SUMMER INTERNSHIP REPORT

Development of TeleOperation and TeleObservance Robot

Sumukh Porwal
Roll No.: **ME20B041**
B. Tech. Mechanical Engineering
Indian Institute of Technology, Tirupati

Mentor:
Srikrishna S
Director
SeiAnmai Tech.

Supervisor:
Prof. Dr. S K Saha
Mechanical Engineering Department
Indian Institute of Technology, Delhi.

Abstract

The ”Development of TeleOperation and TeleObservance Robot” project marks a remarkable journey into the realm of robotics and autonomous systems. Our team at SeiAnmai Tech. collaborated to create an innovative mobile platform capable of autonomous navigation and seamless teleoperation, tailored to address the growing demands of various real-world applications.

Guided by our vision for a versatile robotic system, we set out to design a robot that could operate efficiently without human intervention, while also incorporating features that enable intuitive remote control and observation. The foundation of our robot was built upon the powerful Robot Operating System (ROS2) Humble, providing a robust framework for the integration of diverse functionalities.

Navigating through dynamic and static obstacles with confidence, the robot demonstrated its autonomy through Simultaneous Localization and Mapping (SLAM) techniques. The integration of micro-ROS with Raspberry Pi Pico allowed for seamless communication between the high-level ROS2 environment and the low-level hardware, enhancing the robot's adaptability and responsiveness.

Autonomous docking, driven by Aruco marker detection, showcased the robot's intelligence as it effortlessly navigated and docked with precision. This cutting-edge application exemplified the potential of robotics in practical scenarios.

Throughout the development process, meticulous calibration of motors and fine-tuning of PID control algorithms laid the foundation for the robot's stable and accurate movements. Extensive testing and debugging ensured the system's robustness, making it adaptable to diverse environments and scenarios.

The complete setup was implemented using the Robot Operating System (ROS2 Humble) framework. The primary rationale behind this choice was the ability to reuse code, achieve modularity, and enable seamless communication among different modules.

Acknowledgement

First and foremost, I extend my deepest appreciation to my mentor, Srikrishna S, whose unwavering support and guidance played a crucial role in shaping the course of this project. Their profound knowledge, invaluable insights, and willingness to assist with every challenge I encountered were instrumental in my growth as a roboticist.

I am also immensely thankful to my supervisor, Dr. S K Saha, for providing me with the opportunity to work on this fascinating project. Furthermore, I extend my gratitude to the entire team at SeiAnmai Tech., whose collaborative efforts and constructive feedback enriched the project's development process. Working alongside such talented and supportive individuals has been an inspiring and rewarding experience.

I am grateful to all the experts and researchers whose published work and contributions in the field of robotics served as a valuable reference during my research and development phases.

Last but not least, I am thankful to Indian Institute of Technology, Delhi for providing the necessary resources, infrastructure, and opportunities that made this internship possible.

About SeiAnmai Tech.

SeiAnmai Tech. is a technology company with a clear vision and a promising story behind its inception. The company focuses on providing Internet of Robotics solutions for various sectors, including Home, Office, Industry, and Education.

Vision:
Initially, the company aims to tap into the educational robot market, catering to the needs of schools, colleges, and other educational institutions. This segment provides an excellent opportunity to introduce robotics and automation concepts to students and educators, fostering interest and innovation in the field.

After establishing a foothold in the educational robot market, SeiAnmai Tech. aspires to venture into the realm of telepresence robots. Telepresence robots are devices that allow people to remotely interact and navigate through a physical space. These robots find applications in various industries, enabling telecommuting, virtual meetings, and remote inspections, among others.

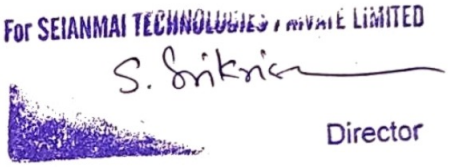
Story:
SeiAnmai Tech.'s journey began as part of the READY (Research Entrepreneurship And Development for You) program initiated by IHFC (Innovation Hub for Cobotics) at IIT Delhi. This program offers a unique platform for young talent in India to collaborate with top academic and research experts in the field of robotics and cobotics (collaborative robotics).

As part of the READY program, SeiAnmai Tech. received crucial support from IHFC. This support included financial assistance, mentorship, and guidance, which played a pivotal role in facilitating the establishment of the company.

Internship Completion Certificate

Srikrishna S
Director
SeiAnmai Tech.

This is to certify that **Sumukh Porwal**, an undergraduate student of ihe Indian Institute Of Technology, Tirupati, Mechanical Engineering Department, (Roll Number: **ME20B041**), had done a great performance during the internship program from **May 14, 2023, to July 14, 2023**, while working on the project '**Development of TeleOperation and TeleObservance Robot**' under my guidance.



(Srikrishna S)

Contents

List of Figures

1 Problem Statement

The aim of this project is to design and construct an advanced autonomous robot capable of teleoperation and teleobservation. The robot should possess the ability to navigate autonomously, demonstrating efficient path planning and obstacle avoidance in real-world environments. Furthermore, the robot should be equipped with a user-friendly teleoperation interface, allowing remote control of its movements and camera orientation. The objective is to create a versatile robotic system that can seamlessly switch between autonomous operation and human interaction, opening possibilities for various applications in fields like surveillance, exploration, and remote monitoring.

2 Introduction

2.1 Mobile robotics

Mobile robotics, involving locomotion via wheels or legs, is a swiftly advancing field with extensive research potential for applications like robot servants, personal assistants, and delivery robots. Key areas of focus include motion planning, exploration, and Simultaneous Localization and Mapping (SLAM). These robots combine artificial intelligence with physical capabilities to navigate surroundings. Ensuring robustness is crucial for operating effectively in complex and uncertain environments.

2.2 ROS2 Humble

ROS2 is an advanced software framework for robot development, offering streamlined processes and eliminating redundant efforts. It enables communication between different programs (nodes) more efficiently, supports multiple programming languages (Python, C++, Matlab), and fosters collaboration with independent packages. ROS2 is compatible with various hardware platforms, including amd64 and arm-based processors, and functions on both Windows and Linux systems. As an open-source platform with an active community, ROS2 continues to improve with new efficient packages, making it an essential tool for roboticists.

2.3 ROS2 Topics

ROS2 topics are communication channels that facilitate data exchange between different parts of a robot system. Publishers publish messages to specific topics, and subscribers receive these messages, allowing efficient decoupled communication within the ROS2 architecture. Topics enable the dissemination of sensor data, status updates, and commands, fostering modular and scalable robot application development.

2.4 Micro-ROS

Micro-ROS is a lightweight and real-time communication framework that extends the Robot Operating System (ROS) to resource-constrained embedded systems. It allows ROS applications to run on microcontrollers, enabling seamless integration of small-scale robots

and IoT devices into the ROS ecosystem. Micro-ROS optimizes memory usage and supports real-time capabilities, making it ideal for low-power and time-critical applications in robotics and automation.

2.5 Hardware

The robot is a differential drive robot with 4 caster wheels. It houses a RPi camera for assimilating RGB images. It has an Raspberry Pi Pico as Microcontroller and a Raspberry Pi 4 for Microprocessor. It is powered by LiPo battery and a custom-build power supply circuit. It also features a charging circuit which allows for autonomous docking and charging.

2.6 Odometry

Odometry is the process of estimating the distance travelled and path travelled by the robot by mounting sensors on the shaft of the actuator. This can be used in the case of differential drive robots such as robomuse to estimate the x,y, yaw of the robot when it moves on a flat horizontal plane. However, this requires the assumption of pure rolling and no noise. So we adopt a probabilistic version of this model.

2.7 Simultaneous Localization and Mapping (SLAM)

SLAM involves mapping the environment and determining the robot's pose using features from the surroundings. This process uses Extended Kalman filter and Particle filter with input data from wheel odometry and the Lidar sensor. By employing a probabilistic model, the robot robustly estimates its pose. The SLAM Toolbox package enables the robot to predict its pose while mapping.

2.8 Navigation

The process of moving the robot autonomously from place to place in a known map is called navigation. Utilizing the map and wheel odometry data, the robot plans both global and local paths. The global path planning relies on the A* search algorithm, while the local motion planning involves the data from the Lidar sensor to perform obstacle avoidance dynamically. By interfacing with the navigation stack, the development of mobile robots becomes more efficient, requiring parameter tuning specific to each robot's attributes for seamless navigation within the map.

2.9 Costmap

A costmap is a key component in robot navigation that represents the environment's occupancy status by assigning costs to different regions. It provides essential information to the robot for path planning and obstacle avoidance during autonomous navigation.
There are two types of costmaps: local and global. The *local costmap* focuses on the immediate surroundings of the robot and is updated in real-time, aiding in obstacle avoidance during close-range movement. In contrast, the *global costmap* represents the broader environment and is used for long-term path planning and navigation decisions.

2.10 TF tree

The TF tree (transform tree) is a fundamental concept in ROS (Robot Operating System) that manages coordinate transformations between various frames of reference. It helps maintain spatial relationships between different sensors, robot components, and the environment, enabling accurate data fusion and navigation planning.

3 Approach

3.1 Simulation

Why were simulations made?

Simulations were used to validate and optimize the performance of an autonomous robot in a virtual environment. They ensured effective web-based remote control and allowed algorithm fine-tuning, navigation strategy evaluation, and hardware integration. Addressing potential issues improved the robot's efficiency and capabilities for real-world implementation.

Procedure

The project phase involved simulating the entire robot with SLAM and Autonomous Navigation using ROS Noetic. Gazebo provided a realistic testing environment without physical hardware. Configuration, ROS navigation stack implementation, and parameter tuning were performed. Transitioning to ROS2 Humble improved real-time performance and communication protocols, enhancing the robot's abilities for real-world implementation.

3.2 Hardware Setup

The harware setup consisted of 3 layers:

Bottom Plate - Electronics Layer

All the electronic components and circuits including battery were on the bottom plate. The bottom plate had the most weight to lower the Centre of Gravity of the robot.

Middle Plate - Lidar Layer

Lidar sensor and cooling system for the micro-processor and battery management system were installed on the middle plate.

Top Plate - Camera and its Pan-Tilt system

Camera and its Pan-Tilt system including indicator LED and Emergency Stop button are placed on the top plate.

3.3 Controller and Convertor programs

Teleop Control

A python program was made to publish velocity for the robot in the form of linear and angular velocity on a ROS2 topic named '*/cmd_vel*'.

Data Conversion Programs

Python programs were written to convert linear and angular velocity to required revolutions per second (rps) for each motor and another program to convert the encoder ticks data of the motor to odometry data for pose estimation of the robot.

PID controller

A closed loop PID controller program was written in C language for microcontrtoller to help the motors reach the required revolutions per second.

Publishing data from Raspberry Pi Pico

Raspberry Pi Pico published Battery Percentage, Hall sensor data and Encoder Tick of each motor which are used to check battery level, docking status and pose estimation of the robot respectively.

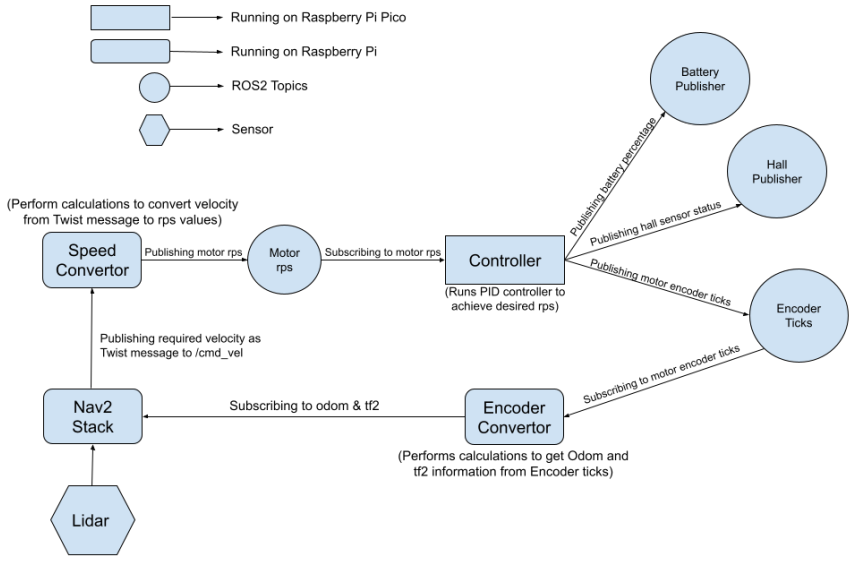


Figure 1: Flowchart of controller loop

3.4 SLAM & Navigation

Integrating Lidar Sensor Data with ROS2 environment

For seamlessly integratation of the Lidar sensor data with robot's ROS2 environment, I modified the standard rplidar_ros package to ignore the robot's hardware setup as a obstacle while mapping and navigating . This adaptation was essential to meet the specific requirements of our robot's SLAM and Navigation functionalities. By efficiently incorporating the Lidar data, our robot was able to perform accurate Simultaneous Localization and Mapping (SLAM) to navigate its surroundings effectively.

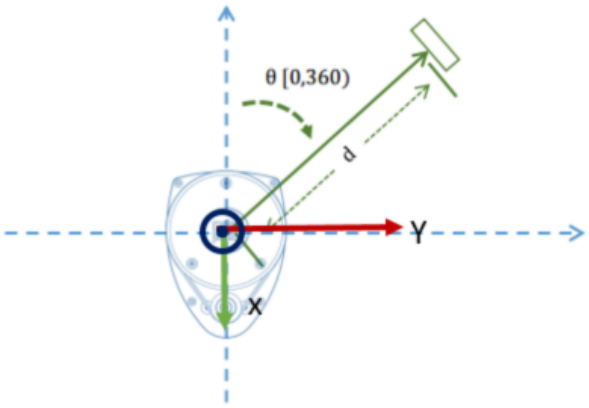


Figure 2: Lidar Schematics

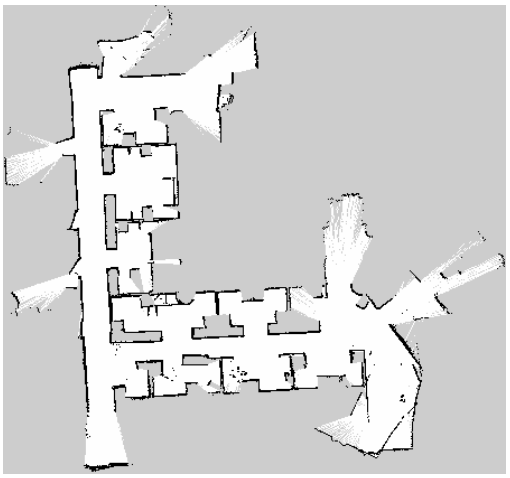


Figure 3: Map generated by robot using SLAM

Generating a map of environment

Map generation for an environment is achieved using the Slam Toolbox. The toolbox employs Simultaneous Localization and Mapping (SLAM) algorithms, which utilize sensor data from LIDAR and other sensors to simultaneously localize the robot's position and map the surroundings. As the robot moves through the environment, SLAM algorithms construct a map by fusing sensor data and estimating the robot's pose. The generated map can then be used for autonomous navigation. Refer Figure 2 for viewing the map.

Configuration of Paramters

Configuration of parameters significantly influences the robot's performance. By fine-tuning parameters such as planner settings, obstacle avoidance behavior, recovory behaviours, and trajectory planning preferences, I optimized the robot's navigation. Adjusting parameters like the inflation radius and cost scaling factors allowed the robot to avoid obstacles more effectively and follow smoother paths. The iterative tuning process involved comprehensive testing and analysis, ultimately enhancing the robot's navigation capabilities and ensuring safer and more efficient movement in various environments.

Integration sharp sensor to Local Costmap

To incorporate the Sharp sensor into the local costmap and enhance obstacle detection, a dedicated program was developed to convert Sharp sensor data into the Range Sensor data type compatible with the navigation stack. Subsequently, the parameters file was carefully modified to facilitate the integration process. By doing so, the robot was able to effectively avoid certain holes and low-level obstacles, optimizing its navigation capabilities in complex environments.

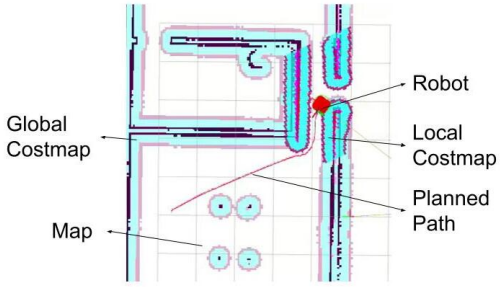


Figure 4: Visualizing Autonomous Navigation Using RViz

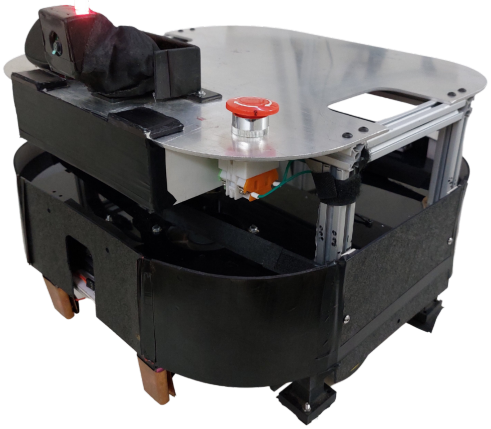


Figure 5: Orthographic Image of the Robot

3.5 Camera and its Pan-Tilt system

Integration of Camera feed to ROS2 Environment

The successful integration of the camera with the ROS2 environment allowed the robot to leverage visual data for object detection and autonomous navigation. I utilized ROS2's image transport system to publish the camera's feed as a ROS2 topic, making it accessible to other nodes. The camera feed was processed and used by the Computer Vision algorithms to help the robot in the docking process.

Joint State Publisher for Camera's Pan and Tilt mechanism

To accurately represent the camera's orientation in the ROS2 environment, I developed a joint state publisher for the pan-tilt system servos. This program published the servo's state to the ROS2 TF tree, which enabled other ROS nodes to access and utilize the camera's orientation information accurately. The joint state publisher is crucial for debugging and verifying the camera's correct alignment during operation.

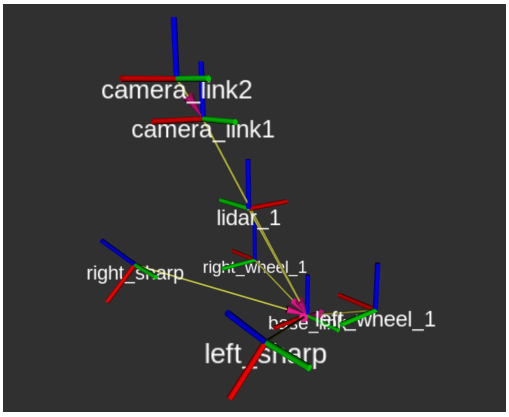


Figure 6: Transform for different robot frames

Figure 7: An example of ArUco marker

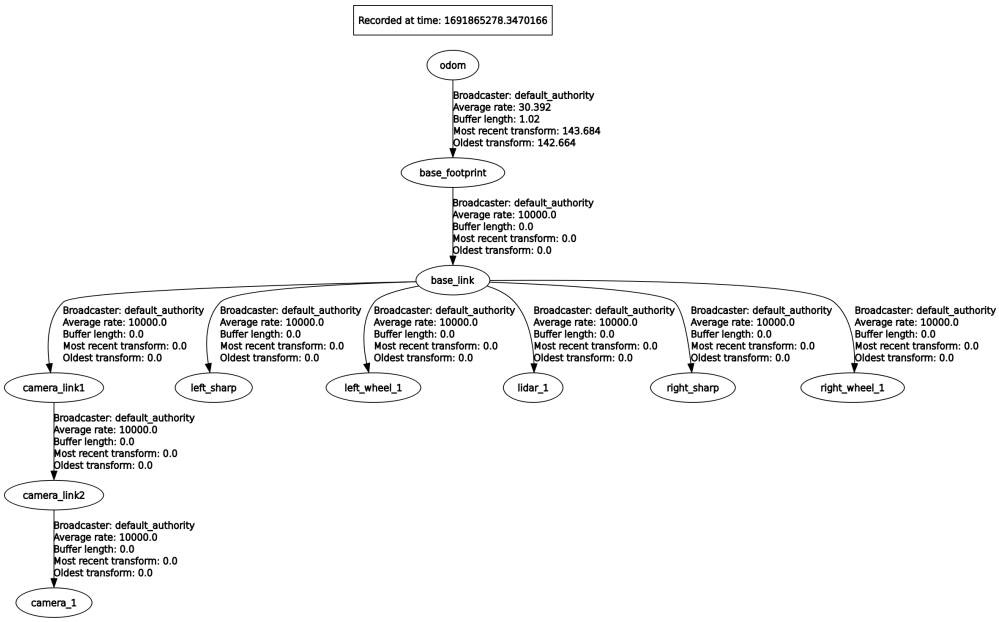


Figure 8: Transformation tree for the robot

3.6 Autonomous Docking

ArUco Markers Detection

Aruco marker detection is a crucial step in the autonomous docking process. Aruco markers are fiducial markers with unique patterns that serve as reference points for localization. To detect these markers, we utilized computer vision techniques in conjunction with the OpenCV library. The robot's camera captures images, and the detection algorithm analyzes the frames to identify the Aruco markers' positions and orientations accurately. This information is then used to determine the robot's relative pose with respect to the docking station, facilitating precise alignment for docking.

Dock process using detected markers

Once the Aruco markers are detected, we compute the desired linear and angular velocities that guide the robot towards the docking station while maintaining proper alignment. By calculating the required velocity commands based on the relative pose obtained from Aruco marker detection, the robot can autonomously navigate towards the docking point. Implementing this step ensures a seamless and accurate docking process, making the autonomous robot capable of independently approaching and aligning with the docking station.

4 Results and Discussion

After extensively testing and evaluating the robot's functionalities in various scenarios to assess its performance . The results and findings of the development process for the autonomous robot are as follows:

Autonomous Navigation Performance:
The robot exhibited exceptional performance in autonomous navigation and SLAM-based environment mapping. With precise localization, it seamlessly navigated through static and dynamic obstacles, ensuring safe and efficient movement. The robot created a map of size 20m x 20m and was able to run navigation effectively for 4 hours with accuracy of 10cm from goal positions. Additionally, effective obstacle detection and avoidance mechanisms, utilizing Sharp sensor to enhance the robot's adaptive navigation capabilities.

Autonomous Docking Performance:
The autonomous docking mechanism using Aruco marker detection exhibited remarkable success, with the robot accurately locating and docking with the charging station or designated points.

Hardware Integration and Communication:
The integration of micro-ROS with Raspberry Pi Pico facilitated smooth data exchange between the ROS2 environment and the low-level hardware, contributing to the robot's adaptability and responsiveness.

Overall Performance and Limitations:
Overall, the robot demonstrated impressive performance in autonomous navigation, teleoperation, obstacle detection, and docking. While the robot showcased strong adaptability, we identified some limitations, particularly in cluttered environments during obstacle avoidance. Further fine-tuning of obstacle detection algorithms could mitigate these challenges.

Discussion of Future Directions:
The successful development of the autonomous robot opens doors to a range of potential future directions. Future enhancements may include advanced machine learning algorithms for object recognition, integration of additional sensors for robust obstacle detection, and further optimization of the trajectory planner for improved navigation efficiency. Additionally, exploring applications in surveillance, exploration, and remote monitoring holds promising avenues for extending the robot's functionalities.

5 Learning and Outcome

Through the project, I have attained substantial learning and advancement across diverse facets of robotics and automation, yielding key outcomes. These comprise the honing of a robust robotics skillset through crafting an autonomous robot imbued with SLAM, obstacle detection, and teleoperation capabilities, augmenting competence in hardware integration and software coding. My mastery of ROS2 Humble's navigation stack has endowed me with a profound comprehension of its sophisticated attributes, enhancing code efficiency, real-time performance, and communication protocols.

Furthermore, my proficiency has expanded into autonomous navigation, having successfully implemented Aruco marker detection for precise docking, amplifying my prowess in computer vision and localization techniques for accurate positioning. This project has also illuminated the practical applications of robotics, unearthing insights into surveillance, exploration, and automation domains. My adeptness in adapting to emerging technologies, exemplified by seamless integration of cutting-edge tools like micro-ROS and Aruco markers, highlights my adaptability and forward-looking approach. These project have not only enriched my understanding of robotics but have also fueled my commitment to continuous growth within this evolving and transformative field of autonomous systems.

6 Summary

As a 3rd-year mechanical engineering undergraduate, the curriculum courses in robotics, control systems, programming, and mechanical design have been instrumental in shaping the success of my internship. The knowledge acquired in robotics enabled me to implement advanced features, such as SLAM and autonomous navigation, in the autonomous robot. Courses in control systems enhanced my understanding of PID tuning for motor controllers, optimizing the robot's behavior. Proficiency in programming allowed me to develop the intuitive Graphical User Interface for teleoperation. Furthermore, the mechanical design courses laid the foundation for efficiently integrating hardware components. The comprehensive education from these courses significantly contributed to the seamless development of the autonomous robot for teleoperation and observation.