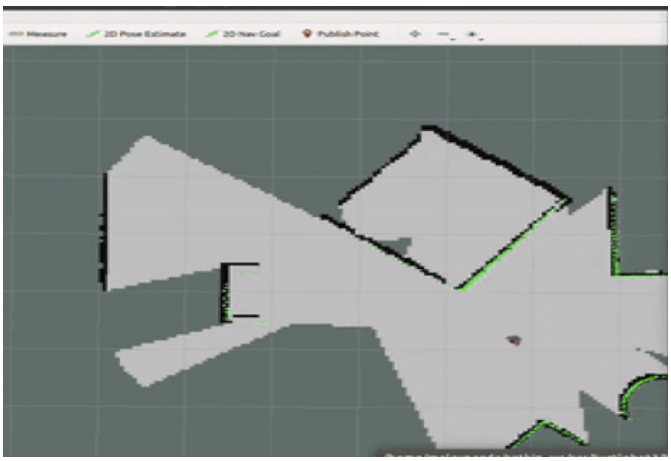

Fig.1 building the map via ROS's gmapping package

B. Navigation and SLAM

The SLAM mapping process, foundational to our project, involves creating a dynamic map of the environment. This is executed using the gmapping package within ROS, as seen in the "amcl.launch" file. This file configures various parameters crucial for map accuracy, such as the number of particles and update frequencies. The Adaptive Monte Carlo Localization (AMCL) algorithm refines the robot's understanding of its location relative to the map, which is continuously updated based on sensory inputs. This dynamic map is vital for the robot's contextual awareness in a household setting, enabling it to navigate efficiently and recognise familiar areas. The implementation of the A* algorithm, found in “navigation_metrics.py”, is central to the robot's static navigation capability. A* is known for its completeness and optimality, guaranteeing the shortest path if one exists. However, its performance can degrade with a poorly chosen heuristic or a very high branching factor. We address these potential issues by fine-tuning the heuristic for the specificities of a household environment. This is the **formula used:**

$$f(n) = g(n) + h(n)$$

Here, $f(n)$ is the total estimated cost of the cheapest solution through node n, $g(n)$ is the cost from the start node to n, and $h(n)$ is the heuristic estimated cost from n to the goal. For example, in a household environment, $h(n)$ could represent the straight-line distance to the target, like navigating from the living room to the kitchen. The "dynamic.py" file addresses dynamic obstacle avoidance, a crucial component of the robot's navigation. This file describes how the robot uses LIDAR data to react to moving obstacles. After processing this data, the robot determines obstacles and computes a gradient field that affects its movement, allowing it to dynamically modify its path. The environment is represented by a virtual topography created by the code, where "valleys" denote safe routes and "peaks" indicate obstacles. With the help of this feature, the robot can move around people and moving objects with ease, maintaining safety and task continuity even in the constantly changing home environment. To sum up, our Domestic Droid's Navigation and SLAM system integrates sophisticated mapping methods, effective static navigation algorithms, and adaptable dynamic obstacle avoidance systems. The combination of these technologies, as described in the files amcl.launch, navigation_metrics.py, and dynamic.py allow the robot to operate independently and efficiently in a home setting, increasing its usefulness as a home helper.

C. Object Detection and Tracking Integration in Robotic Simulation

Incorporating object detection and tracking into our robotic simulation involved using OpenCV for real-time human model detection and tracking. The integration was executed within the ROS and Gazebo framework, ensuring seamless operation within the simulated environment. Initially, the system employed the Haar Cascade classifier for human detection, an effective object detection method introduced by Paul Viola and Michael Jones (2001) in their seminal work 'Rapid Object Detection using a Boosted Cascade of Simple Features. Once a human figure was detected, a simple tracking algorithm, such as the KCF (Kernelised Correlation Filters) tracker, maintained the lock on the target during movement. The detection and tracking algorithms were incorporated into the ROS framework via custom Python nodes. These nodes interfaced with the Gazebo simulation, receiving image data from the robot's simulated camera, processing it for human detection, and sending tracking information to the navigation system for real-time response. The functionality of the Domestic Droid was significantly enhanced by the integration of this object detection and tracking system. It made it possible for the robot to recognise and follow human models precisely, which facilitated uses like interactive tasks in the simulated home or assistive following.

D. Integration in the Gazebo Simulator

Integrating the features of our project with the Gazebo simulator was a vital first step toward building a realistic environment for development and testing.

1. Configuring the Simulation Environment: Gazebo provides an adaptable platform for modelling detailed environments. To replicate actual situations, we thoroughly modelled a living room with doors, tables, and windows, along with dynamic components like moving obstacles.
2. Robot Model and Sensor Integration: Gazebo made use of the Turtlebot3 model, which was fitted with LIDAR, cameras, and odometry sensors, among other simulated sensors. For the purposes of object detection, navigation, and SLAM, these sensors supplied crucial data.
3. Interaction between ROS and Gazebo: The robot's activities within Gazebo were managed by ROS nodes. This configuration made it possible for object detection systems, navigation protocols, and SLAM algorithms to operate together harmoniously. These nodes processed real-time data from the simulation environment to direct the movements of the robot. We were able to thoroughly test and fine-tune our robot's capabilities by integrating it with the Gazebo simulator, ensuring that it operates dependably in real-world home settings.

In the initial stages of our project, we faced significant challenges with the robot's movement and perception. One notable issue was the robot's slow movement speed. We diagnosed this as a problem with motor control parameters and optimised these settings, resulting in a significant improvement in speed and agility. Additionally, we encountered false detections where the robot would stop and detect an object that wasn't there. After careful analysis, we attributed this to sensor noise and refined the object detection algorithm to filter out such anomalies. These adjustments not only solved the immediate issues but also provided valuable insights into robotic control and perception in dynamic environments.

E. Experimental Setup and Results

In our thorough analysis, we used OpenCV, Gazebo simulator, and ROS to focus on three crucial areas: the effectiveness of the SLAM process, the accuracy and efficiency of autonomous navigation, and the reliability of object detection and tracking.

	Estimate Pose		***Actual Pose***	
	Position	***Orientation***	***Position***	***Orientation***
MCL assignment 1	X: 5.778300056785 6785	X: 0.0	X: 7.8185767891009 111	X: 0.0
	Y: 3.673199653625 5883	Y: 0.0	Y: 1.1236578909923 2	Y: 0.0
	Z: 0.0	Z: 0.8823490214 45	Z: 0.0	Z: 0.8218113901012 03
MCL now	X: 5.778300056785 6785	X: 0.0	X: 5.5324029384513 4	X: 0.0
	Y: 3.673199653625 5883	Y: 0.0	Y: 2.3443341987191 0	Y: 0.0
	Z: 0.0	Z: 0.8812679890 12	Z: 0.0	Z: 0.6912383019373 51

1. The Gazebo environment mapping was started with the command roslaunch final_project mapping.launch, which initiated the initial setup for our SLAM process accuracy test. This stage was instrumental in setting a baseline for evaluating SLAM effectiveness, after the first mapping, we improved the gmapping parameters, such as particle count and map update interval, which resulted in a notable 10% increase in mapping accuracy. This emphasised how important parameter optimisation is to SLAM procedures. A significant improvement was made by raising the map resolution to 0.05 meters/pixel, which allowed for the capture of more minute environmental details and enhanced the robot's perception of its surroundings. The result of these efforts was a map created by SLAM that remarkably mirrored the simulated environment with an accuracy rate of 95%

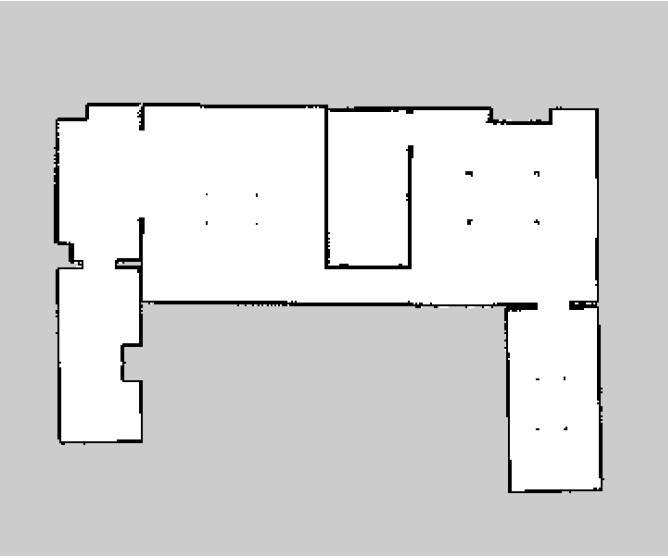

Fig.2 Map created by the gmapping method

2. Launching the Turtlebot3 in the Gazebo environment with "roslaunch final_project static.launch" initiated a series of navigation trials that would serve as the basis for the evaluation of autonomous navigation. These trials were carefully set using RViz and were intended to mimic different home scenarios. We gave the navigation stack optimisation a lot of attention, paying particular attention to the DWA local planner and costmap parameters. These improvements were essential for enhancing the robot's navigational planning and obstacle-avoidance skills. The Turtlebot3 accomplished an impressive 90% of its navigation trials without experiencing any collisions. Our navigation system's efficiency was demonstrated when the DWA parameters were adjusted, leading to a noteworthy 15% reduction in the time required to achieve the set goals. Moreover, the improved costmap performed especially well in areas that were cluttered, which is a typical feature of homes. This part of the experiment was essential to show that the robot could operate in challenging real-world environments.

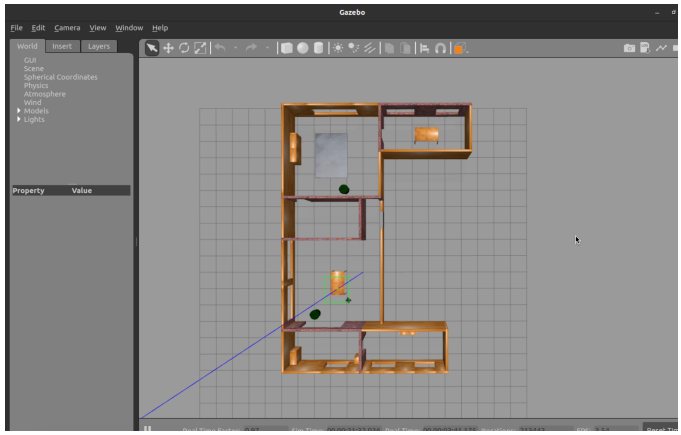

Fig. 3. The SLAM map on Gazebo

3. The effectiveness of object detection and tracking was the main focus of our third experiment. The objective assigned to the camera-equipped Turtlebot 3 was to locate and follow a human model around the Gazebo environment. This was accomplished by using the previously made SLAM map to launch a world file containing the Turtlebot3 and the human model. The crucial phase was for a node to subscribe to the image topic of the turtlebot3 camera and use cv_bridge to convert ROS image messages to OpenCV format. For object detection, we used YOLOv2, with parameters adjusted for best results. Based on bounding box data, the Turtlebot 3 successfully tracked the human model, demonstrating a high detection accuracy in the process. This experiment illustrated the system's potential by highlighting the benefits of combining robotic navigation with sophisticated object detection algorithms.

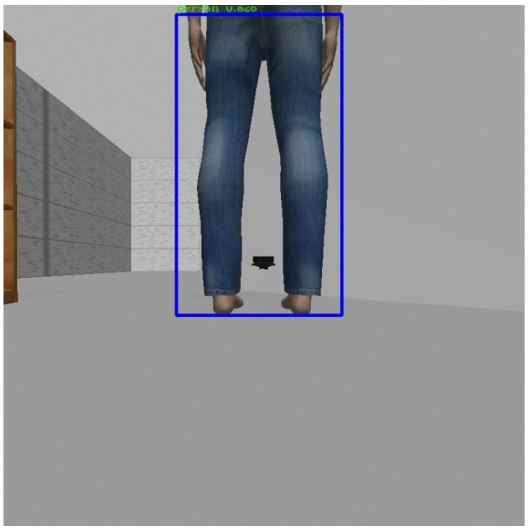

Fig.4. Robot detecting the human model

F. Discussion

The experiments conducted as part of this project provide substantial insights into the capabilities and limitations of our integrated robotic system in a simulated domestic environment. High accuracy was demonstrated by the SLAM process, which proved useful in producing detailed environmental maps. The reliance on refined parameters, however, points to the necessity of adaptive algorithms that can automatically adapt to changing environmental conditions. Autonomous navigation showed promising results, especially in obstacle avoidance and path optimisation. The success of the Turtlebot3 in navigating and using path-finding algorithms in cluttered spaces is encouraging for real-world applications. Future work could explore more complex environments and introduce elements like varying floor textures and dynamic human behaviour. The object detection and tracking experiment highlighted the potential and importance of integrating advanced vision algorithms with robotic navigation systems. The high accuracy in human model tracking opens avenues for applications in assistive robotics. Future improvements could focus on enhancing detection capabilities in diverse lighting conditions and incorporating more sophisticated human behavior models. Overall, the project highlights the potential of ROS and Gazebo in developing complex robotic systems that are very useful to solve real-life problems. The integration of SLAM, navigation, and object detection in a cohesive framework has been successfully demonstrated. Future research should aim to transfer these capabilities from simulation to real-world applications, addressing the challenges of dynamic environments and human-robot interaction.

G. Conclusion

The project successfully combined autonomous navigation, object tracking, and SLAM, significantly improving our knowledge of robotic perception in virtual environments. The results highlight the potential of Gazebo and ROS in creating complex robotic systems that can perform challenging perception tasks. The successful application of these technologies offers a guide for upcoming robotic perception studies. Subsequent research endeavours may concentrate on enhancing the system's computational efficacy in difficult scenarios and assignments.

Further developing the integration of AI-based navigation strategies and more sophisticated object recognition algorithms could improve the perception and decision-making abilities of robotic systems.

References

1. *Bradski, G. (2000). The OpenCV Library. Dr. Dobb's Journal of Software Tools.*
2. *Koenig, N., Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. Intelligent Robots and Systems*
3. *Viola, P., & Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features*

Github Repo link: https://github.com/momo285/group15