A 360 VR and Wi-Fi Tracking Based Autonomous Telepresence Robot for Virtual Tour

Yeonju Oh
Purdue University

Follow this and additional works at: https://docs.lib.purdue.edu/open_access_theses

Recommended Citation
https://docs.lib.purdue.edu/open_access_theses/1493

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.
A 360 VR AND WI-FI TRACKING BASED AUTONOMOUS TELEPRESENCE ROBOT FOR VIRTUAL TOUR

by

Yeonju Oh

A Thesis
Submitted to the Faculty of Purdue University
In Partial Fulfillment of the Requirements for the Degree of
Master of Science

Department of Computer and Information Technology

West Lafayette, Indiana
May 2018
THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF THESIS APPROVAL

Dr. Byung-Cheol Min, Chair
Department of Computer and Information Technology

Dr. Eric T. Matson,
Department of Computer and Information Technology

Dr. John A. Springer,
Department of Computer and Information Technology

Approved by:

Dr. Eric T. Matson
Head of the Departmental Graduate Program
LIST OF TABLES ................................................................. v
LIST OF FIGURES ............................................................. vi
LIST OF ABBREVIATIONS ................................................... viii
ABSTRACT ................................................................. ix

CHAPTER 1 INTRODUCTION ............................................. 1
  1.1 Statement of Problem ................................................. 1
  1.2 Scope ........................................................................... 2
  1.3 Significance ............................................................... 3
  1.4 Research Question ..................................................... 3
  1.5 Assumptions ............................................................. 4
  1.6 Limitations .............................................................. 4
  1.7 Delimitations ............................................................ 4
  1.8 Summary ................................................................. 5

CHAPTER 2 REVIEW OF RELEVANT LITERATURE .................. 6
  2.1 Human-Robot Interaction .............................................. 6
    2.1.1 Remote Interaction ............................................... 6
    2.1.2 Proximate Interaction ............................................. 8
  2.2 Remote Systems and Control Input Methods ..................... 9
    2.2.1 Conventional Control Input Methods ......................... 9
    2.2.2 Natural Interaction Based Control ............................. 10
    2.2.3 Control with Mixed Reality Interface ....................... 11
  2.3 Wireless Signal Tracking ............................................. 12
  2.4 Telepresence Robot Systems with VR .............................. 13
  2.5 Path Planning for Mobile Robots .................................... 14
  2.6 Summary ............................................................... 16

CHAPTER 3 ROBOTIC PLATFORM DEVELOPMENT .................... 17
  3.1 Approach and System Design ....................................... 17
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Used ROS topics</td>
<td>26</td>
</tr>
<tr>
<td>3.2</td>
<td>Paths generated from the simulation shown in Figure 3.8(b)</td>
<td>28</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Usage scenario for the proposed VR-based telepresence robot system</td>
<td>2</td>
</tr>
<tr>
<td>2.1</td>
<td>Streamer headgear with a camera (Kasahara, Nagai, &amp; Rekimoto, 2017)</td>
<td>7</td>
</tr>
<tr>
<td>2.2</td>
<td>Left: the light array visor; middle: the usage of the helmet; right: graphical user interface for robot swarms (Gancet et al., 2010)</td>
<td>8</td>
</tr>
<tr>
<td>2.3</td>
<td>Examples of remote control systems.</td>
<td>12</td>
</tr>
<tr>
<td>2.4</td>
<td>VR interface for controlling a robot arm (Jankowski &amp; Grabowski, 2015)</td>
<td>14</td>
</tr>
<tr>
<td>3.1</td>
<td>An example scenario where the telepresence robot moves to its destination by tracking a series of APs in order. The path is generated as AP1-AP2-AP5-AP6.</td>
<td>17</td>
</tr>
<tr>
<td>3.2</td>
<td>Illustration of multi-path effects</td>
<td>20</td>
</tr>
<tr>
<td>3.3</td>
<td>Examples of RSS measurements.</td>
<td>22</td>
</tr>
<tr>
<td>3.4</td>
<td>Flow chart of DOA estimation process.</td>
<td>22</td>
</tr>
<tr>
<td>3.5</td>
<td>An example of using sensor fusion for obstacle avoidance.</td>
<td>23</td>
</tr>
<tr>
<td>3.6</td>
<td>The developed telepresence mobile robot and its components.</td>
<td>25</td>
</tr>
<tr>
<td>3.7</td>
<td>Flow chart of the overall program logic.</td>
<td>26</td>
</tr>
<tr>
<td>3.8</td>
<td>Path planning simulations. The red line depicts a direct path from starting node to destination node, but this route may be blocked by walls. The green line indicates a sub-optimum path generated by the method proposed in Figure 3.1.</td>
<td>28</td>
</tr>
<tr>
<td>3.9</td>
<td>Structure of the experimental environment.</td>
<td>29</td>
</tr>
<tr>
<td>3.10</td>
<td>DOA estimation results at 60°</td>
<td>30</td>
</tr>
<tr>
<td>3.11</td>
<td>DOA estimation results at 90°</td>
<td>30</td>
</tr>
<tr>
<td>3.12</td>
<td>DOA estimation results at 120°</td>
<td>31</td>
</tr>
<tr>
<td>3.13</td>
<td>Sensor fusion results.</td>
<td>32</td>
</tr>
<tr>
<td>4.1</td>
<td>System overview</td>
<td>34</td>
</tr>
<tr>
<td>4.2</td>
<td>Examples of existing virtual tours.</td>
<td>35</td>
</tr>
<tr>
<td>4.3</td>
<td>A user with the VR interface (left), the mobile robot with a 360 video camera (right)</td>
<td>36</td>
</tr>
</tbody>
</table>
4.4 Rendering objects in a fixed location while the head is moving (Unity - Interaction in VR, n.d.). The pop-up window shown at the lower right is the user’s view through the HMD. ................................................................. 36
4.5 Image shader for spherical object. ...................................................... 37
4.6 Rendering process for the 360-degree video stream. ......................... 38
4.7 Screen capture in HMD; left = left eye view, right = right eye view .... 39
4.8 User interface elements. ................................................................. 39
4.9 Comparison of the angular velocities of a human head and a servo motor . 40
4.10 Navigation experiment design. .......................................................... 42
4.11 Navigation test results. ................................................................. 43
4.12 Screen capture locations ............................................................... 44
4.13 HMD captures at each designated location ................................. 45
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HRI</td>
<td>Human Robot Interaction</td>
</tr>
<tr>
<td>HCI</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>VR</td>
<td>Virtual Reality</td>
</tr>
<tr>
<td>HMD</td>
<td>Head Mounted Display</td>
</tr>
<tr>
<td>FOV</td>
<td>Field of View</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>RSS</td>
<td>Radio Signal Strength</td>
</tr>
<tr>
<td>RSSI</td>
<td>Radio Signal Strength Indicator</td>
</tr>
<tr>
<td>ROS</td>
<td>Robot Operating System</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio-Frequency Identification</td>
</tr>
<tr>
<td>DOA</td>
<td>Direction of Arrival</td>
</tr>
<tr>
<td>AP</td>
<td>Access Point</td>
</tr>
</tbody>
</table>
This study proposes a novel mobile robot teleoperation interface that demonstrates the applicability of a robot-aided remote telepresence system with a virtual reality (VR) device to a virtual tour scenario. To improve realism and provide an intuitive replica of the remote environment for the user interface, the implemented system automatically moves a mobile robot (viewpoint) while displaying a 360-degree live video streamed from the robot to a VR device (Oculus Rift). Upon the user choosing a destination location from a given set of options, the robot generates a route based on a shortest path graph and travels along that the route using a wireless signal tracking method that depends on measuring the direction of arrival (DOA) of radio signals. This paper presents an overview of the system and architecture, and discusses its implementation aspects. Experimental results show that the proposed system is able to move to the destination stably using the signal tracking method, and that at the same time, the user can remotely control the robot through the VR interface.
CHAPTER 1. INTRODUCTION

This thesis focuses on expanding the possible range of a teleoperation in indoor space using a virtual reality (VR) interface. This chapter will give an overview of the research to be conducted, its scope, and its background.

1.1 Statement of Problem

Virtual reality (VR) has a myriad of potential application areas outside the entertainment industry, such as in tourism (YouVisit, 2018), industrial design (Costa et al., 2016), architecture, real estate, medical care (Diana & Marescaux, 2015; Egger et al., 2017), and education (Santana, Ferre, Izaguirre, Aracil, & Hernandez, 2013). Among these applications, we are interested in the use of VR in virtual tour scenarios. A virtual tour is more than just media content, in that it can provide users with vivid experiences and environments similar to reality. For instance, in the 2017 Google I/O conference, Google introduced the motif of educational VR content using smartphones, with the main feature being that students could watch 360-degree field trip videos to visualize firsthand what they read about in a book. In pilot studies, students engaged in more immersive, interactive, and social activities to obtain enhanced learning experiences.

Currently, such virtual tour applications have limitations: a fixed viewer’s location, cumbersome operation, and displaying only prerecorded media (video or panorama images) (Famukhit, Yulianto, & Maryono, 2013; J. Lee, Lu, Xu, & Song, 2016; Wessels, Ruther, Bhurtha, & Schroeder, 2014).

To mitigate these limitations and realize a true immersive remote tour experience, this study proposes a system exploiting a VR display that produces a (near real-time) stream of a live 360-degree camera from a moving agent, specifically a mobile robot. Figure 1.1 illustrates an example scenario employing the proposed system. The VR display shows a general interface so that the user can visualize the robot’s 3D view and also navigate the robot through the remote environment (and to change the viewpoint).
1.2 Scope

The research aims to design an effective VR interface that consists of VR controllers, a robot equipped with a 360-degree camera, and built-in sensors on the robot, such as sonar. At this time, the user and the robot communicate with each other on a one-to-one basis, and a user is limited to remote control of one robot.

The main feature of the user interface renders video captured by the robot’s 360-degree camera. Therefore, most of the interface consists of stitching together a 360-degree video frame to build and play this video stream.

A mobile robot is an agent of the system that receives and handles commands, which is mainly the destination requested by the user. For ease of navigation, the robot should have wheels or be movable. Additionally, a directional antenna is required because navigation is carried out by way of tracking indoor radio signals. The robot automatically creates a path with reference to several options that the user can select. The feasibility of
path generation was demonstrated through simulations, which utilized randomly generated graphs to test the contingency of having numerous access points (APs) in the crawl space.

The primary tasks of the robot are limited to the functions of the teleoperation, such as streaming 360-degree video to the client, and to tracking the generated path that consists of the series of APs.

1.3 Significance

Recent years have witnessed active research in robot control using VR (Chen et al., 2017), particularly immersive teleoperation systems using head-mounted displays (HMDs) (García et al., 2017; Kot & Novák, 2014). Telepresence via the VR interface has shown to be easier and more intuitive than interfaces using liquid crystal displays and joysticks (Bug, 2014; Jankowski & Grabowski, 2015).

Taking inspiration from recent research, this study proposes a VR interface that provides an immersive experience by incorporating imagery from a 360-degree camera on a mobile robot. Additionally, we design an autonomous navigation system based upon a wireless signal tracking method (Min, Matson, & Jung, 2016; Min, Parasuraman, Lee, Jung, & Matson, 2018). With this autonomous system minimizes the mental workload of navigating the environment; the users are expected to feel less fatigue (not having to worry about fine movement control) during the virtual tour, and to have a more relaxed and rewarding experience.

1.4 Research Question

Is it feasible to realize a robotic virtual tour system with a 360-degree based VR interface and Wi-Fi tracking? To show the feasibility, a set of tests is designed to validate main functions of the system: the mobile robot can reach within a threshold range to the destination while providing a live 360-degree video stream through a VR interface.
1.5 Assumptions

The assumptions of thesis include:

• No other disturbances of communication between the VR system and the camera module.

• Human factors cannot affect the system or its performance.

1.6 Limitations

This study is performed acknowledging the following constraints:

• Due to technical network limitation, video streaming quality may get lower and certain delay can occur.

• When rendering a 360-degree panoramic image to the HMD, the image quality may deteriorate due to differences in resolution.

1.7 Delimitations

This research includes the following delimitations:

• This study did not include human subjects in the test. The research only tested teleoperation functions, such as delivering messages to the robot through the VR interface, and path planning through simulation. Thus, in real usage scenarios, performance and effectiveness can vary based on human factors.

• The performance of streaming is affected by various factors such as network, encoding, and decoding format. This study does not test this performance in all aspects.

• This research does not test the system on other interfaces, such as other VR commercial platforms.

• Any disturbance of a network is ignored for the evaluation and testing processes.
This research will not consider the virtual reality sickness problem (a similar symptom with motion sickness) that is mainly caused by the sound effects.

1.8 Summary

This chapter provided an overall description of the research focusing on scope, significance, assumptions, limitations, and delimitations that are considered in the study. Additionally, this section presents the research question that says the key contribution of the study.
CHAPTER 2. REVIEW OF RELEVANT LITERATURE

This literature review describes background knowledge, definitions, and current issues in relevant research domains. The first section provides general introductions to two types of human-computer interaction (HCI) research. The following sections describe related research and information about the elements required to implement the overall system: remote control methods, telepresence systems with VR, and state of the art path planning methods.

2.1 Human-Robot Interaction

As robotics are increasingly applied in many fields of study or industries, research has started focusing on the subject of cooperation between humans and robots. Human-robot interactions (HRIs) have been improved by building systems for bi-directional and more natural communication. According to a survey (Goodrich & Schultz, 2007), this field can be divided into two categories: remote and proximate interaction. In a remote interaction, the robot serves its mission remotely at a distant location (e.g. Mars rovers). In contrast, in a proximate interaction, the robot acts beside the user to carry out services in nearby places. The characteristics of each interaction and recent studies concerning them are described in the following subsections.

2.1.1 Remote Interaction

Remote interaction is also called teleoperation, telepresence, or remote manipulation. A telepresence robot is a remote-controlled mobile robot that has a screen, camera, and wireless internet connectivity (What is telepresence robot? - Definition from WhatIs.com, 2016). The common objective of remote interaction studies is to conduct telemanipulating missions in a separate area. One example of a telepresence robot is used for English education (Kwon, Koo, Kim, & Kwon, 2010). The robot played the video given by the teacher to the students and transferred video of its surroundings from its
attached camera to the teacher, to monitor the students. However, the operator of the robot had a restricted field of view (FOV), defined as the extent of the observable world that is seen at any given moment, (Lee Pazuchanics, 2006). Moreover, the robot control system was divided into two parts, control and monitoring, so the teacher could not manage both students and the robot simultaneously. Another research paper focused on the design and functionality of the telepresence robot (Tsui, Norton, Brooks, Yanco, & Kontak, 2011). This pilot study showed that latency caused malfunctions when the user sent commands to control the robot. For example, the robot often moved in undesired directions and rotated at greater than the designated angle. Another challenge of telepresence systems is the narrow FOV, because media with a wider FOV (up to 360 degrees) require much stronger connections between streamer and receiver (Kasahara et al., 2017). In their study, Kasahara et al. used a wearable camera to give the user immersive experiences with both visual and audible content, as shown in Figure 2.1. They developed custom equipment for both the streamer (head gear with cameras) and the receiver (HMD and audio output) and focused on the synchronization of the streamer with the receivers head direction. Because of technical limitations, the implementation of their real-time system suffered from latency and low quality issues. While the streamer and receiver were able to successfully communicate, some participants felt virtual reality sickness, which is similar to motion sickness and depends upon communication and the latency of video streaming updates.
2.1.2 Proximate Interaction

Proximate interactions have been especially focused on social or specific missions, such as search and rescue operations. Mast et al. defined an interaction concept design specific for seniors that describes user-focused tasks and interface requirements (Mast et al., 2012). In particular, the authors focused on tasks that might be needed in indoor spaces and real life, such as cleaning the windows or floor, calling for help during an emergency, reading small letters, or providing physical assistance with carrying objects. They also conducted a focus group study surveying user interface requirements for those interactions. Cooperation of robots and humans in teams has also been mentioned in other research; for example, National Aeronautics Space Administration (NASA) formed an EVA team of humans and rovers that serves on-the-spot tasks in exploratory missions (Ferketic et al., 2006). The rovers support activities by providing information from their sensors and carrying out dangerous labor in the place of humans.

However, to control these rovers, the human needs to use voice or dedicated devices, such as a computer interface or tablet-like appliance. Because they can monitor the robot with only the display, these control devices do not allow the users precise control of the robot. This issue can be solved by the VR interface suggested in this proposed system. A team of swarm robots has been suggested for use in firefighting, helping at the scene of an incident (Gancet et al., 2010).
The biggest challenges with such a swarm robot teams are establishing communication protocols appropriate for a large number of robots and developing the interface to monitor the robots. The biggest challenges with such a swarm robot teams are establishing communication protocols appropriate for a large number of robots and developing the interface to monitor the robots. For communication, they proposed and devised an array of light that shows encoded commands and depict a safe direction for the user (Figure 2.2. The light view is attached inside a standard firefighting helmet. However, the light indicator is not a straightforward method for presenting users commands, and users have to memorize the light signals before they can understand and use them freely. In addition, there were combination effects of both proximate and remote interactions (Bruemmer, Marble, Dudenhoeffer, Anderson, & McKay, 2003; Fong, Thorpe, & Baur, 2003; Goodrich, Olsen, Crandall, & Palmer, 2001).

2.2 Remote Systems and Control Input Methods

Several control input methods have been developed for operating a computer or robot system, from the traditional joystick to motion and voice control.

2.2.1 Conventional Control Input Methods

The early versions of control systems adopted conventional input devices such as a keyboard, mouse, and joystick (Rovetta, Sala, Wen, & Togno, 1996; Shim, Jun, Lee, Baek, & Lee, 2010). These input devices give the user a familiar interface for operating the systems. For example, the user can assign custom commands on a keyboard (Shim et al., 2010)), which can increase the functionality of the system; on the other hand, doing so can also reduce intuitiveness and introduce complexity. Another study tried to increase the usability of traditional methods by considering the gap between a human and the technology (B. Lee, Isenberg, Riche, & Carpendale, 2012). The authors made a visual representation which utilized a keyboard and mouse for natural interaction, but by relying upon a traditional input device, their proposed interface did not take advantage of the new possibilities. Thus, they suggested using other technologies to enhance liberty of
communication, such as natural language processing and gesture-based cooperation. The next subsection covers suggested technologies for more natural interactions

2.2.2 Natural Interaction Based Control

Efforts for supporting effective social interaction between humans and computers include incorporating natural interactions such as gesture and speech. Motion capture is one of the most popular modes of communication being used between humans and computer systems. Moreover, because of the intuitiveness of motion control, the detection of human gestures for use as controls is an area that has been investigated steadily for decades (Baudel & Beaudouin-Lafon, 1993; Do, Jung, Jung, Jang, & Bien, 2006; Freeman & Weissman, 1995; Y. Zhang, Stellmach, Sellen, & Blake, 2015). Many researchers have tried to capture the human body using a depth or stereo camera and detect specific motions. Y. Zhang et al. suggested the consolidation of gaze tracking and hand gestures to avoid hand lethargy (Figure 2.3(a)) (Y. Zhang et al., 2015). The authors tested and compared two control modes: hand gesture only and a combination of gaze and hand gesture. Most participants preferred the combination controls even though in some phases that method required more number of experiments than the gesture only controls. When the authors asked what aspects of combination control attracted the participants, 80% of subjects cited faster recognition.

Another study focused on expressing social response and showing a facial expression depending on social response (Breazeal, 2003). The author defined several facial expressions and response logic based on Russells pleasure-arousal emotional state model (Russel, 1997), and designed the robot to distinguish the affective meaning of the user. The experiment surveyed 12 children and five adults, showing a video of the robot and letting them guess the associated emotion. The accuracy rate of the survey was 77%, which is slightly low, but this research is meaningful because the system engaged people not only to connect with the robots but also to improve their well-being. Using natural interactions comes with the important drawbacks of initial facility cost and accuracy. In order to detect specific motions, the participants to provide sample motion data. Also,
motion or facial expressions are not adequate for precise control, and modulating the degree of control is hard. Therefore, this control method can make the system more ambiguous rather than more intuitive.

2.2.3 Control with Mixed Reality Interface

Mixed reality technology can offer innovative ways to take advantage of existing traditional and natural methods and offset their disadvantages. By combining real video images and virtual 3D objects, systems could deliver more dynamic information to the user. For example, Nielsen and Goodrich described the usefulness of map information for mobile robot navigation (Nielsen & Goodrich, 2006). They conducted two different experiments: first a pilot study simulating navigation through several interfaces, then actually navigating a real robot via those interfaces. They developed a side-by-side representation that showed a 2D map overview and camera video on both sides of the screen, and a 3D interface that used generated 3D vehicle objects to represent the direction of the robot and the spectators' view. Additionally, conducted experiments with video-only and map-only interfaces for comparison. The authors found that the video-only 3D interface could not support navigation without any cues for the navigation in the video image. This highlighted the necessity of providing additional information, such as using a map overview and video together. The real control scenario also supported that video-only was significantly worse than the other interface conditions. Building upon these findings, this proposed research will actively exploit the combination of a 3D graphical interface with video. Another research shown in Figure 2.3(b) also attempted to develop an augmented reality (AR) interface for managing a laboratory (Andujar, Mejías, & Marquez, 2011). The authors focused on educational applications for engineering students who need practical experience. The virtual practical session in this paper was compared to a real lab session run by the university and found that the AR interface could resolve educational, logistical, and economic problems. AR interfaces are also actively used in manufacturing (Ong & Nee, 2013), where safety and economic costs can be critical issues for business. However, applications of the AR simulator to real
environments are constrained because the controllers still require computer interfaces. A VR controller can solve the inconveniences of existing methods, allowing users to be effectively trained through an AR system.

2.3 Wireless Signal Tracking

Wireless signal tracking is useful for indoor environments where GPS cannot be used. Communicating a lot of information through Wi-Fi signals and mapping the information off-line is costly, but the communication distance is longer than Bluetooth and the installation process is cheaper than deploying radio-frequency identification (RFID). Additionally, additional devices do not have to be deployed in a building if there are enough existing access points; thus, many research studies utilize Wi-Fi fingerprinting, Cell-ID, or triangulation (Castro, Chiu, Kremenek, & Muntz, 2001; Paul & Wan, 2008; Xiang et al., 2004).

Location estimation with only received signal strength indicator (RSSI) tracking has also been proposed (Zárubka, Huber, Kamangar, & Chlamtac, 2007); however, occasional fluctuations in the signal cause the position error to be large. Various techniques are used to filter noise and minimize fluctuation of the received signal; for
example, Záruba et al. used Kalman filtering and Monte Carlo sampling. For online fingerprinting and signal tracking, a topological map using simultaneous localization and mapping (SLAM) and Wi-Fi calibration can be used (Shin & Cha, 2010). Likewise, there have also been efforts to improve the accuracy of indoor localization and tracking by combining other sensor measurements with RSSI (Evennou & Marx, 2006; Fink & Beikirch, 2011). Fink and Beikirch proposed a hybrid localization system for human tracking in indoor space that integrates an inertial navigation system with RSS-based localization. They showed that the inertial sensor could reduce the error significantly. Therefore, the minor disadvantages of wireless tracking can be resolved by combining another sensor with the system. This current study uses an ultrasonic sensor with RSS to detect obstacles and to reduce the multi-path effect.

### 2.4 Telepresence Robot Systems with VR

Telepresence is the physical experience of a spatially separated or virtual place. By connecting a computer to a remote place over a communication line, it is possible to interact with another place that is not physically present in the virtual reality space. There have been many studies on implementing telepresence system through various means, and methods using not only voice and image but also robots have been introduced. In particular, recent research has proposed a new approach to robot control for telepresence applications through the use of VR.

Telepresence via the VR interface has shown to be easier and more intuitive than interfaces using liquid crystal displays and joysticks (Figure 2.4) (Bug, 2014; Jankowski & Grabowski, 2015). Likewise, there are a number of studies taking advantage of the head tracking function in HMDs to enable free vision. In a study using a remotely-operated underwater vehicle (ROV) (Bosch, Ridao, Garcia, & Gracias, 2016), a panoramic image was generated by joining images taken by its omnidirectional camera and the combined image rendered on a VR display; this improved the user’s spatial perception in the underwater environment. In another study (Lipton, Fay, & Rus, 2018), a VR-based teleoperation interface was developed to allow users to perform specific
assembly tasks with a robot manipulator that mimicked movements of the VR controller, which was more accurate than the conventional auto assembly used in automated processes.

This research expands upon the previously described methods to implement a teleoperated robot system with a VR-based interface that can both remotely manipulate the robot and immersively monitor its surroundings using a 360-degree camera.

2.5 Path Planning for Mobile Robots

Path planning is the problem of navigating from an origin node to a target node while avoiding obstacles and minimizing travel costs (fuel, time, distance, cost, etc.) Many studies have been conducted on A* algorithms such as IDA* and D* (Duchoň et al., 2014; D. Zhang, Chen, Huang, & Gao, 2015). In addition, methods have been proposed for locating paths in dynamic maps and for path finding in the presence of multiple start nodes and multiple target nodes (Grimani & Titelbaum, n.d.). A number of these algorithms are discussed below.

Dijkstra's algorithm starts at the origin node and expands in all directions to find the path to the target node. Although the algorithm is not efficient and cannot use negative
weights, it has the advantage of always finding the shortest path from origin node to target node (Stout, 2000).

The Floyd-Warshall algorithm finds the minimum cost route between all nodes, as opposed to Dijkstra’s algorithm that finds only the shortest path from the originating node to neighboring nodes. Its theoretical time complexity is \(O(V^3)\), which seems to be more complicated than Dijkstra’s, but in practice, the shortest path can be obtained more simply. This algorithm is more advantageous when the number of edges is much larger than the number of vertices because the time complexity is affected only by the vertices.

The greedy best-first search works similarly to Dijkstra’s algorithm, but uses heuristic estimates of the distance from an arbitrary node to a target node. In other words, the node closest to the target is selected instead of the one closest to the starting node. The best-first search algorithm has much faster performance because it uses a heuristic function that guides the path to the target, but does not guarantee the shortest path. In a map without obstacles, the shortest path is a straight line from the starting node, and this method is faster than Dijkstra’s algorithm because it searches in one direction.

Heuristic methods adopt the advantages of both the Dijkstra algorithm and the best-first search. In general, heuristic methods do not provide an approximate solution rather than the best solution. The A* algorithm is an example of a heuristic method that guarantees the shortest path. In the A* algorithm, \(g(n)\) is the cost of the path from the starting node to any node \(n\), and \(h(n)\) is a heuristic estimation of the cost from node \(n\) to the target node. The total cost is \(f(n) = g(n) + h(n)\). The A* algorithm evaluates \(g(n)\) and \(h(n)\) while moving from the starting node to the target. Since the speed of the A* algorithm is greatly dependent on the size of the search space, research has focused on minimizing that space. Memory allocation, management problems, and sorting techniques have been addressed to speed up the computational load (Higgins, 2002; Rabin, 2000). The fundamental weakness of the A* algorithm is the cost of managing lists of nodes that are scheduled to be visited and those that are not. To overcome this problem, an IDA* algorithm has been proposed that does not use open and closed lists (D. Zhang et al., 2015). However, the IDA* algorithm has the disadvantage of using depth-first search to revisit the nodes, increasing the needed memory space. Another
algorithm, Fringe, has been proposed that falls between the A* algorithm and the IDA* algorithm, but it requires even more memory to process (DeLisle, n.d.).

### 2.6 Summary

This chapter provides a review of the literature relevant to HCI and HRI, remote control methods, telepresence systems, and general path planning and shortest path finding algorithms. The introduced background information provides general knowledge of the proposed study and a foundation for system and user interface design ideas presented in the next chapter.
CHAPTER 3. ROBOTIC PLATFORM DEVELOPMENT

This chapter describes the design and implementation of a robotic platform, experiments, and their results. Each experimental procedure and its parameters will be delineated in the following sections.

3.1 Approach and System Design

The robot plays an important role as an avatar of the user. The user commands the robot to go where he wants to, and the robot creates and navigates the appropriate path to the destination. At this time, a 360-degree camera attached to the robot captures the surrounding environment for the user to view.

![Diagram](image.png)

*Figure 3.1.* An example scenario where the telepresence robot moves to its destination by tracking a series of APs in order. The path is generated as AP1-AP2-AP5-AP6.

This study uses a method of tracking APs near the location selected by the user Figure 3.1. Site survey information of the locations of nearby APs is stored in advance.
The robot navigates by signal tracking, measuring signal strength over angles within the movable range of the robot toward the destination and then moving in the direction of the strongest appropriate point. This Wi-Fi radio signal tracking method was adopted for the following reasons:

- **Cost effective:** There are several ways to navigate indoor spaces (camera, SLAM, RFID, etc.), but mobile robots usually use expensive equipment such as LIDAR, or installed infrastructure such as beacons that can be tracked inside buildings. In contrast, utilizing existing APs is inexpensive and requires no installation of new infrastructure.

- **Robust video streaming regardless of distance:** By using indoor radio signals for tracking, video streaming becomes possible even when the robot has to move farther away from the user.

- **Consistent path planning:** If the robot cannot connect directly to the destination AP, i.e. that AP is not in line of sight or is far away, path planning using indoor Wi-Fi can provide an alternative route through visiting several way points.

However, radio signals are reflected, refracted, and absorbed by walls and other objects, resulting in multi-path phenomena or shadowing (Figure 3.2). Additionally, if the desired AP is not in line of sight, the system cannot figure out the appropriate DOA for the destination, and the direction of the strongest signal will be different from that of the actual target. Consequently, in this paper, a graph-based path planning model with signal propagation is used to compensate for path loss. This model creates a sub-optimal path rather than a straight line to the destination.

The robots path planning features are integrated with the Robot Operating System (ROS) (Quigley et al., 2009), a powerful tool to simplify the development processes in robotic programming. The VR device and the robot are interconnected with a Hypertext Transfer Protocol (HTTP) server for video streaming and Rosbridge\(^1\)-based socket communications for robot control. The following sections list methods, implementations, experiments, and results in detail for the proposed system.

\(^1\)Rosbridge is a JSON API that provides a connection to non-ROS platforms.
### 3.2 Path Generation based on Access Points

Once the robot agent receives the message about the way-points, the robot figures out the shortest path of the robot and starts the navigation.

The proposed system utilizes the Floyd-Warshall algorithm to find the global shortest path that contains whole information on the shortest routes between all vertices. Notably, the predefined graph has many more edges than vertices. In terms of cache memory efficiency, it is therefore more beneficial to use a Floyd-Warshall algorithm over Dijkstra’s algorithm, which is another famous shortest path algorithm (Pradhan & Mahinthakumar, 2012). When creating a graph, each indoor AP becomes a vertex, and edges are determined by how much path loss occurs between each pair of APs. The indoor path loss model was used to decide appropriate edge costs. However, common signal propagation models have only small penalties for walls and obstacles. In this scenario, it is advantageous to limit results to paths that the robot can easily traverse, and bypassing rather than creating projected paths that penetrate walls. Therefore, penalties differ according to the number of walls. Edge costs were driven by the revisiting Hata-Okumura model (Bose & Foh, 2007) as follows:

\[
PL = PL_0 + 10n\log d - 20\log \lambda + 20\log(4\pi) + \alpha m \quad (3.1)
\]

where \(PL_0\) is path loss at the reference distance, \(n\) is a measure of the influence of obstacles on path loss, \(d\) is the distance between APs, \(\lambda\) is the wavelength of the signal estimated 0.12m. And, \(\alpha\) is the experimentally determined penalty. Constant \(m\) is the number of walls through which a straight line between two APs must pass.

Figure 3.1 provides an illustration of the path planning algorithm. The original shortest path from AP1 to AP6 (shown in red dotted lines) cannot be used because the path is blocked by walls; thus, the robot would instead move in blue dotted paths, tracking AP6 while avoiding the walls. Obviously, these blue paths are undesirable for the virtual tour scenario. Moreover, if AP6 is too far away from the starting point, the robot cannot detect AP6 and generate a path to the destination. Therefore, a sub-optimum path is created via
other APs (shown in green dotted lines). A graph generated by Equation 3.1 enables the generation of more feasible paths that are less prone to obstacles and physical blockage.

3.3 Radio Signal Tracking

Multiple APs are pre-installed throughout the environment, which can serve as a way point or as the destination for robots tracking, as illustrated in Figure 3.1. This approach is cost-effective compared to using beacons or RFID infrastructure, particularly as typical Wi-Fi APs can be utilized. While measuring RSS, several issues can affect finding a proper angle in the scanning range. This section describes the sub-problems that need to be addressed in order to figure out the right DOA. The proposed system could determine DOA through a combination of the following methods: moving window average, probabilistic filtering, and ultrasonic sensor fusion.

3.3.1 Probabilistic RSS Filtering

The wireless signal tracking method measures signal strength transmitted from an AP at different rotation angles (of the directional antenna); when the signal DOA is determined, the robot heads in that direction. This approach is cost-effective, but if the AP is not in line of sight, the radio signal will be reflected or absorbed by obstacles such as walls (shadowing and multi-path fading), as shown in Figure 3.2.

![Figure 3.2. Illustration of multi-path effects](image)
For example, assuming the robot in Figure 3.2 has to head towards the illustrated APs, the robot needs to first track the signal of the closest, $AP_1$. The desired DOA is around 90 degrees; however, its measurements can be affected by signals reflected off the walls, resulting in a calculated DOA that may not be 90 degrees. Because DOA estimation is noisy in nature, it is hard to get an exact DOA from measurements. To alleviate the noise, the system compensates by filtering or smoothing the fluctuating data. One approach for doing so is to use a moving average, determined by Equation 3.2.

$$RSS^f(\theta_i) = \text{AVG}(RSS(\theta_i)) = \sum_{i=i-k+1}^{i} RSS(\theta_i)$$  \hspace{1cm} (3.2)$$

where $RSS(\theta_i)$ is a raw RSS measurement at $\theta_i$, $i$ is the index of a measurement, $k$ is the length of moving window, and $RSS^f$ is the moving averaged RSS value. However, the difference between maximum and minimum values is not significant; using only a moving average cannot allow us to distinguish the maximum RSS point. Thus, this study proposes another approach, a probability-based estimation method that combines the average and variance. Even if an RSS measurement in the wrong direction is higher than that of the direct path, its data will fluctuate and have much higher variance than the data in the direct path.

Accordingly, the following equation can be derived:

$$\hat{\theta}_i = \arg \min_{0 \leq \theta \leq 180} \alpha \text{VAR}(RSS^f(\theta_i)) + \beta RSS^f(\theta_i)$$  \hspace{1cm} (3.3)$$

where $\hat{\theta}_i$ is the estimated DOA at the $i$th time frame, $\text{VAR}(\cdot)$ is the variance of $RSS^f(\theta_i)$, $RSS^f(\theta_i)$ is the moving average from Equation 3.2, and $\alpha$ and $\beta$ are positive gains.

Using Equation 3.3 results in filtered data, as shown in Figure 3.3. Assuming the length of the window is three, this proposed filtering method can consider both average and variance of measurements in noisy environments. The DOA determination should select degree 0, which is the highest RSS signal value. However, if this direction is not accessible because of obstacles, the system should choose the DOA of the second highest
RSS. Taking average values only, the value at 70 degrees is higher than at -30 degrees, but its variance is also significantly higher.

To recap, the overall DOA estimation process is illustrated in Figure 3.4. Filtering provides a consistent DOA over time by reducing fluctuation in measurements.

3.3.2 Sensor Fusion

While DOA provides a general indication of the robot’s direction, an ultrasonic sensor or LIDAR is needed to provide local information for the dynamic avoidance of
obstacles. The proposed method sets a threshold for a certain safety distance and decides DOA according to the highest signal within this safe range. The example shown in Figure 3.5 assumes a safe zone threshold of four meters. Although the highest RSS measurement value is obtained in the vicinity of 30 degrees, the sonar sensor value at this degree indicates an unsafe zone. Thus, this angle is not selected as the DOA. Instead, a DOA of -60 degrees ($P_2$) is selected as it is the closest angle to the highest RSS value located in the safe zone. This method effectively avoids obstacles while maintaining the best DOA.

For implementation of this method, this study utilizes the Pioneer 3AT (P3AT) and its built-in ultrasonic sensors. The robot is equipped with eight ultrasonic sensors on the front side and provides a ROS topic for monitoring their continuously changing values. Since the robot has eight sensors, we can measure distances to objects every 22.5 degrees. Angles between them that cannot be directly measured are interpolated using a linear interpolation. This interpolation estimates the sonar value $\text{sonar}(\theta)$ across the range as follows:

$$\text{sonar}(\theta) = \text{sonar}(i) + frag_{ij} \text{ where } (-90 < \theta \leq 90) \quad (3.4)$$
where \( i \) is the index of the ultrasonic sensor \( (i = 1, 2, \ldots, 8) \), \( \text{sonar}(i) \) is the \( i \)th sonar measurement, and

\[
\text{frag}_{ij} = j \times \frac{\text{sonar}(i+1) - \text{sonar}(i)}{22.5} \quad \text{for each } i = 0, 1, \ldots, 7 \text{ and } j = 0, 1, \ldots, 22.
\] (3.5)

3.4 Implementation

3.4.1 Hardware

Figure 3.6 shows the equipment setup for the P3AT robot. The robot consists of a directional antenna, a wireless adapter, and servo motors to rotate the antenna. The wireless adapter is connected to the on-board laptop via a USB port, and the computer directly manipulates the adapter and monitors the signal strength. A 360-degree camera and a rotating Wi-Fi directional antenna are additionally connected to the onboard computer through ROS-compatible drivers. The directional antenna measures the Wi-Fi RSSI within a specific angular range as rotated by a servo pan system. During the experiment, the directional antenna measures RSS from 0 to 180 degrees based on the direction the robot is oriented. This angular scan of RSSI is then used to calculate the DOA of the Wi-Fi AP. Built-in ultrasonic sensors on the mobile robot are used to determine the DOAs of obstacles, which are then used to avoid any obstacles on the planned trajectory.

3.4.2 Software

Once the user sends a destination to the server, path planning is started to find an appropriate path by which to reach the destination. The robot then starts scanning and navigating by Wi-Fi signals from the waypoint APs. The signal tracking approach effectively finds a reasonable path, tracks the destination, and avoids obstacles while maintaining the best DOA. Table 3.1 indicates the published and subscribed ROS topics.
from the robot. The client side subscribes to the DOA topic to be aware of the DOA value for each RSSI scan. Another published topic, ‘cmd_vel’ broadcasts a linear and angular velocities to the P3AT robot.

Subscribed topics such as ‘RosAria/sonar’, ‘RosAria/motors_state’ and ‘RosAria/pose’ return the current values of sonar sensor measurements, motor status, and odometer values, respectively. ‘Pathwaypoints’ and ‘isMoving’ are custom topics used to synchronize the movement of the robot with the client side. For instance, the Pathwaypoints topic delivers the name of the destination AP.

The overall flow of the program is illustrated in Figure 3.7. The software handles robot control, message exchange, radio signal scanning, and the estimation process. When the program is executed, it starts an initialization operation. First, it creates an instance that contains basic AP information; a ROS node handler, subscriber, and publisher; and
Table 3.1. *Used ROS topics*

<table>
<thead>
<tr>
<th>Publisher</th>
<th>Topic</th>
<th>(Msg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RosAria/cmd_vel</td>
<td>DOA</td>
<td>geometry_msgs/Twist</td>
</tr>
<tr>
<td></td>
<td></td>
<td>std_msgs/Int32</td>
</tr>
<tr>
<td>RosAria/sonar</td>
<td></td>
<td>sensor_msgs/PointCloud</td>
</tr>
<tr>
<td>RosAria/pose</td>
<td></td>
<td>nav_msgs/Odometry</td>
</tr>
<tr>
<td>Subscriber</td>
<td>Pathwaypoints</td>
<td>std_msgs/String</td>
</tr>
<tr>
<td>isMoving</td>
<td>std_msgs/Bool</td>
<td></td>
</tr>
<tr>
<td>RosAria/motors_state</td>
<td>std_msgs/Bool</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.7. Flow chart of the overall program logic.

estimation variables. On the client side, a socket connection is made through the ROS bridge, and the robot waits for a ROS message containing user input that indicates the order of the nodes to be visited. AP information collected in advance by site survey is stored in the form of a directed graph, and the shortest path to the input destination is found using the Floyd-Warshall algorithm. This shortest path is represented as a queue (an order of AP) consisting of the vertices (nodes) of the graph.
When the initial data setup is finished, RSS tracking is repeated until the path queue is empty. The program determines DOA through the estimation process while tracking the RSS. When the estimation value of the DOA becomes smaller than a predetermined threshold value—that is, when the signal strength becomes stronger than the threshold value—the program pops up the queue. Whenever an element is popped from the queue, the network disconnects the existing connection and establishes a new connection with the AP of the element being popped.

For the DOA estimation process, the probability-based approach and sensor fusion techniques were applied in order. The next section details the experimental design and results from actual measurements.

3.5 Experiment and Results

3.5.1 Path Planning

In order to determine if path planning is effective, the shortest path through the obvious example introduced in Figure 3.1 was determined, and then a path planning test was performed with randomly allocated APs. In the case of the random test, graphs were constructed with different numbers of APs (5, 10) and spaces of different sizes ($50^2 m$ and $100^2 m$), and minimum cost paths were created from them. The locations of nodes and the numbers of walls between nodes were all randomly generated, and distances were determined using Equation 3.1. Figure 3.8(a) is the result of simulating the creation of a route that travels through nodes 2 and 3 rather than traveling straight through walls. Other simulations showed reasonable results. However, when a graph was created with many nodes in too small a space, as shown in Figure 3.8(c), it sometimes created inefficient paths. This was because the characteristics of the graphs could result in physically impossible maps.

Table 3.2 shows the paths generated using the graph in Figure 3.8(b). Row and column labels indicate each node in the graph, while cell contents indicate the path from one node to the other. For example, [1,5] means that the shortest path between nodes 1
Table 3.2. Paths generated from the simulation shown in Figure 3.8(b).

<table>
<thead>
<tr>
<th>Node No.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>[1,2]</td>
<td>[1,3]</td>
<td>[1,5,4]</td>
<td>[1,5]</td>
</tr>
<tr>
<td>2</td>
<td>[2,1]</td>
<td>-</td>
<td>[2,3]</td>
<td>[2,4]</td>
<td>[2,4,5]</td>
</tr>
<tr>
<td>3</td>
<td>[3,1]</td>
<td>[3,2]</td>
<td>-</td>
<td>[3,4]</td>
<td>[3,5]</td>
</tr>
<tr>
<td>4</td>
<td>[4,5,1]</td>
<td>[4,2]</td>
<td>[4,3]</td>
<td>-</td>
<td>[4,5]</td>
</tr>
<tr>
<td>5</td>
<td>[5,1]</td>
<td>[5,4,2]</td>
<td>[5,3]</td>
<td>[5,4]</td>
<td>-</td>
</tr>
</tbody>
</table>

(a) Simulation with 4 APs  
(b) Simulation with 5 APs  
(c) Simulation with 10 APs in a small space  
(d) Simulation with 10 APs in a large space

*Figure 3.8. Path planning simulations. The red line depicts a direct path from starting node to destination node, but this route may be blocked by walls. The green line indicates a sub-optimum path generated by the method proposed in Figure 3.1.*

and 5 is a direct route. In the case of [2, 4, 5], it was more appropriate to go through node 4 because there were three walls in the direct line between nodes 2 and 5.
3.5.2 DOA Estimation Test

![Diagram](image)

(a) DOA estimation experiment setting  
(b) Sensor fusion experiment setting

Figure 3.9. Structure of the experimental environment.

DOA estimation is the simplest module of the navigation process. In this test, the DOA was estimated using the moving average and probability filtering method based on RSS measurements in an office environment, at a distance of five meters, and at different angles (60, 90, and 120 degrees). Figure 3.9(a) presents the test environment setup, which excluded external factors such as obstacles and human movement that could affect RSS measurements.

The estimation process was performed using Equation 3.3, where constant $\alpha$ and $\beta$ were both set to 0.5 so as to have equal weights. Figure 3.10, Figure 3.11, and Figure 3.12 plot RSS measurements from the directional antenna at fixed locations. The graphs of mean and deviation show that the maximum signal strength value varies with measurement angle (Figure 3.10(a), Figure 3.11(a), and Figure 3.12(a)). Figure 3.11(b) an ideal case wherein the proposed method finds the peak RSS values. As shown in Figure 3.10(b), even if measured values near the DOA point are similar to or the same as one another, the desired DOA can be found using the moving average and variance. In the case of Figure 3.12(b), the direction of the control could be determined by reflecting the values observed in the previous step even if the highest signal measured by the robot in real time did not correspond to the desired DOA.
3.5.3 Sensor Fusion Test

In the sensor fusion test, one AP and the robot were placed in fixed positions and the change in DOA value measured before and after placing an obstacle in the line of sight between AP and robot. The robot and the AP were placed at a distance of 5 meters, and the obstacle was located at a distance of 1 meter from the robot (Figure 3.9(b)). Assuming no obstacles, the original DOA should be calculated as close to 90 degrees. In the presence of an obstacle, the DOA should be determined as the closest safe zone of 75 degrees since
Figure 3.12. DOA estimation results at 120°.

the obstacle lies on the 90 degree line. A plot of the modified DOA value determined from the ultrasonic sensor is shown in Figure 3.13(c), and illustrates the result of determining whether the direction of movement should be to the left or right of the object. The average DOA was 84.6875 when there were no obstacles; in the presence of obstacles, the closer value of 75 degrees seemed to be selected more frequently than 110 degrees.
Figure 3.13. Sensor fusion results.

(a) RSS measurements with sonar sensors

(b) Marked safe and unsafe zones

(c) DOA frequency
CHAPTER 4. VR SYSTEM DESIGN AND IMPLEMENTATION

This chapter covers the implementation and design of the VR system and introduces the overall system structure into which its components are merged. Also included is the navigation test conducted to show the feasibility of that integrated structure.

4.1 VR Interface Design and Implementation

To fully support the envisioned virtual tour concept, the system architecture includes the following essential components: streaming and rendering of VR video, signal tracking for indoor autonomous navigation, and integration with the VR interface. The system is categorized into three entities: the VR user interface at the client (User), a mobile robot which is the moving agent (Robot), and the path planning and tracking functions to generate the robot path (Server), as shown in Figure 4.1. The system subcomponents must also exchange necessary information with each other. The following sections introduce the detailed VR interface design and implementation.

4.1.1 Virtual Reality (VR) Interface Design

The most common form of virtual tour is web-based. Typically, a 360-degree pre-recorded video is played on a webpage, and the user switches the view between locations using a mouse, keyboard, and buttons. Figure 4.2(a) shows screen captures from a current web-based virtual tour application. Existing telepresence systems have most commonly implemented their virtual tour systems in the same way as web based methods (Jaselskis, Sankar, Yousif, Clark, & Chinta, 2014; Katz & Halpern, 2015). In order to operate these systems, it is necessary to inform the user of destinations that can be reached, as shown in Figure 4.2, and for the user to manually select the next destination using the arrow marker. For the VR interface in this study, rather than manually adjusting the field of view via a mouse, the 360-degree video rendered to the HMD is designed to be
freely adjustable based on the users head motions. The interface includes a list of destinations that are commonly viewed, shown in the example interface of Figure 4.2, and a media element such as a video/picture as is necessary for a virtual tour.

4.1.2 Implementations and Results

The primary roles of the VR interface are to render the live video stream transmitted from the robot’s HTTP server and to exchange ROS messages. The user interface was constructed using the Oculus Rift (2015) and OpenVR SDK. The system is shown in Figure 4.3.

The 360-degree camera shoots a wide angle of view with distorted images through two fisheye lenses, each of which captures a viewing angle of greater than 180 degrees. The interface software then maps the video image to the image shader, which is a function that calculates the position and color of pixels to be displayed on-screen. The shader in Figure 4.5(b) is set to show the original video image, Figure 4.6(a). By stitching together the image frames as shown in Figure 4.6(a) by the orange guidelines in Figure 4.6(a), a
Figure 4.2. Examples of existing virtual tours.

A conventional UI, such as used in a game application, is mostly overlaid on the screen to display information needed for the system. However, in a VR interface, a stereoscopic interface must be displayed differently at two different points of view. First, a control panel is created with a dropdown bar and buttons and fixed at a specific location in the world space. This panel allows the user to control the connection with the robot at a glance or to see selectable destinations. If the general intention is to overlay the UI on the
Figure 4.3. A user with the VR interface (left), the mobile robot with a 360 video camera (right).

(a) Head orientation: Left  (b) Head orientation: Right

Figure 4.4. Rendering objects in a fixed location while the head is moving (*Unity - Interaction in VR*, n.d.). The pop-up window shown at the lower right is the user’s view through the HMD.

... screen, this control panel canvas is overlaid on the camera, giving the user the feeling that the UI elements are settled at a fixed point (Figure 4.4). In fact, these elements rotate according to the user’s head rotation.
Figure 4.5. Image shader for spherical object.

The 3D display the user can see is shown in Figure 4.6(c) (but stretched to 2D in the picture). As the camera uses dual fisheye lenses, it needs to send two photos simultaneously to the user; there is a trade-off every time the program does post processing and sends the images. Streaming latency can be improved by changing settings such as video frame bitrate or streaming buffer size. In this implementation, streaming is performed using the free software ffmpeg, which supports encoding / decoding of various formats of digital voice and video streams. The streams generated by ffmpeg are broadcasted through ffserver. The user can view the broadcasted video on the VR interface by connecting to this server as a client.

At the same time, the interface also sets up wireless communication with the mobile robot. Using ROS Bridge, the interface and the robot exchange ROS messages, which consist of data values. ROS Bridge provides an API compatible with non-ROS platforms on the basis of a Web socket.

The RICOH Theta S is a 360-degree camera that shoots video with a resolution of 1280x720 px in MJPEG format. A video streaming server then converts the MJPEG stream to a format playable by the media player in the VR interface. In essence, the video is encoded and sent to the server, and the server goes through the process of distributing it to the viewer; this usually results in a delay of three to four seconds. As shown in Figure 4.8(c), the user can select and change the destination through the VR controllers.

ffmpeg: https://www.ffmpeg.org/
that interact with UI components, and can do so while watching the 360-degree view displayed on the HMD (Figure 4.8(a)). Additionally, the UI has a fixed canvas showing a list of destinations (Figure 4.8(b)) accessible from the current position. All destinations are mapped to nearby APs so that when the user confirms the destination, a route can be automatically created. The 360-degree video is transmitted with the original distortion from the fish eye lenses, and is converted to equirectangular by the VR interface. As shown in Figure 4.6, the distorted image is projected on a sphere object, of which each hemisphere displays the image for each fisheye feed (Figure 4.6(b)), and the user can watch the projected image in the HMD. Figure 4.7 shows a rendered image in Unity and in the Oculus HMD. The image averaged 13 fps.

Figure 4.6. Rendering process for the 360-degree video stream.
Figure 4.7. Screen capture in HMD; left = left eye view, right = right eye view

(a) UI control panel
(b) List of selectable destinations
(c) UI interaction with VR controller

Figure 4.8. User interface elements.
4.2 Experiment and Results

Experiments were conducted to determine the effectiveness of each module and component. First, simulations were used to validate the path planning method in spaces of different sizes and with different numbers of AP nodes. The signal tracking approach was validated through three tests: DOA estimation, sensor fusion to detect obstacles, and running the entire navigation process in a real environment.

![Figure 4.9. Comparison of the angular velocities of a human head and a servo motor](image)

4.2.1 Effectiveness of VR Interface with 360-Degree Camera

One of the biggest advantages of using a 360-degree camera is that when viewing the video through a VR display, the user can freely move his head without any interruptions or delays. In contrast, when a VR display is used with a normal camera (ELP USB camera module; [link](http://a.co/3YJoHn)), the camera must be rotated following the movements of the user's head. Therefore, this section compares the ordinary head rotation speed of a human and the speed of a servo motor (ROBOTIS Dynamixel AX-12W; [link](http://www.robotis.us/dynamixel-ax-12w/)). The angular velocity $\omega_z$ when a person turns his or her head from left to right is 5.85 rad/s on average (Bussone, 2005)
and the value of servo motor is 4.64 rad/s. It takes less than 1 second for the human head to rotate from 0 to 180 degrees, while the servo motor takes less than 1.5 seconds (Figure 4.9). This delay of about 0.5 seconds leads to differences between the user’s field of view and the camera view, which can cause virtual sickness.

4.2.2 Navigation Test

A navigation test was conducted to demonstrate the feasibility of the entire system that integrates the VR interface, streaming, and robot navigation. The experiment was conducted in an indoor environment with two APs separated by a straight line of about 10 meters and with a corner (Figure 4.10). The robot was assumed to have arrived at its destination when the distance between the robot and the AP was less than one meter or when the signal strength is above a given threshold. In conducting the test, the user first designated the destination using the controller (Figure 4.8(c)), then the robot began navigation while the user observed the robot’s progress using the HMD and checked whether the video stream played without stuttering.

Since there is a wall between the destination and the starting position of the robot, a straight line to AP2 (the dashed line in Figure 4.10) is not feasible, and the path generated goes through AP1 instead. In Figure 4.11(a), it can be seen that there was an error in the movement of the actual robot, but overall it moved well according to the designated path. The reasons for the error include interference from other APs and multi-path effects from walls and objects, but the main cause was the moving average filter used in this test. The fixed-size window used in the moving average calculation may influence the robot’s rotation when addressing significant changes in RSS measurements.

Figure 4.11(b) shows changes of DOA values from the proposed methods for the navigation. The DOA calculated based on the Wi-Fi signal value alone is relatively small; however, the ultrasonic sensor had a higher value between 90 and 180 degrees than in the 0 to 90 degrees interval, and the sensor fusion DOA value seemed to have been calculated to rotate the robot further to the right. Separately, because it renders the distorted image, the boundaries of fisheye lenses seemed particularly curved or looked different from the
actual image. This distorted image does not affect navigation, but the user may feel uncomfortable. Also, if the threshold value used to determine arrival is changed to focus on signal tracking quality, the path to the next AP became ineffective, or the navigation was terminated two or three meters from the AP instead of at the one meter target distance.

Figure 4.13 shows pictures of the robot actively navigating alongside screen captures of the user’s view through the HMD. These were taken at designated places, shown in Figure 4.12. Although the user could freely view the streaming data at 360 degrees, the center of the video remained unchanged even if the robot rotated, so that the user had to adjust the field of view accordingly. This experiment showed the robot could effectively plan a route to avoid passing through the wall and successfully reach an endpoint one meter from the designated AP. In other words, the path planning and navigation components work well together and with the users input. Furthermore, the streaming server was able to transmit video at over 13 fps through the end of navigation. Consequently, the experiment the proposed system is verified as successful.
(a) Robot’s trajectory. The shaded area is a wall or a door.

(b) Change of DOA values

Figure 4.11. Navigation test results.
Figure 4.12. Screen capture locations
Figure 4.13. HMD captures at each designated location
CHAPTER 5. CONCLUSION

This study proposes an intuitive robot telepresence system with VR-based interface and Wi-Fi signal tracking method. Specifically, this thesis focuses on the development of a virtual tour system which integrates the following essential components: streaming of 360-degree video, a VR-based user interface, and a mobile robot for navigating a remote environment. The presented architecture can be used to enhance and realize an immersive telepresence system with more relaxed and intuitive control experience compared to existing methods that require manual control.

The 360-degree video rendered by the VR interface allows the user to easily monitor the environment surrounding the robot during navigation. Due to characteristics of the video, it is necessary to transmit frames of large size; in a normal network environment, noticeable latency occurs. While testing, several frame drops were observed during playback. This issue is affected by the performance and the state of the streaming server, and it was found to disappear when run on a local network with a wired connection.

The mobile robot platform used for autonomous navigation was intended to give the impression of a person walking. Thus, the robot plans a path from the starting point to the destination based on the users input. The path planning method was validated through simulations using both a simple example and complex random maps. The results showed that path planning was successful in general, although inefficient paths were sometimes created when too many APs were distributed in a small space. This indicates that selection of appropriate numbers and locations of APs is important for the proposed system.

To enable independent travel in an indoor space, we adopted a Wi-Fi signal tracking method that is cost-effective and enables video streaming to continue regardless of the position and motion of the robot. A filtering method was used to reduce errors that may occur during signal tracking, and we demonstrated it works well in real driving scenarios. However, in the navigation test, the robot’s direction of movement occasionally and unnecessarily changed to the left or right due to the influence of the RSS and the readings of ultrasonic sensors.
In conclusion, we were able to validate the feasibility of the proposed system by showing that the mobile robot successfully reached the destination through the navigation experiment including a live video streaming, DOA estimation, and sensor fusion.

We observed several technical issues addressed above, but we leave them for future research. Therefore, future research will work to improve the wireless network experience in terms of achieving reasonable latency while maintaining a high-resolution 360-degree video stream for VR. This issue can be resolved by cropping and rendering only the part of the video corresponding to the direction the user is looking. Additionally, a user study is planned to qualitatively compare the differences between the VR interface using a 360-degree camera and a control environment with conventional teleoperation interfaces. We will incorporate feedback from the user study and select other essential functionalities to be added.
REFERENCES


