

Augmented Reality Based SLAM



By

Izma Naveed

(Registration No: 00000327564)

Department of Robotics and Artificial Intelligence

School of Mechanical and Manufacturing Engineering

National University of Sciences & Technology (NUST)

Islamabad, Pakistan

(2024)

Augmented Reality Based SLAM



By

Izma Naveed

(Registration No: 00000327564)

A thesis submitted to the National University of Sciences and Technology, Islamabad,

in partial fulfillment of the requirements for the degree of

Master of Science in

Robotics and Intelligent Machine Engineering

Supervisor: Dr. Khawaja Fahad Iqbal

School of Mechanical and Manufacturing Engineering

National University of Sciences & Technology (NUST)

Islamabad, Pakistan

(2024)

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS Thesis written by Mr / Ms Izma Naveed
_____ (Registration No. 00000327564), of SMME (School/College/Institute) has
been vetted by undersigned, found complete in all respects as per NUST Statutes/
Regulations/ Masters Policy, is free of plagiarism, errors, and mistakes and is accepted
as partial fulfillment for award of Masters degree. It is further certified that necessary
amendments as point out by GEC members and evaluators of the scholar have also been
incorporated in the said thesis.


Signature: 

Name of Supervisor: Khawaja Fahad Iqbal

Date: 21-08-2024

Signature (HOD): 

Date: 21-08-2024

Signature (Dean/ Principal): 


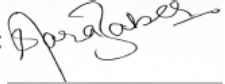
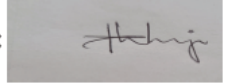
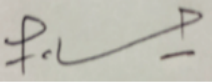
Date: 21-08-2024



National University of Sciences & Technology (NUST)
MASTER'S THESIS WORK

We hereby recommend that the dissertation prepared under our supervision by: Izma Naveed (00000327564)
Titled: Augmented Reality Based SLAM be accepted in partial fulfillment of the requirements for the award of MS in Robotics & Intelligent Machine Engineering degree.

Examination Committee Members

- | | | |
|---------------------------------|--|--|
| 1. | Name: Yasar Ayaz | Signature:  |
| 2. | Name: Sara Baber Sial | Signature:  |
| 3. | Name: Khudaija Ashfaq Ahmad | Signature:  |
| Supervisor: Khawaja Fahad Iqbal | Signature:  | |
| | Date: <u>19 - Aug - 2024</u> | |


Head of Department

19 - Aug - 2024

Date

COUNTERSIGNED

19 - Aug - 2024

Date



Dean/Principal

CERTIFICATE OF APPROVAL

This is to certify that the research work presented in this thesis, entitled “ Augmented Reality Based SLAM ” was conducted by Mr./Ms Izma Naveed under the supervision of Dr. Khawaja Fahad Iqbal .

No part of this thesis has been submitted anywhere else for any other degree. This thesis is submitted to the Department of Robotics and Artificial Intelligence in partial fulfillment of the requirements for the degree of Master of Science in Field of Robotics and Intelligent Machine Engineering .

Department of Robotics and Artificial Intelligence National University of Sciences and Technology, Islamabad.

Student Name: Izma Naveed

Signature: 

Examination Committee:

a) External Examiner 1: Name

Signature: NIL

(Designation & Office Address)

b) External Examiner 2: Name

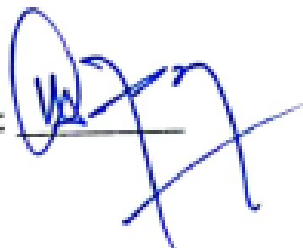
Signature: NIL

(Designation & Office Address)

Supervisor Name: Dr. Khawaja Fahad Iqbal

Signature: 

Name of Dean/HOD: Dr. Kunwar Faraz Ahmed Khan

Signature: 

AUTHOR'S DECLARATION

I Izma Naveed hereby state that my MS thesis titled "Augmented Reality Based SLAM" is my own work and has not been submitted previously by me for taking any degree from National University of Sciences and Technology, Islamabad or anywhere else in the country/ world.

At any time if my statement is found to be incorrect even after I graduate, the university has the right to withdraw my MS degree.

Name of Student: Izma Naveed

Date: 21-08-2024



PLAGIARISM UNDERTAKING

I solemnly declare that research work presented in the thesis titled “Augmented Reality Based SLAM” is solely my research work with no significant contribution from any other person. Small contribution/ help wherever taken has been duly acknowledged and that complete thesis has been written by me.

I understand the zero tolerance policy of the HEC and National University of Sciences and Technology (NUST), Islamabad towards plagiarism. Therefore, I as an author of the above titled thesis declare that no portion of my thesis has been plagiarized and any material used as reference is properly referred/cited.

I undertake that if I am found guilty of any formal plagiarism in the above titled thesis even after award of MS degree, the University reserves the rights to withdraw/revoke my MS degree and that HEC and NUST, Islamabad has the right to publish my name on the HEC/University website on which names of students are placed who submitted plagiarized thesis.



Student Signature: _____

Name: Izma Naveed

DEDICATION

To my parents, for their unwavering support, endless encouragement, and boundless love. To my grandparents for always believing in me and being a steady source of love and support.

ACKNOWLEDGEMENTS

I would like to extend my heartfelt gratitude towards my colleague Saad Jameel, for his invaluable support and continuous encouragement. I am also grateful to the rest of my peers who have been a source of support and camaraderie. Your constructive feedback and shared enthusiasm are appreciated.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	VIII
TABLE OF CONTENTS	IX
LIST OF TABLES	XI
LIST OF FIGURES	XII
LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS	XIV
ABSTRACT	XV
CHAPTER 1. INTRODUCTION	1
1.1 Background and Motivation	1
1.2 Human-Robot Interaction and Augmented Reality	3
1.3 Gap-Based Navigation Techniques and Their Limitations	5
1.4 Proposed AR-Based Hector SLAM Approach	7
1.5 Problem Statement and Research Objectives	8
1.5.1 Problem Statement	8
1.5.2 Research Objectives	8
1.6 Summary	9
CHAPTER 2. LITERATURE REVIEW	10
2.1 Colocalization Methods	10
2.1.1 Marker-Based Colocalization	11
2.1.2 Markerless Colocalization	12
2.2 AR-based Motion Planning	14
2.2.1 Graph-based Algorithms	15
2.2.2 Sampling-based Algorithms	15
2.2.3 Optimization-based Algorithms	15
2.2.4 Potential Field Methods	16
2.2.5 Gap-based Algorithms	16
2.3 Summary	18
CHAPTER 3. METHODOLOGY	19
3.1 AR-HMD Colocalization with Mobile Robot	19
3.2 AR-ROS Communication Framework	23
3.3 Hector SLAM with EKF sensor fusion	25
3.3.1 Global Planner	26
3.3.2 Local Planner	27
3.4 Summary	28
CHAPTER 4. EXPERIMENTAL RESULTS	29
4.1 Experimental Setup	29

4.1.1	Scenario 1	30
4.1.2	Scenario 2	31
4.1.3	Scenario 3	32
4.1.4	Evaluation Metrics	33
4.2	Summary	34
CHAPTER 5. DISCUSSION		36
5.1	Jerk	36
5.1.1	Scenario 1	36
5.1.2	Scenario 2	38
5.1.1	Scenario 3	39
5.2	Stress	41
5.2.1	Scenario 1	41
5.2.2	Scenario 2	41
5.2.3	Scenario 3	42
5.3	Quantitative Performance Comparison	42
5.3.1	Scenario 1	42
5.3.2	Scenario 2	44
5.3.3	Scenario 3	45
5.3.4	Overall average improvement	47
5.4	Summary	49
CHAPTER 6. CONCLUSION		50
6.1	Contributions	51
6.2	Future Work	52
6.3	Research Publication	52
CHAPTER 7. REFERENCES		53

LIST OF TABLES

	Page No.
Table 1: Comparison of AR-based Hector SLAM and AG Navigation	37
Table 2: Scenario 1 t-test.....	44
Table 3: Scenario 2 t-test.....	46
Table 4: Scenario 3 t-test.....	48

LIST OF FIGURES

	Page No.
Figure 1.1: Warehouse robot.....	1
Figure 1.2: Surgical robot	2
Figure 1.3: Milgram and Kishino’s Reality-Virtuality Continuum	2
Figure 1.4: Using Magic Leap 2 AR headset for innovative problem-thinking	4
Figure 1.5: Marker-based AR	5
Figure 2.1: AR Colocalization Methods	10
Figure 2.2: (a) Apriltag (b) ArUco (c) ARTAG	11
Figure 2.3: Motion planning methods.....	17
Figure 3.1: System Architecture	19
Figure 3.2: Frame Representation (a) Robot (b) ML1 (c) Sim.....	20
Figure 3.3: Transformation between ML1 and Sim.....	21
Figure 3.4: Final Transformation.....	22
Figure 3.5: Surface Detection	24
Figure 3.6: AR view for AR-based Hector SLAM (a) Detect Robot (b) Set goal.....	25
Figure 4.1: Simulation environment view for the three scenarios	29
Figure 4.2: (a) Scenario 1 real-world set up (b) Path from initial to goal point.	30
Figure 4.3: Scenario 1 Navigation	30
Figure 4.4: Scenario 2 real-world set up.	31
Figure 4.5: Scenario 2 navigation	32
Figure 4.6: Scenario 3 real-world set up.	32
Figure 4.7: Scenario 3 navigation	33
Figure 5.1: Velocity, acceleration, jerk, and stress profiles for AR-based Hector SLAM (green) and AG navigation (red) for Scenario 1	38
Figure 5.2: Velocity, acceleration, jerk, and stress profiles for AR-based Hector SLAM (green) and AG navigation (red) for Scenario 2	39
Figure 5.3: Velocity, acceleration, jerk, and stress profiles for AR-based Hector SLAM (green) and AG navigation (red) for Scenario 3	40

Figure 5.4: Scenario 1 mean decrease in jerk and stress.....	43
Figure 5.5: Scenario 2 mean decrease in jerk and stress.....	45
Figure 5.6: Scenario 3 mean decrease in jerk and stress.....	47
Figure 5.7: AR-based Hector SLAM average decrease in jerk and stress.....	49

LIST OF SYMBOLS, ABBREVIATIONS AND ACRONYMS

SLAM	Simultaneous Localization and Mapping
EKF	Extended Kalman Filter
AMCL	Adaptive Monte Carlo Localization
AG	Admissible Gap
HRI	Human-Robot Interaction
AR	Augmented Reality
AR-HMD	Augmented Reality Head Mounted Display
S1	Scenario 1
S2	Scenario 2
S3	Scenario 3

ABSTRACT

Motion planning is crucial for helping autonomous robots navigate complex environments efficiently. Recently, Augmented Reality (AR) has been introduced to improve Human-Robot Interaction (HRI) in mobile robot motion planning. However, AR gap-based reactive control systems often suffer from issues like sensor noise and inaccuracies, leading to higher levels of jerk and stress. On the other hand, Simultaneous Localization and Mapping (SLAM) provides a global understanding of the environment, ensuring robust navigation even in dynamic or unfamiliar areas. In this paper, we propose an AR-based Hector Simultaneous Localization and Mapping (SLAM) method for intuitive indoor mobile robot navigation that reduces jerk and stress. Our approach uses AR to set navigation goals and provide visual markers for the user, while SLAM ensures accurate real-time mapping for precise navigation and obstacle avoidance. For path planning, the robot uses Dijkstra's algorithm for global planning and Trajectory Rollout for local planning. We tested the effectiveness of our AR-based Hector SLAM in three different scenarios and compared the results with an admissible gap-based navigation algorithm. Experimental results showed that our method improved jerk and stress by 11.63% and 11.39% respectively, leading to smoother and safer trajectories.

Keywords: Motion planning, mobile robot navigation, Augmented Reality, Simultaneous Localization and Mapping (SLAM), Human-Robot Interaction (HRI)

CHAPTER 1. INTRODUCTION

In recent years, the field of robotics has seen significant advancements, particularly in autonomous mobile robots. As these robots are increasingly deployed in environments shared with humans, the need for effective human-robot interaction (HRI) becomes crucial. Augmented Reality (AR) has emerged as a promising technology to bridge the gap between human operators and robotic systems, providing intuitive interfaces and real-time data visualization. This chapter introduces the motivation behind this research, outlines the key problems addressed, and presents the research objectives. It also discusses the limitations of existing navigation techniques and proposes a novel AR-based Simultaneous Localization and Mapping (SLAM) approach to enhance robot navigation in dynamic environments.

1.1 Background and Motivation

Environments where robots and humans work in a shared space are increasing rapidly, making human-robot interaction (HRI) a critical area of research and development. Robots are increasingly deployed in sectors such as e-commerce warehouses [1], transportation [2], [3] and medical care [4], [5].



Figure 1.1: Warehouse robot [3]



Figure 1.2: Surgical robot [5]

Ensuring seamless communication and collaboration between robots and humans is critical. Autonomous mobile robots must navigate complex and dynamic surroundings, generating feasible trajectories that guarantee safe and efficient movement. Augmented Reality (AR) has emerged as a state-of-the-art technology that can significantly enhance HRI. By overlaying digital information in the real world, AR provides an intuitive and immersive interface for interacting with robots [6].

This capability is particularly beneficial for tasks that require precise coordination and real-time feedback, such as object recognition or robot navigation. The concept of AR, situated within Milgram and Kishino's Reality-Virtuality Continuum [7], bridges the gap between the real and virtual worlds, offering innovative methods for improving HRI.

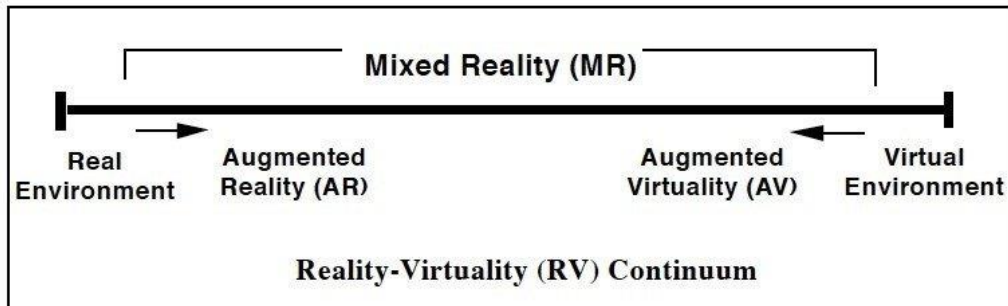


Figure 1.3: Milgram and Kishino's Reality-Virtuality Continuum [7]

By leveraging AR, robots can receive real-time instructions and data for performing tasks in a collaborative augmented space alongside humans, enhancing overall efficiency and safety. AR-based robotics has various applications, such as in collaborative tasks [8] and object recognition [9]. However, one of the most significant challenges is effective motion planning for autonomous mobile robots.

Model-based, learning-based, and sensor-based motion planning algorithms have strengths and weaknesses. Early gap-based navigation methods, inspired by the Bug algorithm [10], have evolved to improve trajectory smoothness and obstacle avoidance. However, these methods often assume specific robot shapes and kinematic configurations, which restrict their applicability. The Admissible Gap-Based (AG) Navigation algorithm [11] introduced by Mujahed et al. addresses some of these limitations by considering the exact shape and kinematics of the robot. However, AG Navigation's reliance on real-time analysis without environmental retention can lead to unstable and jerky motions, posing safety risks in dynamic settings. This highlights the need for more robust navigation methods that can adapt to changing environments and ensure smooth operation.

SLAM (Simultaneous Localization and Mapping) presents a promising solution to the drawbacks of AG navigation. It continuously builds and updates a map of the environment while localizing the robot within that map, enabling real-time adaptability to environmental changes. We propose an AR-based SLAM approach to enhance indoor mobile robot navigation, leveraging the strengths of both AR and SLAM to overcome the limitations of traditional gap-based navigation methods.

1.2 Human-Robot Interaction and Augmented Reality

The concept of AR comes from Milgram and Kishino's Reality-Virtuality Continuum [7], which defines a spectrum from the real world to a fully virtual world. AR falls in between this spectrum, allowing users to view augmented data and information on top of the real-world environment [12].

According to Goodrich et al. [13], the interaction between robots and humans falls into one of four categories: speech and natural language, visual displays, haptics and physical

interactions, and gestures. AR interfaces fall within the second category, offering a new modality as a means of enhancing HRI in mobile robot navigation. By leveraging AR, robots can receive real-time instructions and data for performing tasks in a collaborative augmented space alongside a human.



Figure 1.4: Using Magic Leap 2 AR headset for innovative problem-thinking [12]

The integration of robotics and AR is an active research area, offering new technological innovations [14]. It presents new methods for enhancing HRI so that robots and humans can work collaboratively [15], [16]. AR enables robots to receive real-time instructions and feedback, enhancing their autonomy and adaptability in dynamic environments.

Effective colocalization strategies are crucial for aligning the coordinate frames of AR devices, such as head-mounted displays (HMDs), with mobile robots. Traditional methods

often rely on fiducial markers [17] , [18] or natural landmarks [19] for spatial alignment, reducing the flexibility of the system. We propose a markerless approach by using mathematical transforms to establish precise relationships between AR-HMDs and robots, improving robustness in HRI applications. The integration of AR with robotics enhances HRI by providing user-friendly control interfaces [20], real-time spatial awareness [21], and visualization [22].



Figure 1.5: Marker-based AR [18]

1.3 Gap-Based Navigation Techniques and Their Limitations

Gap-based navigation originated with early methods like the Bug algorithm [10], which focused on reactive obstacle avoidance by navigating through open spaces (gaps) between obstacles. These methods provided initial solutions for robots to move in complex environments without detailed map information. The Nearness-Diagram (ND) [23] approach improved upon early techniques by optimizing motion paths through smoother trajectory planning and better obstacle avoidance. However, these methods often struggled with oscillations [24] and deviations, especially in complex environments or with dynamic obstacles. Enhancements such as tangential navigation [25] and the Follow the Gap Method (FGM) [26] integrated with the Dynamic Window Approach (DWA) [27] further refined gap-based navigation [28]. These approaches aimed to improve trajectory smoothness,

safety, and navigation efficiency by dynamically adjusting robot velocities and accelerations based on real-time sensor data.

Traditional gap-based techniques have certain limitations as listed below:

- **Assumption of Robot Shape and Dynamics:** Many gap-based methods assume a specific robot shape (often disc-shaped) and idealized kinematic properties. This assumption limits their applicability to robots with different physical configurations or non-holonomic constraints.
- **Real-Time Adaptability:** Traditional gap-based methods focus on real-time analysis of immediate surroundings to navigate through admissible gaps. They may struggle in dynamic environments where obstacles appear suddenly or where the environment changes over time, leading to suboptimal path planning and increased collision risks.
- **Environmental Mapping and Memory:** These methods typically do not build or update a comprehensive map of the environment. They rely solely on real-time sensor data to navigate, which can limit long-term spatial awareness and planning capabilities.

To overcome the drawbacks of traditional gap-based techniques, Mujahed et al. [11] developed the Admissible Gap-Based (AG) Navigation algorithm. AG Navigation considers the exact shape and kinematics of the robot, allowing for more precise navigation through narrow passages and complex environments. It integrates robot-specific parameters into its path-planning process, enhancing adaptability and safety in dynamic scenarios.

While AG Navigation has addressed several limitations of other gap-based algorithms, it has its own drawbacks. AG Navigation prioritizes real-time analysis to identify admissible gaps for navigation. It cannot retain information about the surroundings or construct a map and is computationally expensive [29]. Consequently, without comprehensive environmental retention, AG Navigation may induce higher levels of jerk and stress during robot motion. These factors contribute to unstable and jerky robot motion. AG Navigation also struggles with real-time adaptability. In dynamic environments where obstacles can

suddenly appear or change, AG Navigation's real-time analysis may not be sufficient to respond effectively, leading to decreased performance and increased collision risk.

Chacón-Quesada et al. [30] developed a smart wheelchair using Microsoft's HoloLens [31] and the AG Navigation method for motion planning. The wheelchair avoids obstacles by finding and navigating through gaps in the environment. However, as explained previously, AG Navigation depends on real-time gap detection, nor does it keep track of a global map. Additionally, it gives higher levels of jerk and stress, which can affect sensor accuracy and lead to incorrect navigation and obstacle avoidance.

These limitations underscore the need for more robust methods that can dynamically update their understanding of the environment and adapt accordingly.

1.4 Proposed AR-Based Hector SLAM Approach

In contrast, Simultaneous Localization and Mapping (SLAM) offers a robust solution for real-time environment mapping, localization, and obstacle avoidance. SLAM continuously builds and updates a map of the environment while simultaneously determining the robot's location within that map. This allows SLAM to dynamically adapt to changes in the environment, making it highly effective in both static and dynamic settings. By leveraging sensor data to create a detailed and accurate representation of the surroundings, SLAM enables robots to plan paths and avoid obstacles efficiently.

In this thesis, we propose an augmented reality (AR)-based Hector SLAM method for smooth motion planning of indoor mobile robots. Hector SLAM is a variant of the SLAM algorithm that is particularly well-suited for environments with limited computational resources or where rapid mapping is required. The AR system is utilized primarily for setting a goal location and providing spatial cues to the user, creating an intuitive and immersive interface for navigation tasks. Our proposed method leverages Dijkstra's algorithm for global path planning and Trajectory Rollout with Dynamic Window Approach (DWA) for local path planning. Dijkstra's algorithm is well-known for finding the shortest path in a graph with non-negative edge weights, making it reliable for global navigation. Trajectory Rollout with DWA provides a robust solution for local path

planning, ensuring that the robot can navigate around obstacles and adapt to dynamic changes in the environment. This combination of algorithms ensures both optimal path planning and real-time adaptability, addressing the limitations of AG Navigation.

1.5 Problem Statement and Research Objectives

1.5.1 Problem Statement

AG navigation is the current state-of-the-art method for AR-based indoor mobile robot navigation. However, it focuses on real-time analysis to identify admissible gaps for navigation, which means it does not retain information about the environment or build a map. This lack of environmental retention can result in higher values of jerk and stress, leading to unstable and jerky motion for mobile robots.

1.5.2 Research Objectives

- **Use Markerless AR Colocalization:** Create a robust method to determine the relative transformations between an AR head-mounted display (AR-HMD) and a mobile robot, eliminating the need for fiducial markers or natural landmarks.
- **Implementation of proposed AR-based Hector SLAM:** Implement the proposed AR-based Hector SLAM method on a mobile robot, leveraging high-frequency LIDAR data.
- **Enhance Environmental Adaptability:** The system should be capable of retaining environmental information over time, ensuring robust and adaptable navigation in dynamic indoor environments.
- **Reduce Stress and Jerk in Navigation:** Demonstrate that the AR-based Hector SLAM method significantly reduces stress and jerk in robot navigation compared to the Admissible Gap-Based (AG) navigation method.
- **Validate Through Experiments:** Experimental evaluations in both simulated and real-world scenarios to validate the effectiveness of the proposed method.

1.6 Summary

In summary, Chapter 1 has provided a comprehensive overview of the background and motivation behind this research. We highlight the importance of efficient and reliable human-robot interaction (HRI) and the role of augmented reality (AR) in enhancing this interaction. We observed the limitations of existing gap-based navigation methods and proposed an AR-based Hector SLAM approach for smooth motion planning of indoor mobile robots.

CHAPTER 2. LITERATURE REVIEW

This chapter highlights existing studies and advancements in the fields of AR, SLAM, and robot navigation. We will examine various colocalization methods, motion planning algorithms, and the integration of AR with robotics. By critically analyzing the current techniques and state-of-the-art algorithms, we aim to identify gaps in the existing literature.

2.1 Colocalization Methods

Colocalization of AR-HMDs in robotics is the process of accurately aligning the spatial coordinates of both systems. This allows robots and human operators using AR-HMDs to share a common understanding of their environment, facilitating more effective collaboration and interaction [6]. Colocalization is crucial for applications such as industrial automation, teleoperation, collaborative assembly, and remote maintenance, among others. Figure 2.1 shows common AR-colocalization methods.

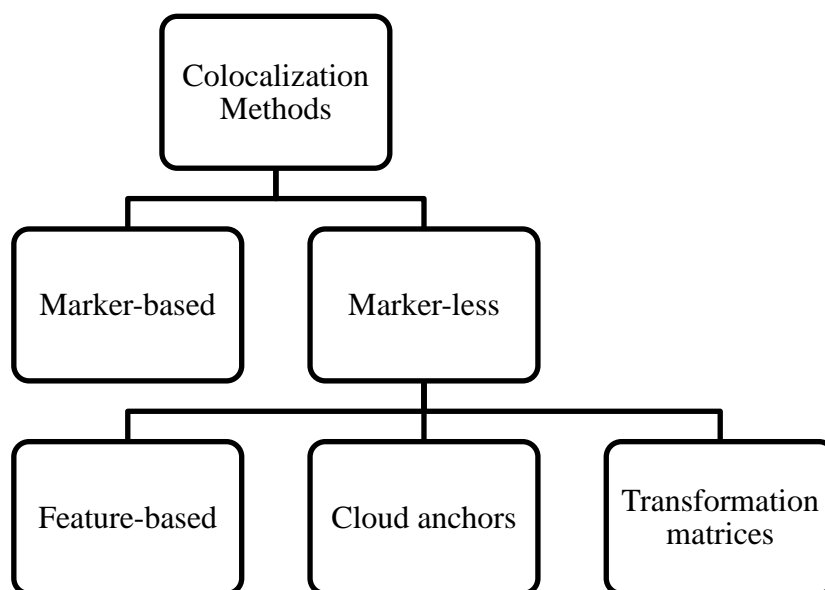


Figure 2.1: AR Colocalization Methods

2.1.1 Marker-Based Colocalization

Marker-based colocalization methods rely on visual markers strategically placed within the environment to serve as reference points for aligning the coordinate systems of AR-HMDs and robots. These markers can be simple patterns, such as QR codes, or more sophisticated fiducial markers like AprilTag [32], ArUco [33], and ARTAG [34]. The process begins with both systems detecting these markers within their respective fields of view. Each marker's unique pattern allows these systems to identify and determine the precise position and orientation of the markers relative to their cameras.

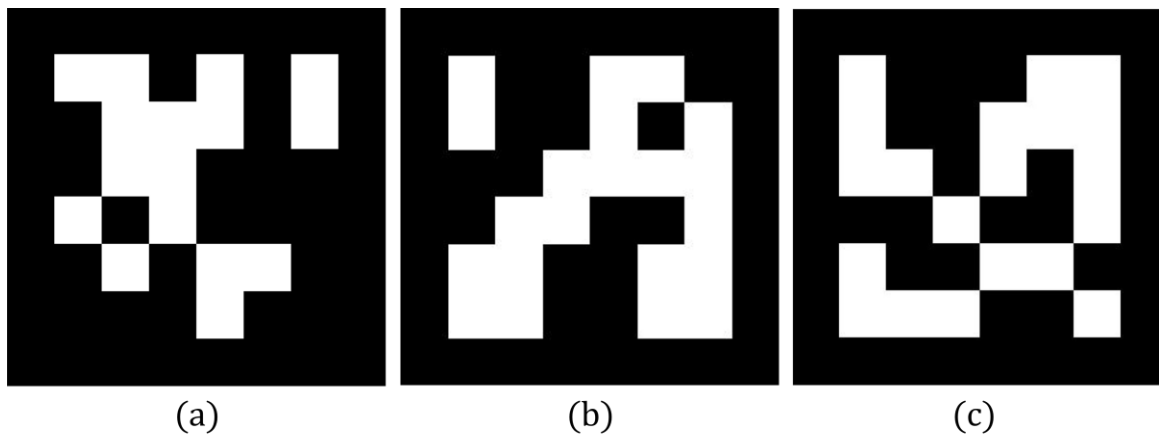


Figure 2.2: (a) Apriltag [32] (b) ArUco [33] (c) ARTAG [34]

Once the markers are detected, the systems calculate the transformation matrices required to map the marker positions to their own coordinate frames. By sharing the positions of these common markers, the AR-HMD and the robot can align their coordinate systems. This method is highly accurate because the markers provide well-defined points for calibration. However, it requires physical modifications to the environment, which may not be feasible in all scenarios. Additionally, the presence of visual clutter or dynamic changes in the environment can impede the detection and accuracy of marker-based systems. Marker-based AR has been used by many robotics systems such as UAVs [35] and manipulators [36].

2.1.2 *Markerless Colocalization*

Markerless colocalization methods do not rely on physical markers in the environment. Instead, they use various techniques to align the coordinate systems of the AR-HMD and the robot. These techniques include feature-based colocalization, location-based colocalization, cloud anchors, and transformation matrices.

2.1.2.1 Feature-Based Colocalization

Feature-based colocalization relies on detecting and tracking features within the environment, such as corners, edges, or textures, to build a shared map and align the coordinate systems. The process begins with the AR-HMD and the robot detecting and extracting features from their camera images. These features are then tracked across successive frames to estimate movement and build a map of the environment. The systems use the shared map to align their coordinate systems.

The main advantages of feature-based colocalization are that it does not require modifications to the environment and is capable of operating in dynamic and unstructured environments. However, it is computationally intensive, requiring significant processing power, and its accuracy can be affected by lighting conditions and feature-poor environments.

2.1.2.2 Cloud Anchors

Cloud anchors use cloud-based services to store and retrieve anchor points in the environment, facilitating colocalization by providing a common reference frame [37]. The process begins with the AR-HMD and the robot creating and uploading anchor points to a cloud service. The systems can then retrieve these anchors from the cloud to align their coordinate systems.

The advantages of cloud anchors are that they enable persistent and shared anchors across different sessions and devices, and they do not require physical markers. However, this method is dependent on network connectivity and cloud service reliability, and there are potential privacy and security concerns with data transmission.

Zolotas et. al [38] and Chacón-Quesada et. al [30] used the Hololens to place three virtual objects as spatial anchors in the environment, while [39] used landmark detection. Apple's ARKit [40], Microsoft's Azure Spatial Anchors (ASA) [41], and Google's ARCore [42] are popular cloud-based services that allow you to create spatial anchors.

2.1.2.3 Transformation Matrices

Markerless colocalization using transformation matrices involves aligning the coordinate systems of AR-HMDs and robots without relying on physical markers or feature detection. Instead, this method uses an initial set of reference points to establish a common frame of reference, facilitating accurate and seamless coordination between the two systems. The process begins by acquiring the initial position of the robot as seen through the AR-HMD. These initial points are then transmitted to the robot. Upon receiving these points, the robot performs frame transformation to align its local coordinate system with that of the AR-HMD.

Once the initial alignment is completed, the AR-HMD can send goal positions or commands within its own coordinate system to the robot. The robot uses the established transformation matrix to interpret these commands correctly, ensuring that it follows the spatial coordinates accurately. This method allows for dynamic interaction and control, as the robot can adjust its movements in real-time based on inputs from the AR-HMD.

The primary advantage of this markerless method is that it does not require any modifications to the environment, making it highly flexible and suitable for various applications, including those in dynamic or unstructured settings. It also simplifies the setup process by eliminating the need for physical markers or complex feature detection algorithms. However, the initial calibration step is critical, as any errors in the initial points or the transformation matrix can lead to misalignment and inaccuracies in subsequent operations.

Each colocalization method for AR-HMDs and robots has its unique advantages and challenges. Marker-based methods offer high accuracy but require environmental modifications. Feature-based colocalization provides flexibility but demands significant

computational resources. Cloud anchors enable persistent and shared reference points but rely on network connectivity and cloud services. Among these methods, markerless colocalization using transformation matrices stands out as the most cost-effective and robust method. It eliminates the need for physical markers and simplifies the setup process, making it suitable for a wide range of applications, including those in dynamic and unstructured environments. By leveraging initial calibration and frame transformation, this method ensures accurate and seamless colocalization between AR-HMDs and robots, offering a practical and efficient solution for modern AR-robot integration.

2.2 AR-based Motion Planning

Motion planning for mobile robots is a critical aspect of robotics. It enables the robot to navigate from an initial to a goal position while avoiding obstacles. The fundamental goal of motion planning is to find a feasible path for a robot to follow within its operational environment. This involves understanding the robot's dynamics, the nature of the environment, and the potential obstacles that may hinder its movement. The complexity of motion planning arises from the need to balance computational efficiency with the accuracy and reliability of the planned paths. As a result, multiple types of motion planning algorithms have been developed, each with its own strengths and weaknesses.

Despite the rapid advancements in motion planning algorithms, AR-based motion planning remains an area with limited comprehensive research. Much of the current research in AR and mobile robot navigation focuses on leveraging AR to visualize data and information. For instance, Walker et al. [43] have utilized AR to visualize a robot's intended motion, providing operators with an intuitive understanding of the robot's planned path and actions. This visualization aids in decision-making and enhances the safety and efficiency of robot operation. Furthermore, Kastner et al. [44] evaluated the use of Microsoft HoloLens for visualizing navigation data of mobile robots in 3D. This approach allows users to perceive the robot's environment and navigation data in a more immersive and interactive manner, improving situational awareness and facilitating better control and monitoring of the robot's movements. [45] explored the use of AR in robot teleoperation to improve human-robot interaction.

2.2.1 *Graph-based Algorithms*

One of the earliest and most intuitive approaches to motion planning is through graph-based algorithms. These algorithms represent the environment as a graph where nodes correspond to discrete positions and edges represent possible paths between these positions. The most well-known graph-based algorithms include Dijkstra's algorithm [46], A* [47], D* [48], and vector field histogram* (VFH*) [49] algorithms. Mostafa et.al [50] used Dijkstra's algorithm to find the shortest path in a graph with non-negative edge weights.

2.2.2 *Sampling-based Algorithms*

As environments and robotic tasks became more complex, the limitations of graph-based algorithms in high-dimensional spaces led to the development of sampling-based algorithms. These methods, such as Rapidly-exploring Random Tree (RRT) [51], do not explicitly construct a graph of the entire environment. Instead, they randomly sample the space to build a roadmap or tree that captures feasible paths. RRT starts from the initial position and incrementally builds a tree towards the goal, making it particularly effective in real-time applications. Both methods are highly effective in dealing with complex, high-dimensional environments where traditional graph-based methods would be computationally prohibitive. Christos et.al [52] used an aerial robot and AR for structural inspections, using RRT* for planning.

2.2.3 *Optimization-based Algorithms*

Another approach focuses on optimizing certain criteria along the path, such as minimizing travel time. These optimization-based algorithms include methods like Model Predictive Control (MPC) [53], [54]. These techniques use mathematical optimization to refine paths iteratively, considering the robot's dynamic constraints and environmental factors. Optimization-based algorithms are powerful for generating smooth and feasible trajectories that respect the physical limitations of the robot, though they often require significant computational resources and accurate models of the robot and environment.

2.2.4 *Potential Field Methods*

Potential field methods [55] represent another category of motion planning algorithms, inspired by physical systems where robots are treated as particles moving under the influence of artificial potential fields. These fields are constructed such that the goal exerts an attractive force, while obstacles exert repulsive forces. The robot navigates by following the gradient of the resultant field. This approach is intuitive and computationally efficient, making it suitable for real-time applications. However, potential field methods can suffer from issues such as local minima, where the robot gets stuck in areas where the forces cancel each other out, preventing it from reaching the goal.

2.2.5 *Gap-based Algorithms*

Gap-based algorithms detect gaps between obstacles, creating a path from the start to the goal point. Artificial potential field (APF) [56] and Follow the Gap Method (FGM) [26] are popular examples of gap-based algorithms. This detection can be achieved through various sensing modalities such as laser range finders, sonar, or stereo vision systems, which provide information about the distances to nearby obstacles. The robot then identifies gaps—regions of the environment where the distance to obstacles on either side is sufficiently large to permit safe passage. Recently, Mujahed et. al [11] introduced the concept of Admissible gap-based navigation for mobile robots.

The term "admissible" refers to gaps that meet predefined conditions ensuring the safety, efficiency, and effectiveness of navigation. In admissible gap-based navigation, the robot continuously evaluates potential gaps based on factors such as width, distance to the goal, and the presence of obstacles. This evaluation helps the robot to prioritize gaps that are nearer to the goal.

Chacón-Quesada et al. introduced an innovative concept that integrates dynamic signifiers in AR for robot navigation [30]. This approach combines AG navigation with AR to create a state-of-the-art method for reactive collision avoidance.

Although Admissible Gap (AG) Navigation has addressed several limitations of traditional gap-based algorithms, it has its own drawbacks. One of the primary limitations of AG

Navigation is its lack of environmental retention; it does not build or maintain a map of the environment. Instead, AG Navigation focuses on real-time analysis to identify admissible gaps for navigation. While this approach allows for quick decision-making and reactive collision avoidance, it can result in higher values of jerk and stress, leading to unstable and jerky motion for mobile robots. This can be particularly problematic in scenarios requiring smooth and consistent movement. Moreover, AG Navigation struggles with real-time adaptability. In dynamic environments where obstacles can suddenly appear or change, AG Navigation's real-time analysis may not be sufficient to respond effectively. This can result in increased collision risk with high jerks.

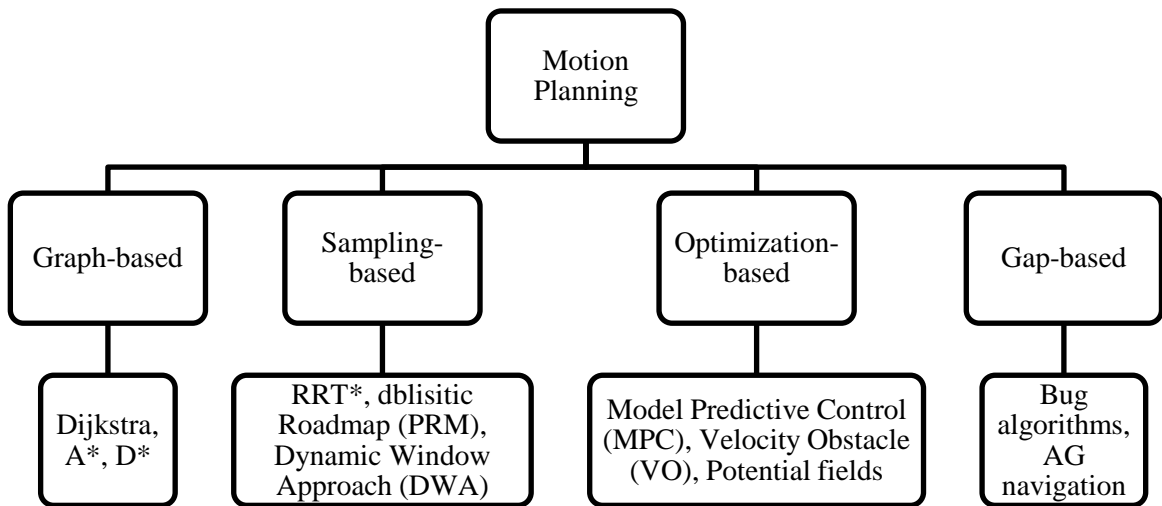


Figure 2.3: Motion planning methods

In contrast, Simultaneous Localization and Mapping (SLAM) offers a robust solution for real-time environment mapping, localization, and obstacle avoidance. SLAM continuously builds and updates a map of the environment while simultaneously determining the robot's location within that map. SLAM can dynamically adapt to changes in the environment, making it highly effective in both static and dynamic settings. SLAM algorithms use sensor data to create a detailed and accurate representation of the surroundings, which the robot can use to plan its path and avoid obstacles efficiently.

2.3 Summary

Chapter 2 gives a detailed review of existing colocalization methods and AR-based motion planning. We examined both marker-based and markerless colocalization techniques, highlighting their advantages and limitations. In AR-based motion planning, we explored various approaches, including graph-based, sampling-based, optimization-based, potential field methods, and gap-based algorithms. This comprehensive review of existing research helped us understand the gaps in the current work and formed the basis for developing our proposed AR-based Hector SLAM approach.

CHAPTER 3. METHODOLOGY

The complete system architecture is shown in Figure 3.1. There are three major modules of our system. The first module is the AR-headset module, which is associated with the user. The surface detection node helps detect different surfaces when the user points with the controller. Once an initial or goal position is set, the coordinates are sent to the robot via ROSBridge [57]. The robot uses SLAM to map the environment in real-time and navigate to the goal position. A detailed explanation is given in the following sections.

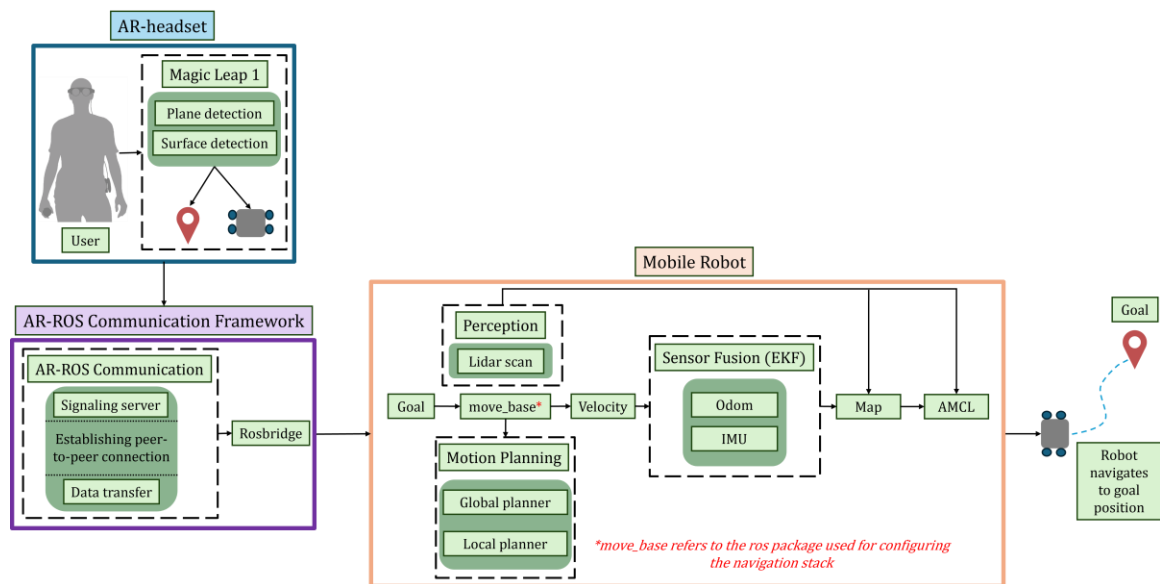


Figure 3.1: System Architecture

3.1 AR-HMD Colocalization with Mobile Robot

Colocalization is the process by which each device, such as an augmented reality head-mounted display (AR-HMD) and a mobile robot, determines its position within a shared coordinate system. This process is crucial for enabling seamless interaction and coordination between the devices. In our research, we focus on calculating the relative transformations between an AR-HMD and a mobile robot. Given that both systems are dynamic and continuously moving, a robust and continuous localization method is necessary.

Our approach to colocalization does not rely on artificial landmarks, such as fiducial markers or spatial anchors, nor does it depend on natural landmarks like vision-based cues. This markerless method simplifies the calibration process significantly. Without predefined markers, the setup becomes easier and faster, reducing potential sources of error associated with marker placement. Additionally, our approach is more robust for real-time applications where both the robot and the AR device need to operate in coordination.

For our research, we used a Magic Leap 1 (ML1) headset [58] and a 4-wheel differential drive mobile robot. Figure 3.2 illustrates that the robot and the AR headset follow different coordinate frame conventions. Initially, both the ML1 and the robot are oriented in their respective forward-facing directions. This discrepancy can lead to inaccuracies in spatial data interpretation and integration, as the x-axis of the ML1 and the robot may not be aligned. Such misalignment can cause problems when setting navigation goals, as the robot may misinterpret the positional data.

The dynamic nature of both the AR-HMD and the robot introduces additional complexity. As both systems move, it is essential to maintain accurate localization through real-time updates. This continuous movement requires a robust transformation method to ensure precise alignment and coordination. To simplify the complexity associated with dynamic frames, we assume that the x-direction of both the ML1 and the robot are aligned, which helps reduce errors in interpreting the coordinate data.

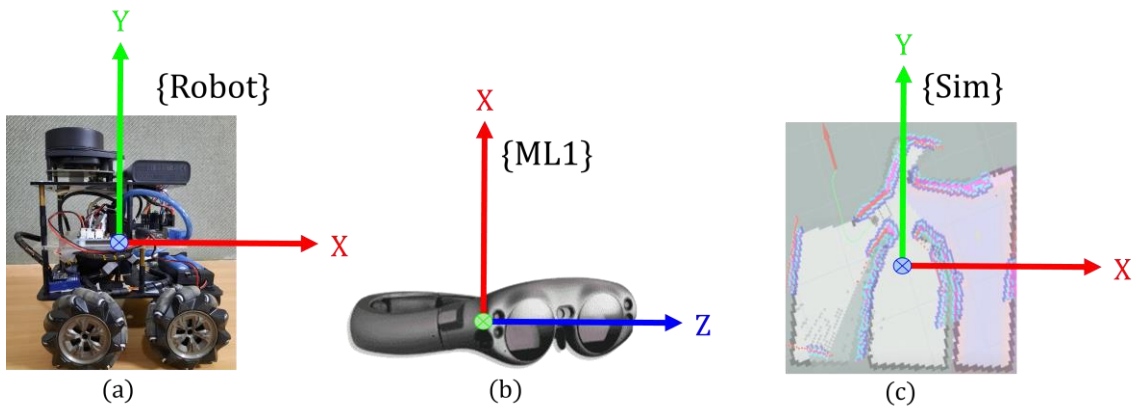


Figure 3.2: Frame Representation (a) Robot (b) ML1 (c) Sim

Figure 3.2 (c) presents the third frame, known as the *Sim* frame, which represents the position and orientation of the robot in the simulation software. This frame is crucial for Simultaneous Localization and Mapping (SLAM) navigation, as it provides a reference for the robot's real-world location. Even after correcting the orientation of both systems, we must account for the translation between the *ML1* and the *Robot*. To address this, we introduce a Global *G* frame to find coordinates of the robot in the real world, with respect to *Sim*. This frame ensures that all spatial data is referenced correctly in the real-world coordinate system. Figure 3.3 shows the transformation between *ML1* and *Sim*, which represents the robot's pose.

Once the coordinate frames are aligned, we can send the initial and goal poses to the robot for navigation. Using the controller, we detect the robot and establish its initial position. We then set a goal position in the environment using the same procedure. This process ensures that the coordinates received by the robot adhere to the real-world coordinates provided by the ML1. This alignment guarantees that the robot correctly interprets and follows the goal position sent from the AR-HMD, resulting in accurate navigation (Figure 3.6).

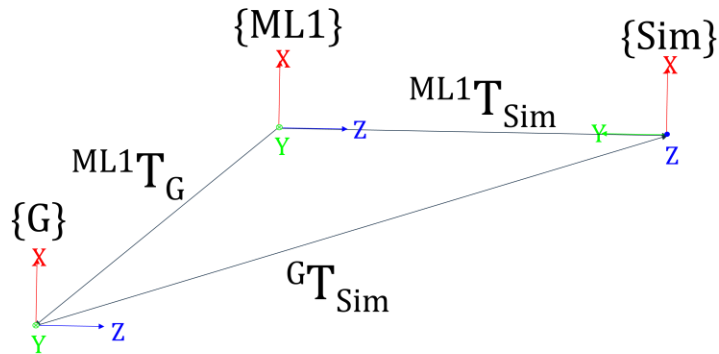


Figure 3.3: Transformation between ML1 and Sim

$${}^{ML1}T_{Sim} = (R_x(90))_x (P_{Sim})$$

$$(P_G) = {}^{ML1}T_{Sim}^{-1} \times (P_{ML1})$$

$${}^{ML1}T_{Sim} = \begin{bmatrix} I & G \\ 0 & 1 \end{bmatrix}$$

$${}^G T_{Sim} = ({}^{ML1}T_G)^{-1} \times ({}^{ML1}T_{Sim})$$

Both the robot and goal position are with respect to the ML1 frame, but for navigation, these frames must be with reference to the RVIZ frame. The final transformation is given in Figure 3.4. Once we have these transformations, we can send the coordinates via ROS, given by ${}^{Sim}T_{Robot}$ and ${}^{Sim}T_{Goal}$.

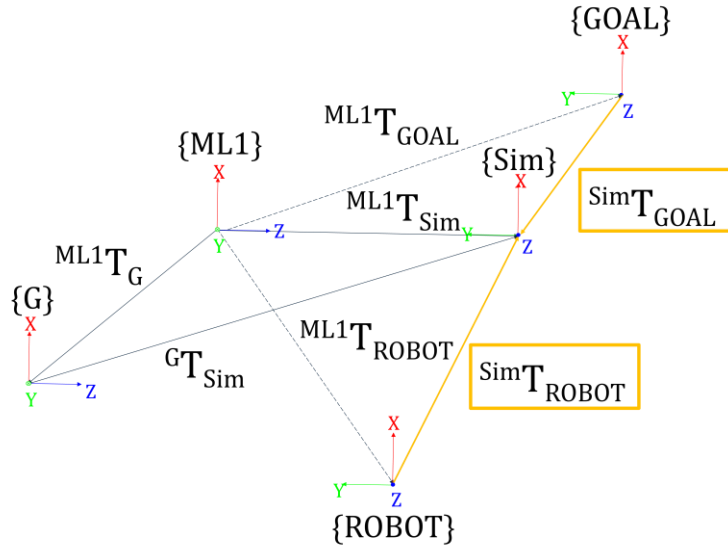


Figure 3.4: Final Transformation

$${}^{Sim}T_{ROBOT} = ({}^G T_{RVIZ})^{-1} \times ({}^{ML1}T_G)^{-1} \times ({}^{ML1}T_{ROBOT})$$

$${}^{Sim}T_{GOAL} = ({}^{ML}T_{GOAL})^{-1} \times ({}^{ML1}T_G)^{-1} \times ({}^{ML1}T_{GOAL})$$

By implementing this colocalization method, the mobile robot and the AR-HMD can work in coordination. The accurate alignment of coordinate frames and the continuous updates ensure that the robot navigates effectively within the shared space, responding to dynamic changes and achieving the desired goals with precision.

3.2 AR-ROS Communication Framework

Integrating the ML1 headset with the Robot Operating System (ROS) posed a significant challenge since the headset does not inherently support ROS communication, unlike other devices such as the Microsoft HoloLens [31]. To bridge this gap, we utilized the Unity 3D game engine to design and develop a custom application for the ML1. Our application leverages ML1's surface detection capabilities and the ML Web Real-Time Communication (MLWEBRTC) framework [59] to enable effective communication between the ML1 headset and ROS.

The communication between the ML1 and MLWEBRTC is established through a signaling server, which acts as a mediator to facilitate data exchange. One of the key features of ML1 that we utilized is plane detection, which allows the headset to identify horizontal and vertical surfaces in the environment. This feature is crucial for generating a spatial map, enabling the ML1 to recognize surfaces such as walls, tables, floors, ceilings, and undersides. We added functionality to detect our mobile robot based on its kinematic constraints to tailor this capability to our specific needs.

In our system, we focused on detecting three types of surfaces: walls, floors, and the robot itself (Figure 3.5). The detection process involves checking the dot product of the surface normal to distinguish between vertical and horizontal surfaces. If the dot product is less than or equal to a specified wall threshold, the surface is classified as a wall. If not, the next step is to check if the dot product is negative, which would indicate a downward-facing surface, classifying it as a ceiling. For detecting the robot, we use the vertical range of a raycast to identify surfaces within the robot's operational height above the floor. If the detected point falls within this range, the surface is classified as a robot.

Robot initialization and setting goals for the robot are accomplished using a point-and-click method with the ML1 controller. The user looks around the environment and points the controller at the mobile robot to set its initial position. Similarly, the user chooses a goal point by aiming the controller at the floor. Our framework includes a safety feature that allows navigation goals to be set only when the floor is detected by the ML1, preventing incorrect goal coordinates from being assigned to surfaces other than the floor.

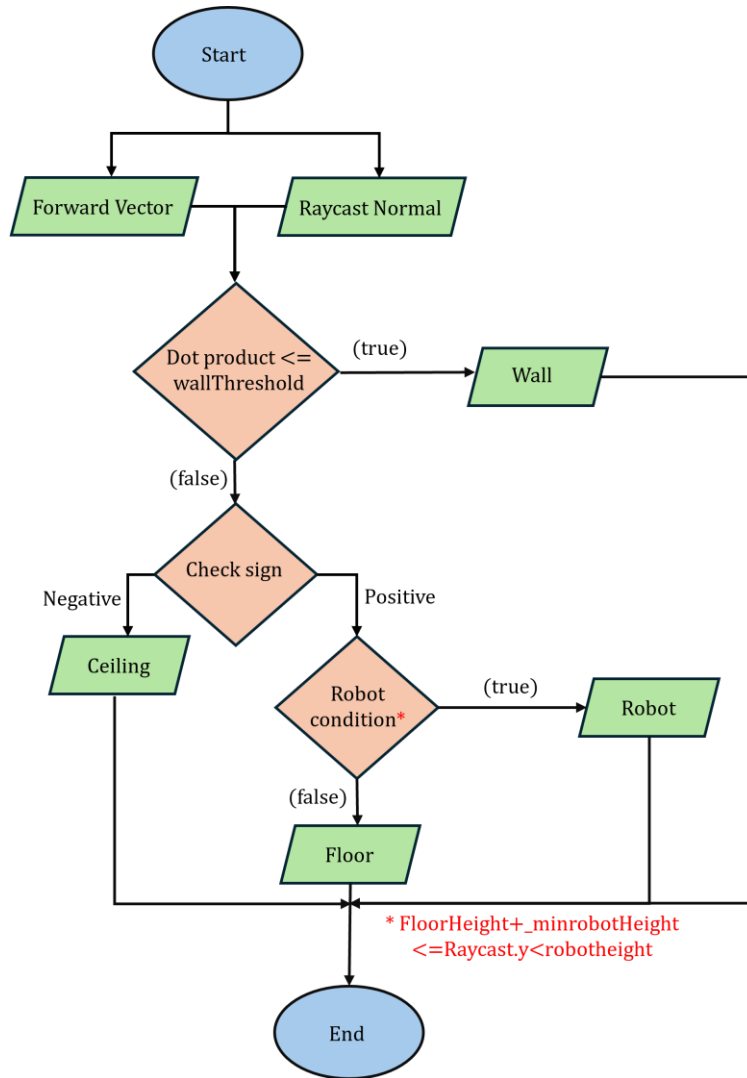
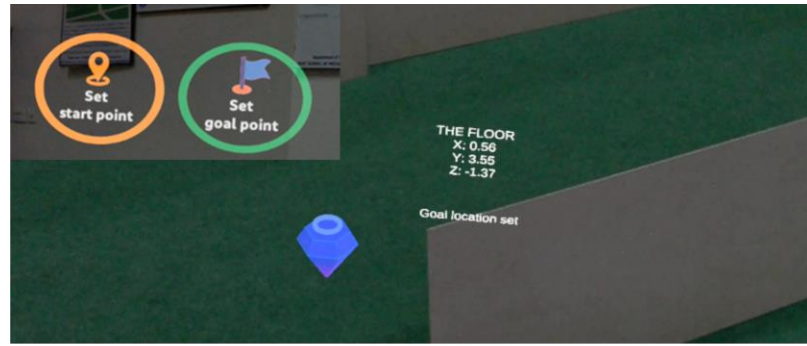


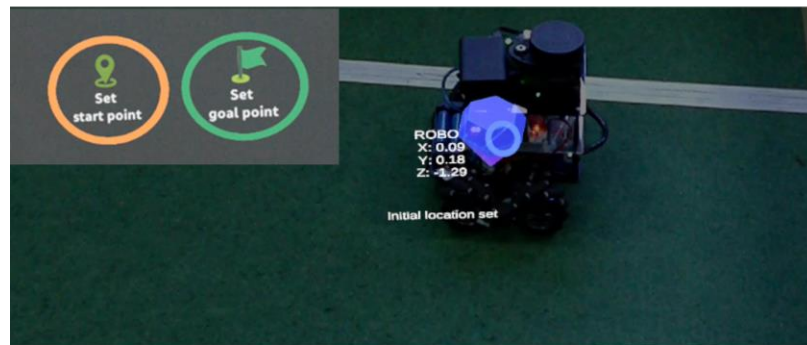
Figure 3.5: Surface Detection

. For this research, we conducted experiments in a non-cluttered environment to ensure accurate detection, considering only the mobile robot as the robot.

The ML1’s ability to anchor virtual objects to detected planes ensures stability in their placement, maintaining consistency in sending data between coordinate systems. Once a surface is detected, a virtual object is created at the goal or initial position coordinates when the user pulls the controller’s trigger. These coordinates are then received by MLWEBRTC and transmitted to the ROS navigation stack via ROSBridge.



(a)



(b)

Figure 3.6: AR view for AR-based Hector SLAM (a) Detect Robot (b) Set goal

ROSBridge enables the ML1 to communicate with the ROS navigation stack. The initial and goal coordinates sent from the ML1 are received by the robot, which then uses them for navigation. This seamless integration of the ML1 with ROS allows for accurate and efficient navigation of the mobile robot, leveraging the AR capabilities of the ML1 to enhance the overall user experience and operational efficiency.

3.3 Hector SLAM with EKF sensor fusion

Hector SLAM [60] is a technique that leverages high-frequency LIDAR data to create a detailed map of the robot's environment. It operates by aligning consecutive LIDAR scans through a process known as scan matching. In this method, Hector SLAM does scan matching based on the Gauss-Newton approach for aligning the new LIDAR scans with the previous ones. The objective of the algorithm is to find the transformation that minimizes the error between the current scan and the existing map. This process involves

iteratively refining the transformation to update both the map and the robot's pose, resulting in a detailed occupancy grid map that is continuously published on the map topic.

In addition to map generation, Hector SLAM provides transformations via the `tf` (transform) library, which describes the robot's movement relative to the map frame. This continuous stream of transformation data allows for accurate tracking of the robot's position and orientation over time. Once a map has been generated, the robot can utilize AMCL (Adaptive Monte Carlo Localization) [61] to localize itself within this map. AMCL subscribes to the map topic provided by Hector SLAM and uses incoming LIDAR scans to estimate the robot's pose through a particle filter. The resulting pose estimate is published to the `tf` tree, offering a real-time transformation between the map frame and the robot's base frame.

To achieve more accurate and reliable state estimation, our system incorporates an Extended Kalman Filter (EKF) [62] for sensor fusion. The EKF node integrates data from multiple sources: odometry, IMU (Inertial Measurement Unit) data, and pose estimates from AMCL. By combining these inputs, the EKF produces a refined estimate of the robot's state. This step is crucial for mitigating issues such as laser drifts and inaccuracies in localization, which can occur due to the inherent noise and limitations of individual sensors.

With a robust state estimate in place, the robot can effectively navigate toward its goal position using planners from the ROS navigation stack. This navigation process involves two key components: the global planner and the local planner.

3.3.1 Global Planner

For our global planner [63], we employed Dijkstra's algorithm [46]. This algorithm is well-known for its reliability and optimal path-finding capabilities, as it always finds the shortest path in a weighted graph. In the context of our system, the global planner uses the costmap generated from sensor data. Each cell in the costmap corresponds to a vertex in the graph, and the movement cost between cells forms the edge weights. According to [64], the distance to the start node s is initialized to 0, i.e., $d(s) = 0$, while the distance to all other

nodes v is set to infinity, i.e., $d(v) = \infty$, for all $v \neq s$. The algorithm then iteratively selects an adjacent node u with the smallest distance $d(u)$ and updates the distances to its neighbors v via the edge (u, v) .

This method ensures that the robot can plan a reliable and optimal path from its current position to the goal position, taking into account the layout and obstacles within the environment as represented by the costmap.

3.3.2 Local Planner

For local planning, we utilized the Trajectory Rollout algorithm [65] with the Dynamic Window Approach (DWA) [66]. It is specifically designed for real-time obstacle avoidance and navigation. The Trajectory Rollout algorithm evaluates possible trajectories by simulating the robot's motion over a short time period. It then selects the optimal path based on a cost function that balances three key criteria: progress toward the goal, distance from obstacles, and velocity. The DWA component further refines this process by limiting the search space to dynamically feasible trajectories.

$$V_t = \{(v, w) | v_{\min} \leq v \leq v_{\max}, w_{\min} \leq w \leq w_{\max}\}$$

where v_{\min} , v_{\max} are the minimum and maximum feasible linear velocities and w_{\min} , w_{\max} are the minimum and maximum feasible angular velocities.

It also considers the robot's current velocity (v, w) and acceleration constraints (\dot{v}, \dot{w}) .

$$V_d = \{(v, w) | v \in [v_a - \dot{v} \cdot t, v_a + \dot{v} \cdot t], w \in [w_a - \dot{w} \cdot t, w_a + \dot{w} \cdot t]\}$$

where \dot{v} and \dot{w} represent the linear and angular accelerations applied at time t , and v_a and w_a are the actual linear and angular velocities.

This means that the local planner not only accounts for the desired movement but also respects the robot's physical limitations in terms of linear and angular acceleration. By doing so, the DWA ensures that the planned trajectories are not only optimal but also executable given the robot's capabilities.

In summary, the integration of Hector SLAM with EKF sensor fusion and the ROS navigation stack's global and local planners creates a comprehensive and robust system for indoor mobile robot navigation. The use of high-frequency LIDAR data for map generation, combined with advanced localization and state estimation techniques, ensures accurate and efficient navigation even in complex environments. The global and local planners work together to provide optimal and feasible paths, enabling the robot to reach its goals safely and reliably.

This chapter outlined the methodology adopted for our proposed method. In the next chapter, we'll highlight the results obtained with a detailed discussion.

3.4 Summary

Chapter 3 has outlined the methodology used in this research. We described the AR-HMD colocalization process with the mobile robot, ensuring accurate spatial alignment between the AR-HMD and the robot. The AR-ROS communication framework explains how data is transferred from the ML1 to the mobile robot. Additionally, we discussed the implementation of Hector SLAM with EKF sensor fusion, which combines LIDAR and IMU data to enhance the robot's localization and mapping capabilities. The chapter also covered the global and local planning strategies, specifically Dijkstra's algorithm and Trajectory Rollout, respectively, to ensure optimal pathfinding and obstacle avoidance. Highlighting our methodology sets the stage for the experimental results and analysis presented in the next chapter, validating the effectiveness of the proposed approach.

CHAPTER 4. EXPERIMENTAL RESULTS

In this chapter, we present and discuss the results of our proposed method. We compare the performance of Hector SLAM and AG, explaining why Hector SLAM is the better approach. Next, we test our proposed AR-based Hector SLAM method for indoor mobile robot navigation.

4.1 Experimental Setup

We created custom environments in Gazebo for three experimental scenarios (S1, S2, S3), based on Mujahed et al.'s [11] experiments 1, 2, and 5. In each scenario, we evaluated the performance of robot navigation using both Hector SLAM and the AG navigation method. Figure 4.1 illustrates the simulation environment setup for each scenario. The robot creates a real-time map of the environment for Hector SLAM. Our simulation experiments utilized a Turtlebot 3 Burger model [67], which has a maximum translational velocity of 0.22 m/s and a maximum rotational velocity of 2.84 rad/s and is equipped with a laser and an IMU.

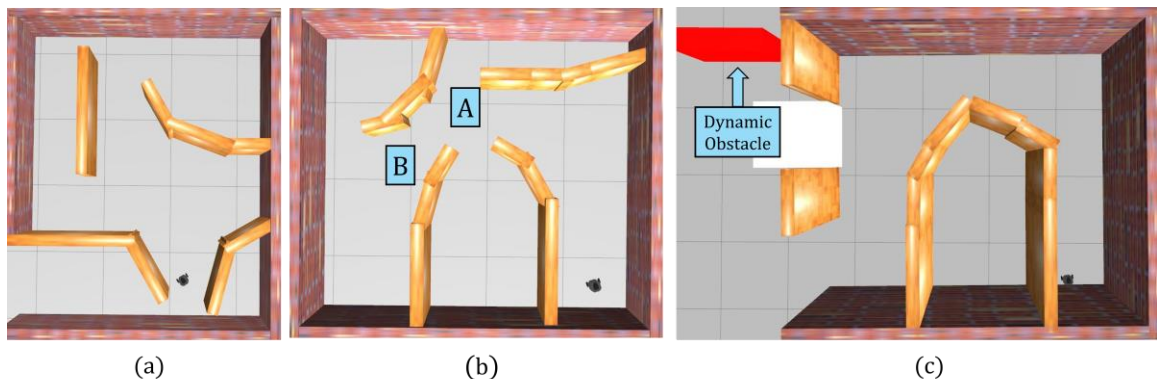


Figure 4.1: Simulation environment view for the three scenarios

After testing the effectiveness of Hector SLAM in our simulation experiments, we tested our proposed AR-based Hector SLAM in a real-world scenario. The setup is similar to the simulation environment. In Figure 4.3, Figure 4.5, and Figure 4.7, the pink lines surrounded by dark blue lines indicate obstacles in the local costmap as detected by the robot along its path. We used a four-wheeled mobile robot platform operating on a differential drive

model. This platform uses a Raspberry Pi 4B model (8GB RAM) and an MPU5060. An RPLIDAR A1 is mounted on top, with an angular resolution of 1° and a scanning range of 0.15 to 12 meters. Magic Leap 1 (ML1) is used as the AR-HMD for this experiment.

4.1.1 Scenario 1

The robot must navigate through a sparse environment as seen in Figure 4.2.

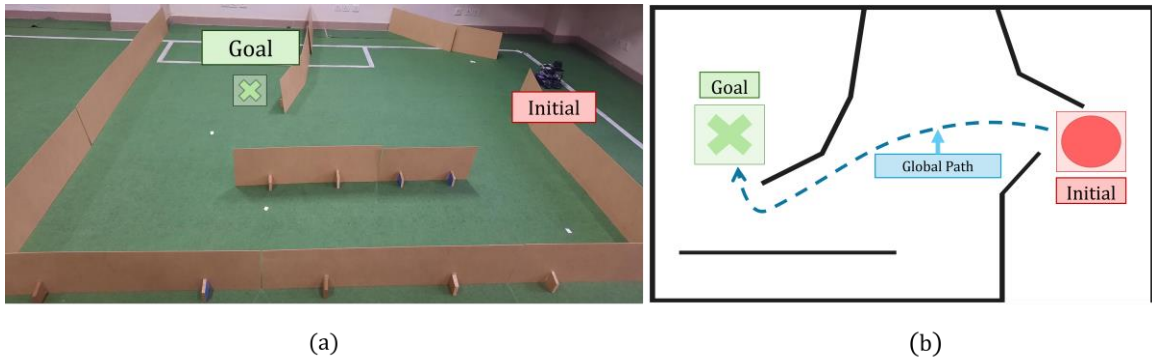


Figure 4.2: (a) Scenario 1 real-world set up (b) Path from initial to goal point.

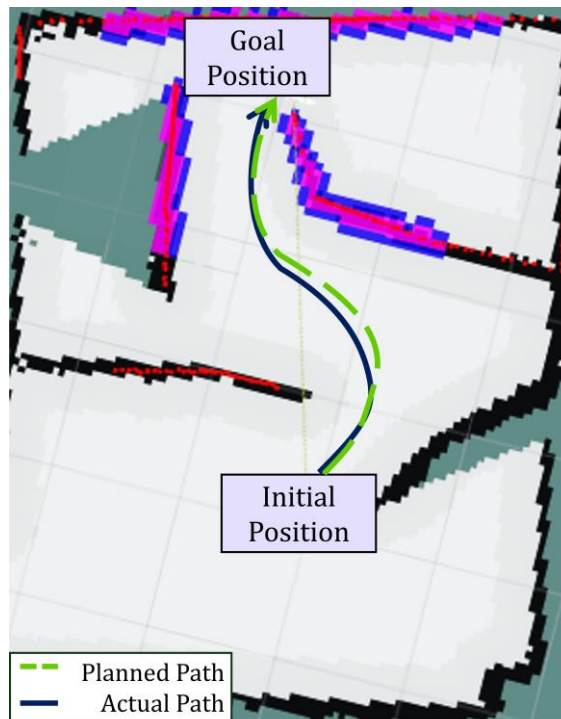


Figure 4.3: Scenario 1 Navigation

The primary objective is to reach the specified goal while detecting and avoiding these obstacles. This requires the robot to continuously update its path, ensuring it can navigate around the obstacles without collisions. The sparse nature of the environment means that the obstacles are not densely packed, but still present enough of a challenge to test the robot's navigation capabilities. Figure 4.3 shows the real-world navigation view of our mobile robot.

4.1.2 Scenario 2

The second scenario's environment, as shown in Figure 4.4, is designed with a challenging U-shaped obstacle, creating a local minima that the robot must navigate around.

The objective is to ensure the robot can successfully avoid getting trapped in this local minima while making its way towards the goal point. It must also navigate through a relatively narrow passage, labeled *A* and *B*, which lies outside the U-shaped obstacle.

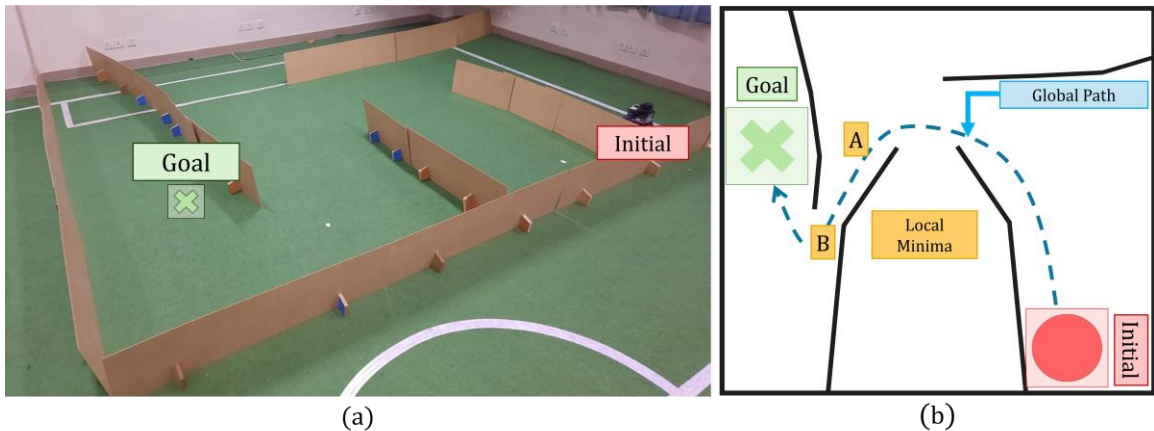


Figure 4.4: (a) Scenario 2 real-world set up (b) Path from initial to goal point, avoiding the local minima. Points A and B indicate the narrow passage that the robot must go through.

This requires precise and accurate path planning to avoid collision with obstacles. Figure 4.5 shows the real-world navigation view.

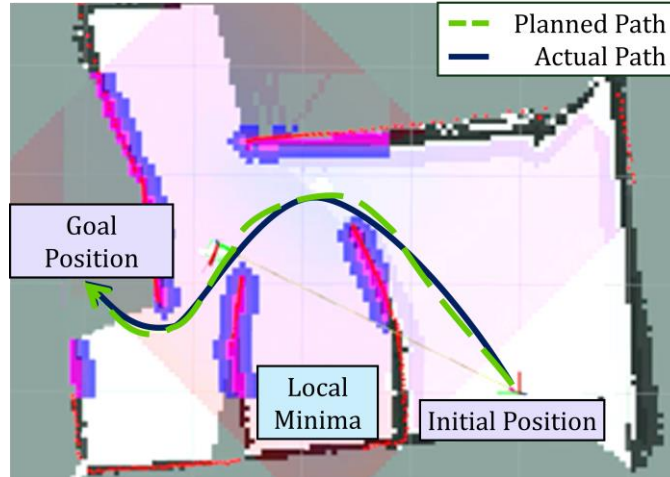


Figure 4.5: Scenario 2 navigation

4.1.3 Scenario 3

Scenario 3 involves a dynamic obstacle that is placed in the robot's path, as it is navigating towards the goal position. The robot must find the shortest path to the goal position.

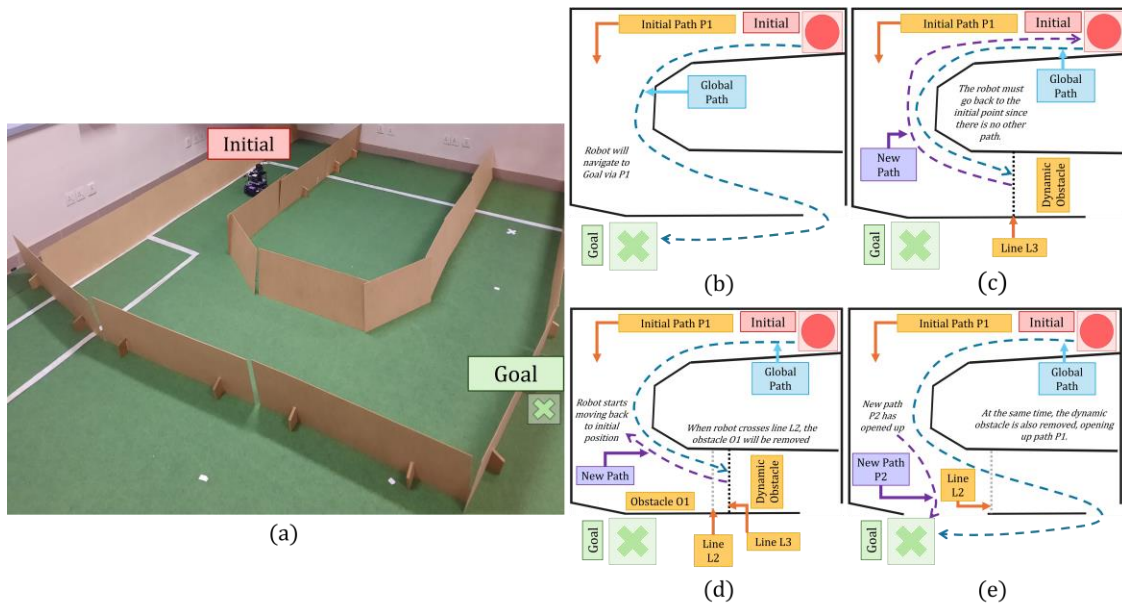


Figure 4.6: (a) Scenario 3 real-world set up (b) Robot starts at the initial point, going to Goal via Path P1. (c) A dynamic obstacle is placed in its path when robot reaches line L3. (d) It starts to move back to its initial position. (e) The robot must take the shortest path to reach the goal, and so it chooses path P2.

It starts from path P1, navigating to the goal position after receiving the coordinates (Figure 4.6 (a)). Upon reaching line L3, we introduce a dynamic obstacle in path P1 (Figure 4.6 (b)). Now, there is only one path that the robot can follow, i.e., back to the initial position (Figure 4.6 (c)). As soon as it crosses the line L2, we remove obstacle O1. A new path P2 opens up, and at the same time, the dynamic obstacle is removed from P1. Now the robot must choose between the new path and the original planned path for reaching the goal. It must choose the path with the shortest distance to the goal, so it chooses P2 (Figure 4.6 (d)). Figure 4.7 shows the real-world navigation view.

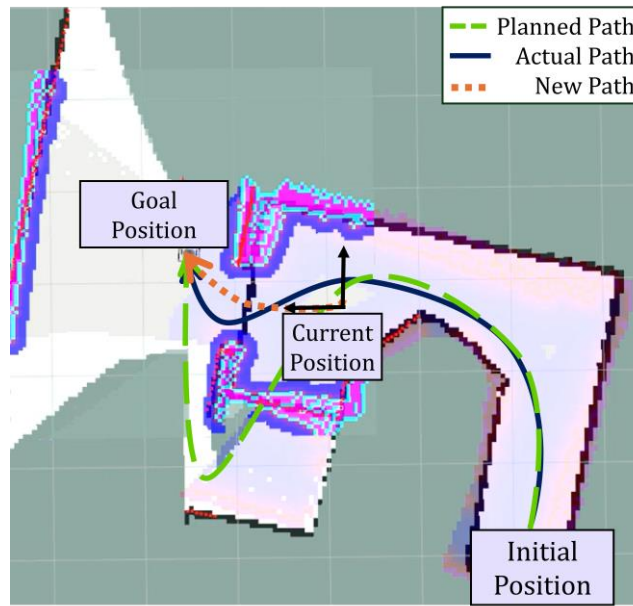


Figure 4.7: Scenario 3 navigation

4.1.4 Evaluation Metrics

The evaluation metrics [11] for calculating the performance and efficiency of our system are described below:

1. **Total execution time (T_{tot}):** Total time needed by the robot to reach the goal.
2. **Path length (P_{len}):** Total distance traveled from the robot's initial position to the goal position:

$$P_{len} = \int_{x_i}^{x_g} \left(1 + (f'(x))^2\right)^{1/2} dx$$

where x_i and x_g denote the coordinates of the initial and target locations.

- 3. Accumulated linear and angular jerks (J_{acc} , ζ_{acc}):** Sudden changes in acceleration are called jerks. It measures the smoothness of the robot's movement along the trajectory:

$$J_{acc} = \frac{1}{T_{tot}} \int_0^{T_{tot}} [\ddot{v}(t)]^2 dt$$

$$\zeta_{acc} = \frac{1}{T_{tot}} \int_0^{T_{tot}} [\ddot{w}(t)]^2 dt$$

- 4. Lateral stress (S_{lat}):** It tells us about the stability of the robot by measuring the centrifugal force acting upon it along the length of the trajectory.

$$S_{lat} = \int_0^{T_{tot}} \frac{v(t)^2}{r(t)} dt$$

- 5. Tangential stress (S_{tng}):** This metric tells us about sudden decelerations or accelerations that might cause the wheels to slip while turning:

$$S_{tng} = \int_0^{T_{tot}} |v(t)| dt$$

Table 1 shows the mean results of our simulation-based experiments for both methods. We tested each scenario 15 times, for both Hector SLAM and AG, resulting in a total of 90 experiments. In Hector SLAM, the robot does not depend upon the presence or absence of gaps for navigating to the goal position. It must choose the path with the shortest distance to the target location.

4.2 Summary

This chapter explains the experimental setup for both simulation and real-world experiments. We create three scenarios, replicating experiments 1, 2, and 5 of Mujahed's [11] work. We first test the performance of Hector SLAM and AG in our simulation

environment, using evaluation metrics to check its effectiveness. Our real-world experiments are set up similarly like our simulation-based scenarios. We'll discuss the results for both in the next chapter.

CHAPTER 5. DISCUSSION

This chapter presents a discussion of the results obtained for each experiment. First, we evaluate its performance in a simulation environment, testing both Hector SLAM and AG navigation. Then we test our AR-based Hector SLAM method in a real-world set up. In Hector SLAM, the robot maps the environment while it navigates towards the goal position. In contrast, AG navigation does not map the environment, since it is a reactive collision avoidance method. It detects obstacles in real-time and navigates around them to avoid collision. This method may work well in smaller environments, but it can be difficult in larger or complex environments. The robot can quickly create a path to the goal point if it keeps track of obstacles or landmarks.

For both simulation and real-world experiments, the robot uses Hector SLAM with EKF for sensor fusion to improve localization accuracy while mapping. There is no pre-built map of the environment. We used Dijkstra's algorithm for global path planning and the Trajectory rollout algorithm for local path planning.

5.1 Jerk

We analysed linear and angular jerks (J_{acc} , ζ_{acc}) for each scenario and presented the results in the following sections.

5.1.1 Scenario 1

AR-based Hector SLAM showed improvement in performance for both linear and angular jerks. From Table 1, the J_{acc} for Hector SLAM is 0.598, while for AG it is 0.929. SLAM has a significantly lower linear jerk than AG, indicating that it has a smooth movement with fewer abrupt changes in acceleration. The ζ_{acc} for Hector SLAM is 1.06, lower than AG's 1.25 showing that it has smoother rotational movement as well.

Hector SLAM's ability to map the environment in real-time is the reason why we get lower values of jerk in scenario 1.

Table 1: Comparison of AR-based Hector SLAM and AG Navigation

Scenario	Method	Total time	Path length	Linear jerk	Angular jerk	Lateral stress	Tangential stress
1	AR-based Hector SLAM	10.5	3.32	0.598	1.06	0.801	0.915
	AG	10.6	3.398	0.929	1.25	1.015	1.052
2	AR-based Hector SLAM	16.4	3.703	1.037	0.96	1.463	1.019
	AG	16.9	3.871	0.87	1.181	1.706	0.939
3	AR-based Hector SLAM	25.97	4.865	1.386	1.385	2.675	2.414
	AG	29.774	4.839	1.522	1.547	3.235	2.719

It can plan its path while avoiding obstacles, so it doesn't produce abrupt jerks. This allows the robot to make planned and precise movements. In contrast, we get higher values of jerk with AG navigation because of its reactive nature. Its trajectory is less smooth since it makes adjustments in real-time when it detects an obstacle.

On average, AR-based Hector SLAM decreased J_{acc} by 35.63% and ζ_{acc} by 15.20% in scenario 1. The jerk profile in Figure 5.1 shows a comparison of velocity, acceleration, and jerk for both methods.

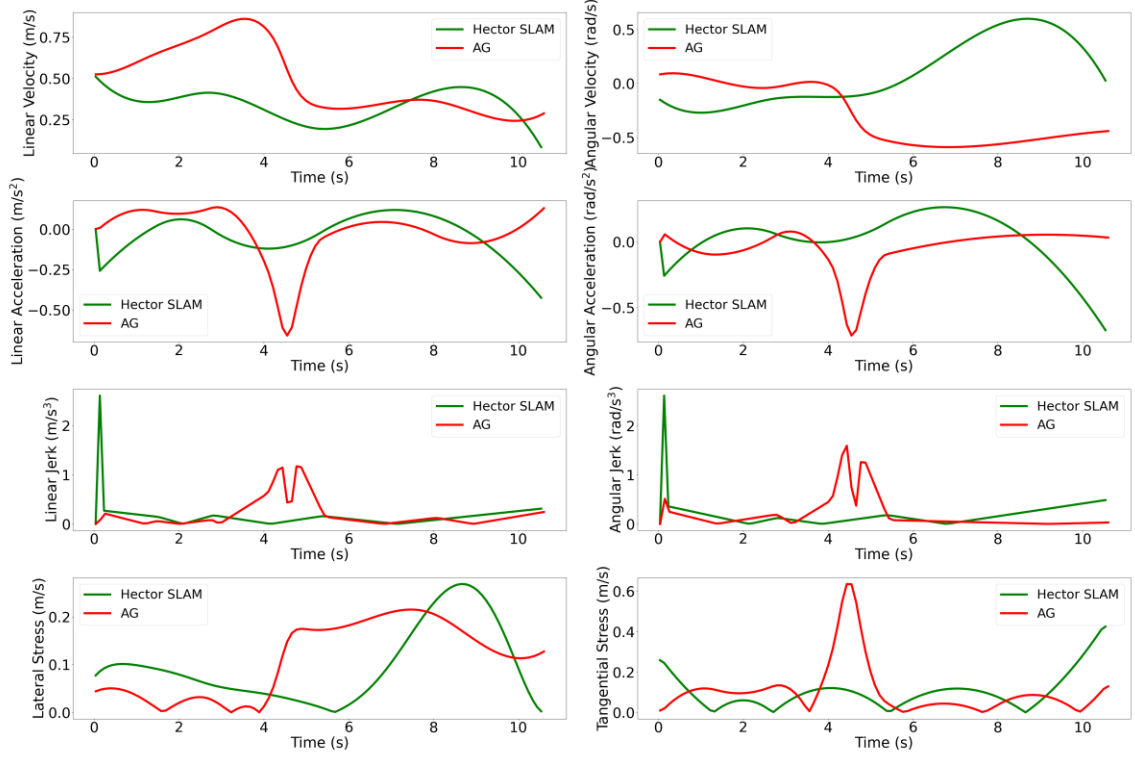


Figure 5.1: Velocity, acceleration, jerk, and stress profiles for AR-based Hector SLAM (green) and AG navigation (red) for Scenario 1

5.1.2 Scenario 2

In Scenario 2, the environment includes a U-shaped obstacle creating a local minimum and a narrow passage that the robot must navigate to reach its goal. From Table 1, it is evident that AR-based Hector SLAM has a higher J_{acc} value of 1.037 compared to AG navigation's 0.87, indicating that AG navigation achieves smoother linear movements in this scenario. The reactive nature of AG navigation seems to be better suited for quickly adapting to the narrow passage, resulting in smoother linear motions.

Conversely, AR-based Hector SLAM demonstrates a lower ζ_{acc} value of 0.96 compared to AG navigation's 1.181, indicating smoother rotational movements. Effective navigation through the local minima and narrow passage requires precise control, which is facilitated by the global path planning and mapping capabilities of Hector SLAM. This helps avoid abrupt changes in angular acceleration.

AR-based Hector SLAM decreased ζ_{acc} by 18.71% on average in Scenario 2. The jerk profile for this scenario, shown in Figure 5.2, presents a comparison of velocity, acceleration, and jerk for both methods.

5.1.1 Scenario 3

Scenario 3 presents the most complex environment, incorporating a dynamic obstacle in the robot's path. In this scenario, AR-based Hector SLAM records a J_{acc} value of 1.386 and a ζ_{acc} value of 1.385, whereas AG navigation shows values of 1.522 and 1.547, respectively. Even with increased complexity, SLAM performs better than AG navigation in terms of trajectory smoothness.

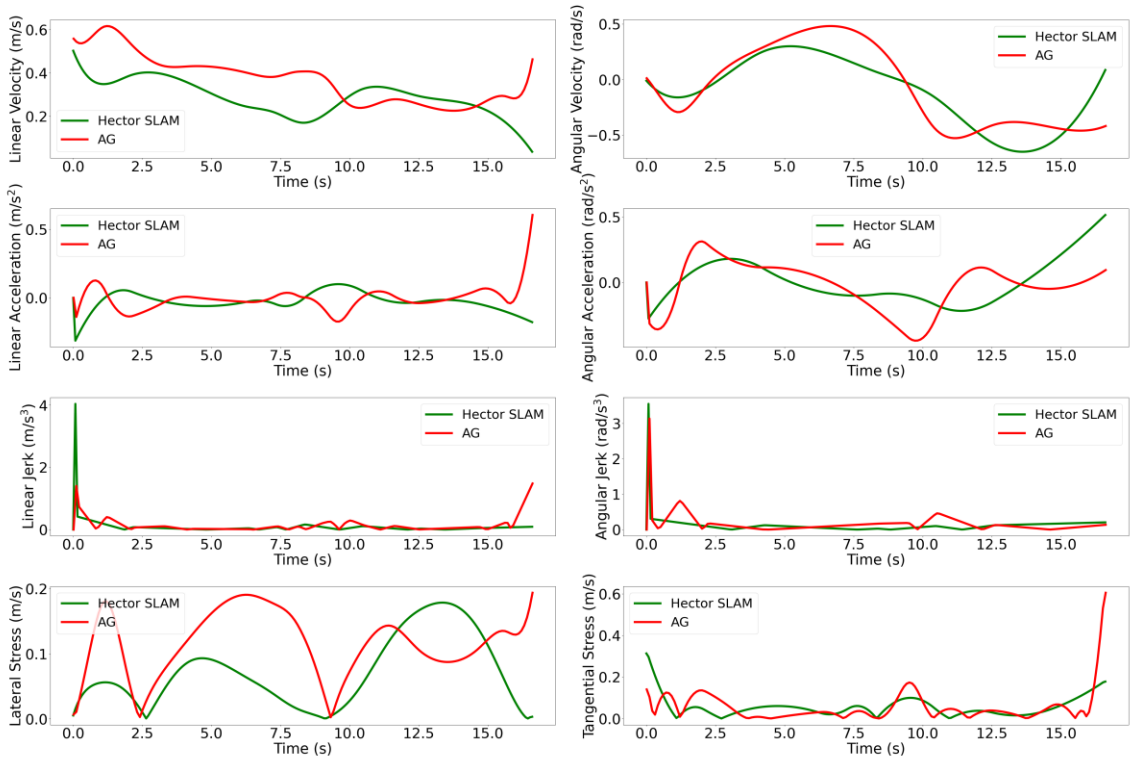


Figure 5.2: Velocity, acceleration, jerk, and stress profiles for AR-based Hector SLAM (green) and AG navigation (red) for Scenario 2

The lower jerk values for Hector SLAM indicate its ability to handle dynamic obstacles while maintaining a smoother trajectory. Conversely, the higher jerk values for AG navigation suggest that its reactive nature leads to more abrupt and frequent changes in linear and angular acceleration.

AG navigation’s strength lies in its reactive collision avoidance capability, allowing it to avoid dynamic obstacles as they appear. However, its lack of path planning can result in increased navigation time in larger environments or scenarios with multiple gaps. On average, Hector SLAM decreased J_{acc} by 8.94% and ζ_{acc} by 10.47% in scenario 3. The jerk profile for this scenario, as shown in Figure 5.3 , presents a comparison of velocity, acceleration, and jerk for both methods.

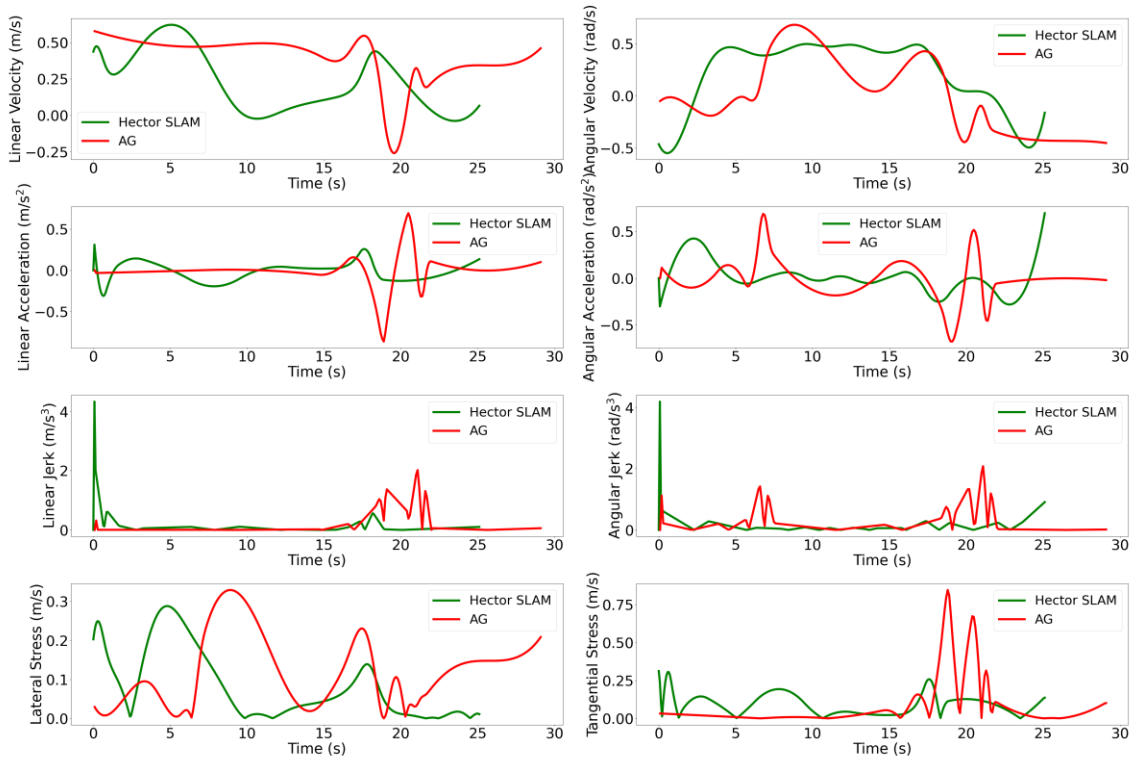


Figure 5.3: Velocity, acceleration, jerk, and stress profiles for AR-based Hector SLAM (green) and AG navigation (red) for Scenario 3

5.2 Stress

We analyzed lateral (S_{lat}) and tangential stress (S_{tng}) for each scenario and present the results in the following sections.

5.2.1 Scenario 1

In Scenario 1, which focuses on navigation in a sparse environment, AR-based Hector SLAM exhibited a lower S_{lat} value of 0.801, in contrast to AG navigation's value of 1.015. The reduced S_{lat} suggests that Hector SLAM offers enhanced stability by minimizing the centrifugal forces acting on the robot. This reduction is critical for ensuring the robot's stability and preventing tipping, thereby maintaining a smoother and safer trajectory.

The S_{tng} for AR-based Hector SLAM is measured at 0.915, lower than AG navigation's 1.052. This indicates that Hector SLAM results in fewer abrupt accelerations or decelerations. The lower S_{tng} implies that the robot's speed transitions more gradually and in a controlled manner, thereby reducing the risk of wheel slippage and enhancing traction.

On average, AR-based Hector SLAM demonstrates a 21.08% decrease in lateral stress (S_{lat}) and a 13.02% improvement in tangential stress (S_{tng}). Figure 5.1 shows the comparison of velocity, acceleration, and stress for Scenario 1 for both methods.

5.2.2 Scenario 2

Table 1 presents the stress values for Scenario 2. The S_{lat} for AR-based Hector SLAM is 1.463, which is lower than AG navigation's 1.706. This reduced lateral stress suggests that Hector SLAM offers better stability while avoiding the local minima and navigating through the narrow passage. In environments with confined spaces, the robot must have lower lateral stress to prevent unstable motion.

However, AR-based Hector SLAM has a higher S_{tng} value at 1.019 as compared to AG navigation's 0.939. This indicates that Hector SLAM may experience more sudden accelerations or decelerations. Increased tangential stress is because of the precise control required to navigate through the narrow passage, necessitating abrupt speed adjustments.

The risk of wheel slippage increases but the overall stability due to lower S_{lat} indicates that Hector SLAM maintains controlled movement.

On average, AR-based Hector SLAM shows a 14.24% decrease in S_{lat} . The comparison of velocity, acceleration, and stress for Scenario 2, for both methods, is illustrated in Figure 5.2.

5.2.3 Scenario 3

Scenario 3 presents a more complex environment with dynamic obstacles. AR-based Hector SLAM has a S_{lat} of 2.675, lower than AG navigation's 3.235. These values show that Hector SLAM maintains good stability even if there are moving obstacles within the environment. With a lower S_{lat} , the robot will not be destabilized by the centrifugal forces that act upon it during dynamic obstacle avoidance. This helps the robot navigate safely through unpredictable environments with moving obstacles.

AR-based Hector SLAM has a lower S_{tng} at 2.414, compared to AG navigation's 2.719. This shows that Hector SLAM handles sudden speed changes smoothly when avoiding dynamic obstacles. Lower S_{tng} means that accelerations and decelerations are controlled, thus, reducing the risk of wheel slippage for better maneuverability and traction.

Hector SLAM shows a 17.31% decrease in S_{lat} and S_{tng} by 11.22%, on average. The comparison of velocity, acceleration, and stress for scenario 3, in both methods, is shown in Figure 15c.

5.3 Quantitative Performance Comparison

In this section, we summarize the results of our experiments, highlighting the percentage decrease AR-based Hector SLAM has shown over AG navigation.

5.3.1 Scenario 1

In scenario 1, AR-based Hector SLAM decreased:

- Linear jerk by 35.63%

- Angular jerk by 15.20%
- Lateral stress by 21.08%
- Tangential stress by 13.02%

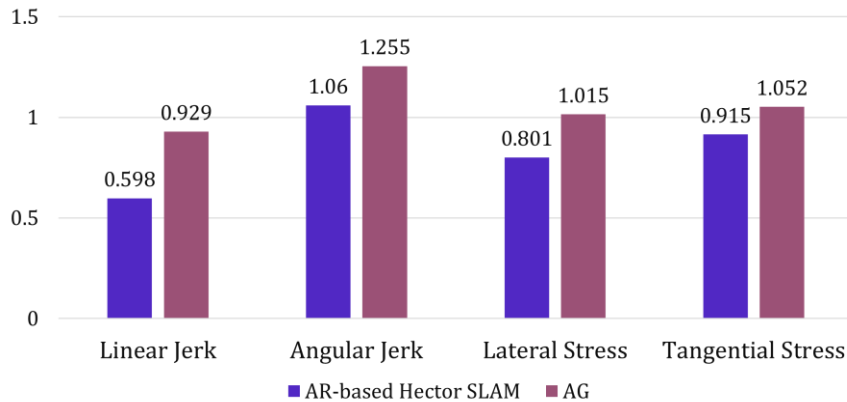


Figure 5.4: Scenario 1 mean decrease in jerk and stress

A t-test for each evaluation metric gave us the following results:

- **Linear jerk:** Comparing Hector SLAM (M=0.5984, SD=0.234) and AG (M=0.929, SD=0.327) yielded a p-value of 0.003, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.003$).
- **Angular jerk:** Comparing Hector SLAM (M=1.063, SD=0.218) and AG (M=1.255, SD=0.299) yielded a p-value of 0.05, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.05$).
- **Lateral stress:** Comparing Hector SLAM (M=0.801, SD=0.246) and AG (M=1.015, SD=0.124) yielded a p-value of 0.005, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.005$).
- **Tangential stress:** Comparing Hector SLAM (M=0.915, SD=0.17) and AG (M=1.052, SD=0.162) yielded a p-value of 0.03, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.03$).

Table 2: Scenario 1 t-test

Evaluation Metric	Method	Mean	SD	t-value	p-value
Linear jerk	AR-based Hector SLAM	0.5984	0.234	2.145	0.003
	AG	0.929	0.327		
Angular jerk	AR-based Hector SLAM	1.063	0.218	2.145	0.05
	AG	1.255	0.299		
Lateral stress	AR-based Hector SLAM	0.801	0.246	2.145	0.005
	AG	1.015	0.124		
Tangential stress	AR-based Hector SLAM	0.915	0.17	2.145	0.03
	AG	1.052	0.162		

5.3.2 Scenario 2

In scenario 2, AR-based Hector SLAM decreased:

- Angular jerk by 18.71%
- Lateral stress by 14.24%

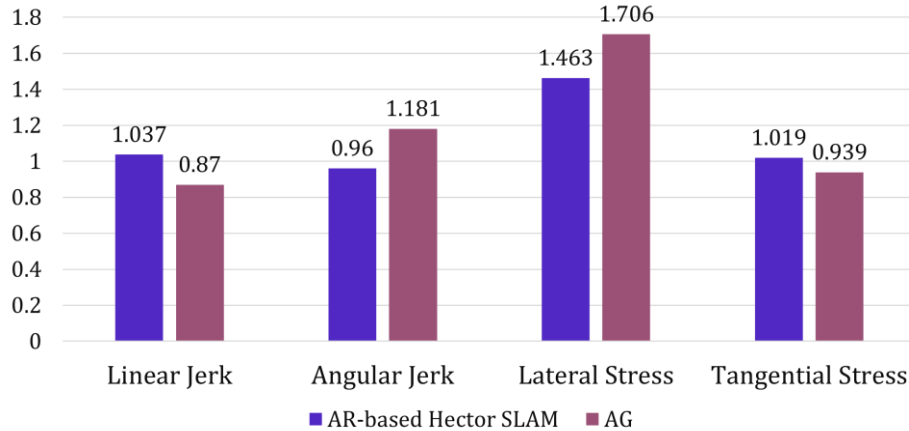


Figure 5.5: Scenario 2 mean decrease in jerk and stress

A t-test for each evaluation metric gave us the following results:

- **Linear jerk:** Comparing Hector SLAM (M=1.037, SD=0.281) and AG (M=0.877, SD=0.295) yielded a p-value of 0.13, indicating that SLAM shows an insignificant increase, as compared to AG. ($t(14) = 2.145, p = 0.13$).
- **Angular jerk:** Comparing Hector SLAM (M=0.96, SD=0.236) and AG (M=1.181, SD=0.303) yielded a p-value of 0.03, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145, p = 0.03$).
- **Lateral stress:** Comparing Hector SLAM (M=1.463, SD=0.292) and AG (M=1.706, SD=0.169) yielded a p-value of 0.009, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145, p = 0.009$).
- **Tangential stress:** Comparing Hector SLAM (M=1.019, SD=0.371) and AG (M=0.939, SD=0.09) yielded a p-value of 0.42, indicating that SLAM shows an insignificant increase, as compared to AG. ($t(14) = 2.145, p = 0.42$).

5.3.3 Scenario 3

In S3, AR-based Hector SLAM decreased:

- Linear jerk by 8.94%
- Angular jerk by 10.47%
- Lateral stress by 17.31%
- Tangential stress by 11.22%

Table 3: Scenario 2 t-test

Evaluation Metric	Method	Mean	SD	t-value	p-value
Linear jerk	AR-based Hector SLAM	1.037	0.281	2.145	0.13
	AG	0.877	0.295		
Angular jerk	AR-based Hector SLAM	0.96	0.236	2.145	0.03
	AG	1.181	0.303		
Lateral stress	AR-based Hector SLAM	1.463	0.292	2.145	0.009
	AG	1.706	0.169		
Tangential stress	AR-based Hector SLAM	1.019	0.371	2.145	0.42
	AG	0.939	0.09		

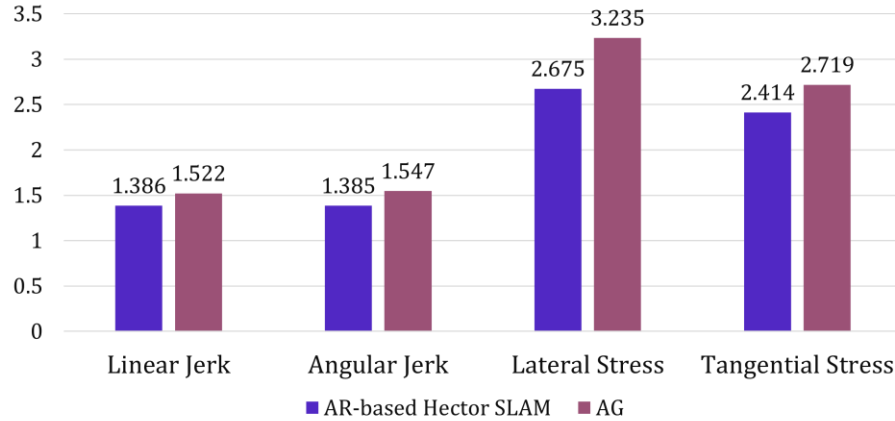


Figure 5.6: Scenario 3 mean decrease in jerk and stress

A t-test for each evaluation metric gave us the following results:

- **Linear jerk:** Comparing Hector SLAM (M=1.386, SD=0.131) and AG (M=1.522, SD=0.125) yielded a p-value of 0.007, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.007$).
- **Angular jerk:** Comparing Hector SLAM (M=1.385, SD=0.24) and AG (M=1.547, SD=0.201) yielded a p-value of 0.05, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.03$).
- **Lateral stress:** Comparing Hector SLAM (M=2.675, SD=0.677) and AG (M=3.235, SD=0.749) yielded a p-value of 0.04, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.04$).
- **Tangential stress:** Comparing Hector SLAM (M=2.414, SD=0.295) and AG (M=2.719, SD=0.507) yielded a p-value of 0.05, indicating that SLAM shows a significant decrease, as compared to AG. ($t(14) = 2.145$, $p = 0.05$).

5.3.4 Overall average improvement

Figure 5.7 shows average improvement of AR-based Hector SLAM over AG navigation for J_{acc} , ζ_{acc} , S_{lat} , and S_{tng} .

Table 4: Scenario 3 t-test

Evaluation Metric	Method	Mean	SD	t-value	p-value
Linear jerk	AR-based Hector SLAM	1.386	0.131	2.145	0.007
	AG	1.522	0.125		
Angular jerk	AR-based Hector SLAM	1.385	0.24	2.145	0.05
	AG	1.547	0.201		
Lateral stress	AR-based Hector SLAM	2.675	0.677	2.145	0.04
	AG	3.235	0.749		
Tangential stress	AR-based Hector SLAM	2.414	0.295	2.145	0.05
	AG	2.719	0.507		

If we look at the combined decrease for jerk and stress, we get improvement in:

- Average jerk (combined) by: 11.63%
- Average stress (combined) by: 11.39%

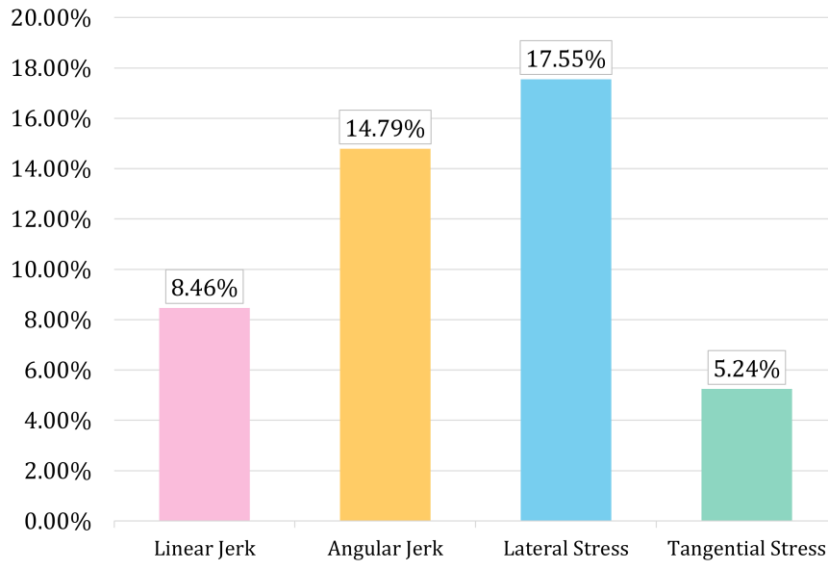


Figure 5.7: AR-based Hector SLAM average decrease in jerk and stress

5.4 Summary

In this chapter, we presented a detailed analysis for both our simulation experiments as well as the results obtained from our proposed method. AR-based Hector SLAM has shown significant improvement in mobile robot navigation over AG navigation. It has reduced jerk and stress which leads to safer and smoother robot motion.

CHAPTER 6. CONCLUSION

This research contributes to the field of HRI, AR, and AR-based motion planning. Environments where robots and humans work together are becoming increasingly popular. AR interfaces are a new mode of communication that allow intuitive and immersive experiences. AR-based robotics is an active research area, leading to improved human-robot collaboration and interactions. Autonomous robots should be able to navigate to target locations while avoiding obstacles in their path. The first step is to establish colocalization between the AR-HMD and the mobile robot. Current systems use fiducial markers, spatial anchors, or object features for colocalization. However, fiducial markers can be intrusive and require precise placement, spatial anchors may not be flexible in dynamic environments, and object features can be unreliable in cluttered or poorly lit areas. An alternative approach, using mathematical transform representation, offers a more robust solution. This method leverages precise geometric transformations, providing accurate and consistent colocalization across varying conditions and environments.

AG navigation is the current state-of-the-art AR-based motion planning method; however, it has certain limitations. It is a reactive collision avoidance method that focuses on real-time obstacle avoidance. It uses sensory information to detect obstacles and navigates around them to get to the goal point. The drawback of this method is that it does not retain environmental information such as the location of obstacles or create a map of the environment. It cannot plan a path from the initial to the goal position, decreasing its flexibility. Its reactive nature causes abrupt changes in acceleration when it detects obstacles, which leads to high values of jerk and stress. This can be unsafe, especially in situations where smooth robot motion is required.

In contrast, SLAM creates a map of the environment in real time. It tracks the location of obstacles, improving localization accuracy. It can plan paths to the target location which gives smooth trajectories. Moreover, it can replan its path if the original path is blocked, saving time that would otherwise be spent searching for a new path. It makes the system robust and adaptable to changes in the environment.

Chapter 1 laid the groundwork by introducing the motivation behind this study, presenting the problem statement, and outlining the research objectives. It highlighted the limitations of existing gap-based navigation techniques and proposed an AR-based Hector SLAM approach for smooth motion planning of indoor mobile robots to address these challenges. Chapter 2 provided a comprehensive literature review, exploring various colocalization methods and AR-based motion planning algorithms. Chapter 3 discussed the methodology, highlighting markerless colocalization between an AR-HMD and a mobile robot, the AR-ROS communication framework, and the integration of Hector SLAM with EKF sensor fusion. Chapter 4 presented the experimental results, in three test scenarios with both static and dynamic obstacles. Chapter 5 presents a detailed analysis, where we compared AR-based Hector SLAM with AG navigation, showing the reduced jerk and stress. We used a t-test to establish the significance of our results.

6.1 Contributions

The main contributions of this research are:

- **Markerless Colocalization Method:** Developed a robust markerless colocalization technique for aligning AR-HMDs with mobile robots, enhancing the flexibility and reliability of HRI.
- **AR-Based Hector SLAM:** Introduced an AR-based Hector SLAM method for smooth motion planning of indoor mobile robots.
- **AR-ROS Communication Framework:** Created an effective communication framework between AR devices and the mobile robot operating ROS.
- **Experimental Validation:** Experimental results showed the effectiveness of Hector SLAM over AG. AR-based Hector SLAM improved jerk and stress by 11.63% and 11.39% respectively. We validated these results with a t-test that showed significant change. Moreover, we implemented our proposed method for indoor mobile robot navigation.

6.2 Future Work

We propose visual SLAM methods for better localization accuracy. Integrating visual SLAM with the existing system could improve the robot's ability to navigate in complex and dynamic environments, especially where LiDAR data alone might be insufficient.

6.3 Research Publication

The research carried out during this thesis has resulted in the following publication:

I. Naveed, K.F. Iqbal, Y. Ayaz, S. Ali, K. Faraz, K.A. Ahmed, "Integration of Augmented Reality and Hector SLAM for Smooth Motion Planning of Indoor Mobile Robots", Knowledge-Based Systems. (submitted) (Q1, IF 7.2)

CHAPTER 7. REFERENCES

- [1] A. Bolu and O. Korcak, “Adaptive Task Planning for Multi-Robot Smart Warehouse,” *IEEE Access*, vol. 9, pp. 27346–27358, 2021, doi: 10.1109/ACCESS.2021.3058190.
- [2] J. Hu, P. Bhowmick, and A. Lanzon, “Group Coordinated Control of Networked Mobile Robots With Applications to Object Transportation,” *IEEE Trans Veh Technol*, vol. 70, no. 8, pp. 8269–8274, Aug. 2021, doi: 10.1109/TVT.2021.3093157.
- [3] Agility Robotics, “Advanced automation with humanoid robots.” Accessed: Aug. 12, 2024. [Online]. Available: <https://agilityrobotics.com/industries/retail-ecommerce>
- [4] A. Zemmar, A. M. Lozano, and B. J. Nelson, “The rise of robots in surgical environments during COVID-19,” *Nat Mach Intell*, vol. 2, no. 10, pp. 566–572, Oct. 2020, doi: 10.1038/s42256-020-00238-2.
- [5] Intuitive, “Da Vinci .” Accessed: Aug. 12, 2024. [Online]. Available: <https://www.intuitive.com/en-us>
- [6] J. Delmerico *et al.*, “Spatial Computing and Intuitive Interaction: Bringing Mixed Reality and Robotics Together,” *IEEE Robot Autom Mag*, vol. 29, no. 1, pp. 45–57, Mar. 2022, doi: 10.1109/MRA.2021.3138384.
- [7] P. Milgram and K. Fumio, “A taxonomy of mixed reality visual displays,” *IEICE Trans Inf Syst*, 1994.
- [8] F. De Pace, F. Manuri, A. Sanna, and C. Fornaro, “A systematic review of Augmented Reality interfaces for collaborative industrial robots,” *Comput Ind Eng*, vol. 149, p. 106806, Nov. 2020, doi: 10.1016/j.cie.2020.106806.

- [9] Y. Ghasemi, H. Jeong, S. H. Choi, K.-B. Park, and J. Y. Lee, “Deep learning-based object detection in augmented reality: A systematic review,” *Comput Ind*, vol. 139, p. 103661, Aug. 2022, doi: 10.1016/j.compind.2022.103661.
- [10] J. Ng and T. Bräunl, “Performance Comparison of Bug Navigation Algorithms,” *J Intell Robot Syst*, vol. 50, no. 1, pp. 73–84, 2007, doi: 10.1007/s10846-007-9157-6.
- [11] M. Mujahed, D. Fischer, and B. Mertsching, “Admissible gap navigation: A new collision avoidance approach,” *Rob Auton Syst*, vol. 103, pp. 93–110, May 2018, doi: 10.1016/j.robot.2018.02.008.
- [12] Magic Leap, “Magic Leap 2.” Accessed: Aug. 14, 2024. [Online]. Available: <https://www.magicleap.com/magic-leap-2>
- [13] M. A. Goodrich and A. C. Schultz, “Human-Robot Interaction: A Survey,” *Foundations and Trends® in Human-Computer Interaction*, vol. 1, no. 3, pp. 203–275, 2007, doi: 10.1561/11000000005.
- [14] J. Fu *et al.*, “Recent Advancements in Augmented Reality for Robotic Applications: A Survey,” *Actuators*, vol. 12, no. 8, p. 323, Aug. 2023, doi: 10.3390/act12080323.
- [15] R. Suzuki, A. Karim, T. Xia, H. Hedayati, and N. Marquardt, “Augmented Reality and Robotics: A Survey and Taxonomy for AR-enhanced Human-Robot Interaction and Robotic Interfaces,” in *CHI Conference on Human Factors in Computing Systems*, New York, NY, USA: ACM, Apr. 2022, pp. 1–33. doi: 10.1145/3491102.3517719.
- [16] M. Zolotas and Y. Demiris, “Towards Explainable Shared Control using Augmented Reality,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Nov. 2019, pp. 3020–3026. doi: 10.1109/IROS40897.2019.8968117.
- [17] M. Kalaitzakis, B. Cain, S. Carroll, A. Ambrosi, C. Whitehead, and N. Vitzilaios, “Fiducial Markers for Pose Estimation,” *J Intell Robot Syst*, vol. 101, no. 4, p. 71, 2021, doi: 10.1007/s10846-020-01307-9.

- [18] Microsoft Hololens, “Embed a link in a QR code to easily open a guide in Dynamics 365 Guides.” Accessed: Aug. 14, 2024. [Online]. Available: <https://learn.microsoft.com/en-us/dynamics365/mixed-reality/guides/pc-app-anchor-embed-qr-code-link>
- [19] K. Chandan, V. Kudalkar, X. Li, and S. Zhang, “ARROCH: Augmented Reality for Robots Collaborating with a Human,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2021, pp. 3787–3793. doi: 10.1109/ICRA48506.2021.9561144.
- [20] K. C. Hoang, W. P. Chan, S. Lay, A. Cosgun, and E. A. Croft, “ARviz: An Augmented Reality-Enabled Visualization Platform for ROS Applications,” *IEEE Robot Autom Mag*, vol. 29, no. 1, pp. 58–67, 2022, doi: 10.1109/MRA.2021.3135760.
- [21] R. S. Lunding, M. N. Lystbæk, T. Feuchtner, and K. Grønbaek, “AR-supported Human-Robot Collaboration: Facilitating Workspace Awareness and Parallelized Assembly Tasks,” in *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, IEEE, 2023, pp. 1064–1073. doi: 10.1109/ISMAR59233.2023.00123.
- [22] M. Żelechowski, M. Karnam, B. Faludi, N. Gerig, G. Rauter, and P. C. Cattin, “Patient positioning by visualising surgical robot rotational workspace in augmented reality,” *Comput Methods Biomech Biomed Eng Imaging Vis*, vol. 10, no. 4, pp. 451–457, 2022, doi: 10.1080/21681163.2021.2002192.
- [23] J. Minguez and L. Montano, “Nearness Diagram (ND) Navigation: Collision Avoidance in Troublesome Scenarios,” *IEEE Transactions on Robotics and Automation*, vol. 20, no. 1, pp. 45–59, Feb. 2004, doi: 10.1109/TRA.2003.820849.
- [24] M. Mujahed, D. Fischer, and B. Mertsching, “Tangential Gap Flow (TGF) navigation: A new reactive obstacle avoidance approach for highly cluttered environments,” *Rob Auton Syst*, vol. 84, pp. 15–30, Oct. 2016, doi: 10.1016/j.robot.2016.07.001.

- [25] M. Mujahed, H. Jaddu, D. Fischer, and B. Mertsching, “Tangential Closest Gap based (TCG) reactive obstacle avoidance navigation for cluttered environments,” in *2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, IEEE, Oct. 2013, pp. 1–6. doi: 10.1109/SSRR.2013.6719343.
- [26] V. Sezer and M. Gokasan, “A novel obstacle avoidance algorithm: ‘Follow the Gap Method,’” *Rob Auton Syst*, vol. 60, no. 9, pp. 1123–1134, Sep. 2012, doi: 10.1016/j.robot.2012.05.021.
- [27] D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robot Autom Mag*, vol. 4, no. 1, pp. 23–33, Mar. 1997, doi: 10.1109/100.580977.
- [28] A. Ozdemir and V. Sezer, “A Hybrid Obstacle Avoidance Method: Follow the Gap with Dynamic Window Approach,” in *2017 First IEEE International Conference on Robotic Computing (IRC)*, IEEE, Apr. 2017, pp. 257–262. doi: 10.1109/IRC.2017.25.
- [29] M. Z. Butt, N. Nasir, and R. B. A. Rashid, “A review of perception sensors, techniques, and hardware architectures for autonomous low-altitude UAVs in non-cooperative local obstacle avoidance,” *Rob Auton Syst*, vol. 173, p. 104629, Mar. 2024, doi: 10.1016/j.robot.2024.104629.
- [30] R. Chacon-Quesada and Y. Demiris, “Augmented Reality Controlled Smart Wheelchair Using Dynamic Signifiers for Affordance Representation,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Nov. 2019, pp. 4812–4818. doi: 10.1109/IROS40897.2019.8968290.
- [31] Microsoft Corporation, “Microsoft hololens.” [Online]. Available: <https://www.microsoft.com/en-us/hololens>.
- [32] E. Olson, “AprilTag: A robust and flexible visual fiducial system,” in *2011 IEEE International Conference on Robotics and Automation*, IEEE, 2011, pp. 3400–3407. doi: 10.1109/ICRA.2011.5979561.

- [33] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognit*, vol. 47, no. 6, pp. 2280–2292, 2014, doi: 10.1016/j.patcog.2014.01.005.
- [34] M. Fiala, “ARTag, a Fiducial Marker System Using Digital Techniques,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, IEEE, pp. 590–596. doi: 10.1109/CVPR.2005.74.
- [35] R. M. Claro, D. B. Silva, and A. M. Pinto, “ArTuga: A novel multimodal fiducial marker for aerial robotics,” *Rob Auton Syst*, vol. 163, p. 104398, May 2023, doi: 10.1016/j.robot.2023.104398.
- [36] Y. Cai, Y. Wang, and M. Burnett, “Using augmented reality to build digital twin for reconfigurable additive manufacturing system,” *J Manuf Syst*, vol. 56, pp. 598–604, Jul. 2020, doi: 10.1016/j.jmsy.2020.04.005.
- [37] A. Corotan and J. J. Z. Irgen-Gioro, “An Indoor Navigation Robot Using Augmented Reality,” in *2019 5th International Conference on Control, Automation and Robotics (ICCAR)*, IEEE, 2019, pp. 111–116. doi: 10.1109/ICCAR.2019.8813348.
- [38] M. Zolotas, J. Elsdon, and Y. Demiris, “Head-Mounted Augmented Reality for Explainable Robotic Wheelchair Assistance,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, Oct. 2018, pp. 1823–1829. doi: 10.1109/IROS.2018.8594002.
- [39] K. Chandan, V. Kudalkar, X. Li, and S. Zhang, “ARROCH: Augmented Reality for Robots Collaborating with a Human,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, May 2021, pp. 3787–3793. doi: 10.1109/ICRA48506.2021.9561144.
- [40] Apple Developer, “ARKit.” Accessed: Aug. 04, 2024. [Online]. Available: <https://developer.apple.com/augmented-reality/arkit/>

- [41] Microsoft, “Azure spatial anchors.” Accessed: Aug. 04, 2024. [Online]. Available: <https://azure.microsoft.com/en-us/products/spatial-anchors>
- [42] Google, “Arcore.” Accessed: Aug. 04, 2024. [Online]. Available: <https://developers.google.com/ar>
- [43] M. Walker, H. Hedayati, J. Lee, and D. Szafir, “Communicating Robot Motion Intent with Augmented Reality,” in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, New York, NY, USA: ACM, 2018, pp. 316–324. doi: 10.1145/3171221.3171253.
- [44] L. Kastner and J. Lambrecht, “Augmented-Reality-Based Visualization of Navigation Data of Mobile Robots on the Microsoft Hololens - Possibilities and Limitations,” in *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, IEEE, Nov. 2019, pp. 344–349. doi: 10.1109/CIS-RAM47153.2019.9095836.
- [45] M. E. Walker, H. Hedayati, and D. Szafir, “Robot Teleoperation with Augmented Reality Virtual Surrogates,” in *2019 14th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, IEEE, 2019, pp. 202–210. doi: 10.1109/HRI.2019.8673306.
- [46] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numer Math (Heidelb)*, vol. 1, no. 1, pp. 269–271, 1959, doi: 10.1007/BF01386390.
- [47] J. Yao, C. Lin, X. Xie, A. J. Wang, and C.-C. Hung, “Path Planning for Virtual Human Motion Using Improved A* Star Algorithm,” in *2010 Seventh International Conference on Information Technology: New Generations*, IEEE, 2010, pp. 1154–1158. doi: 10.1109/ITNG.2010.53.
- [48] D. Ferguson and A. Stentz, “Using interpolation to improve path planning: The Field D* algorithm,” *J Field Robot*, vol. 23, no. 2, pp. 79–101, 2006, doi: 10.1002/rob.20109.

- [49] I. Ulrich and J. Borenstein, “VFH/sup*: local obstacle avoidance with look-ahead verification,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, IEEE, pp. 2505–2511. doi: 10.1109/ROBOT.2000.846405.
- [50] M. Elgendy, M. Herperger, T. Guzsvinecz, and C. S. Lanyi, “Indoor Navigation for People with Visual Impairment using Augmented Reality Markers,” in *2019 10th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, IEEE, 2019, pp. 425–430. doi: 10.1109/CogInfoCom47531.2019.9089960.
- [51] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, “Anytime Motion Planning using the RRT*,” in *2011 IEEE International Conference on Robotics and Automation*, IEEE, May 2011, pp. 1478–1483. doi: 10.1109/ICRA.2011.5980479.
- [52] C. Papachristos and K. Alexis, “Augmented reality-enhanced structural inspection using aerial robots,” in *2016 IEEE International Symposium on Intelligent Control (ISIC)*, IEEE, 2016, pp. 1–6. doi: 10.1109/ISIC.2016.7579983.
- [53] R. T. Chadalavada, H. Andreasson, M. Schindler, R. Palm, and A. J. Lilienthal, “Bi-directional navigation intent communication using spatial augmented reality and eye-tracking glasses for improved safety in human–robot interaction,” *Robot Comput Integr Manuf*, vol. 61, p. 101830, 2020, doi: 10.1016/j.rcim.2019.101830.
- [54] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, “Probabilistic roadmaps for path planning in high-dimensional configuration spaces,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996, doi: 10.1109/70.508439.
- [55] S.-O. Park, M. C. Lee, and J. Kim, “Trajectory Planning with Collision Avoidance for Redundant Robots Using Jacobian and Artificial Potential Field-based Real-time Inverse Kinematics,” *Int J Control Autom Syst*, vol. 18, no. 8, pp. 2095–2107, 2020, doi: 10.1007/s12555-019-0076-7.

- [56] O. Khatib, “Real-Time Obstacle Avoidance for Manipulators and Mobile Robots,” *Int J Rob Res*, vol. 5, no. 1, pp. 90–98, Mar. 1986, doi: 10.1177/027836498600500106.
- [57] ROS.org, “Rosbridge suite.” Accessed: Jul. 30, 2024. [Online]. Available: Retrieved from http://wiki.ros.org/rosbridge_suite
- [58] Magic Leap, “Magic Leap 1.” Accessed: Aug. 05, 2024. [Online]. Available: <https://ml1-developer.magicleap.com/en-us/home>
- [59] Magic Leap, “Magic Leap Unity API Documentation (Version 0.26.0.” Accessed: Aug. 04, 2024. [Online]. Available: <https://ml1-developer.magicleap.com/en-us/learn/guides/unity-web rtc-guide>
- [60] S. Kohlbrecher, O. von Stryk, J. Meyer, and U. Klingauf, “A flexible and scalable SLAM system with full 3D motion estimation,” in *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, IEEE, Nov. 2011, pp. 155–160. doi: 10.1109/SSRR.2011.6106777.
- [61] M.-A. Chung and C.-W. Lin, “An Improved Localization of Mobile Robotic System Based on AMCL Algorithm,” *IEEE Sens J*, vol. 22, no. 1, pp. 900–908, Jan. 2022, doi: 10.1109/JSEN.2021.3126605.
- [62] S. Thrun, “Probabilistic robotics.,” *Commun ACM*, pp. 52–57, 2002.
- [63] ROS.org, “global_planner.” Accessed: Aug. 04, 2024. [Online]. Available: http://wiki.ros.org/global_planner
- [64] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, Third. MIT press, 2009.
- [65] B. P. Gerkey and Kurt Konolige, “Planning and control in unstructured terrain,” in *ICRA workshop on path planning on costmaps*, 2008.

- [66] ROS.org, “base_local_planner.” Accessed: Aug. 04, 2024. [Online]. Available: http://wiki.ros.org/base_local_planner
- [67] ROBOTIS, “TurtleBot3 e-Manual.” Accessed: Aug. 04, 2024. [Online]. Available: <https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/>