RoboCup 2024 - RoboCup Industrial @Work Robo-Erectus@Work Technical Description Paper

Yusuf Pranggonoh (Team Leader), Carlos Acosta (Faculty Member), Emanuel Joseph Rodrigues, Ng Yan En, Chia Zhi Feng, Mahesh S Kunalan, Tan Tian Kovan, and Chang Yong Quan (Student Members)

Singapore Polytechnic,
School of Electrical and Electronic Engineering (SEEE),
500 Dover Road, Singapore, 139651
yusuf_pranggonoh@sp.edu.sg
http://www.roboerectus-atwork.com/

Abstract. This paper provides a brief description of Robo-Erectus@Work that is set to participate in the RoboCup Industrial @Work category in RoboCup 2024. The robot mechanical base is based on the four mecanum wheels (omni-directional) with GLUON-2L6-4L3 arm manipulator and customised end-effector. The software framework is developed under Robotics Operating System (ROS) environment. Some features that are required for operating in industrial environments includes optimal object manipulation and recognition.

1 Introduction

The Robo-Erectus project was started as early as 2002 in Advanced Robotics and Intelligent Control Centre (ARICC) of Singapore Polytechnic. It was one of the pioneer soccer playing humanoid teams in the RoboCup Humanoid League. The team Robo-Erectus@Work was established in 2013 and has achieved the following results:

Year	**Venue**	**Result**
RoboCup 2023	Bordeaux, France	**5th Place**
RoboCup 2022	Bangkok, Thailand	**3rd Place**
RoboCup 2021	Virtual (Worldwide)	**8th Place**
RoboCup 2019	Sydney, Australia	**3rd Place**
RoboCup 2018	Montreal, Canada	**3rd Place**
RoboCup 2016	Leipzig, Germany	**3rd Place**
RoboCup 2015	Hefei, China	**2nd Place**
RoboCup 2013	Eindhoven, Netherlands	**4th Place**

Table 1. Robo-Erectus@Work Result in RoboCup Competition

Our main research includes mobile manipulation in controlled environment and semi outdoor environment such as grasping objects on floor and workstation, omni-directional localization and navigation in unconstrained environment, object recognition and handling in logistics environment.

The team comprises of students from the respective four disciplines, Diploma in Electrical and Electronic Engineering (DEEE), Diploma in Computer Engineering (DCPE), Diploma in Aerospace Electronics (DASE) and Diploma in Engineering with Business (DEB). With the learning experience in SEEE, it gave the team the necessary level of experience and skills to cope with the workload of the project. Some of the lessons the team learnt were about ROS [1], python and C++ languages, electronic circuit design, design and prototyping, and artificial intelligent required to make the project a success.

2 Robot Platform

Robo-Erectus@Work robot platform is equipped with four omni-directional wheels and a six-degree-of-freedom GLUON-2L6-4L3 [2] arm with two-parallel ROBOTIS Dynamixel XM430 as an end-effector. Two outdoor laser scanners (SICK TiM561) are mounted at the front and rear of the robot respectively to localize and navigate robot in the competition arena. A RGB-D camera (Intel Realsense D435) is mounted on the end-effector to see object for detection and recognition. The processor (Intel NUC11PHKi7) is used for computer vision processing, conduct artificial intelligent strategy in the competition and motion control of the robot. An additional processor (NVIDIA Jetson AGX Xavier Developer Kit) for deep learning purposes. Figure 1 below shows the fully assembled Robo-Erectus@Work robot, SPlackie.

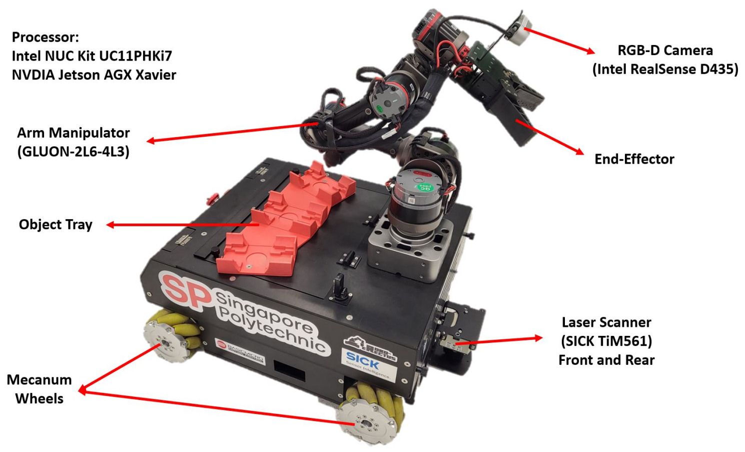


Fig. 1. Robot of Team Robo-Erectus@Work, SPlackie

3 Hardware

SPlackie hardware consists of electrical architecture and mechanical design. The electrical architecture showed the overall layout on the electrical distribution for the robot from the main supply (Batteries or AC Power) to different parts of the robots such as motor drivers, processors, sensors, arm manipulator, end-effector and LED lighting. The mechanical design showed the overall design of the robot from the base layer to top layer.

The motors used to move the omni-directional wheels are DC Brushed Motors (FAULHABER Series 3863), which drive by Motor Controller (Basicmicro RoboClaw). An Arduino Mega board is used to perform inverse kinematics calculation for the robot's position and velocity. The robot information will then sent to processor to perform necessary localization.

3.1 Electrical Architecture

The electrical architecture consists of two layers mainly as Power Layer and Distribution Layer. The Power Layer has two input supplies, one to DCDC Converter (19V/15A) to power the main processor and the other one to main components for the robot. Refer to Figure 2 below for the Power Layer architecture.

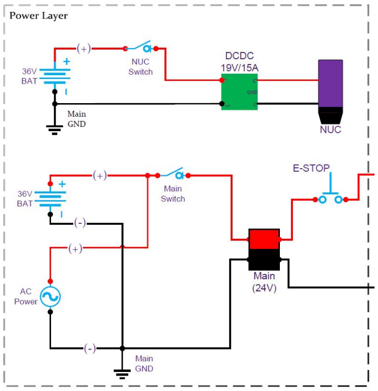


Fig. 2. Power Layer

The Distribution Layer consists three DCDC Converters (24V/15A, 19V/15A, and 12V/15A) to supply different parts of the robots. Refer to Figure 3 below. The following are the lists of power distribution for the robot:

- **24V/15A - Distribution Layer**: For two RoboClaw Motor Driver and GLUON-2L6-4L3 arm.
- **19V/15A - Power Layer**: For main processor (Intel NUC11PHKi7).

- **19V/15A - Distribution Layer**: For additional processor (NVIDIA Jetson AGX Xavier Developer Kit) and two laser scanners (SICK TiM561).
- **12V/15A - Power Layer**: For end-effector and LED lighting.

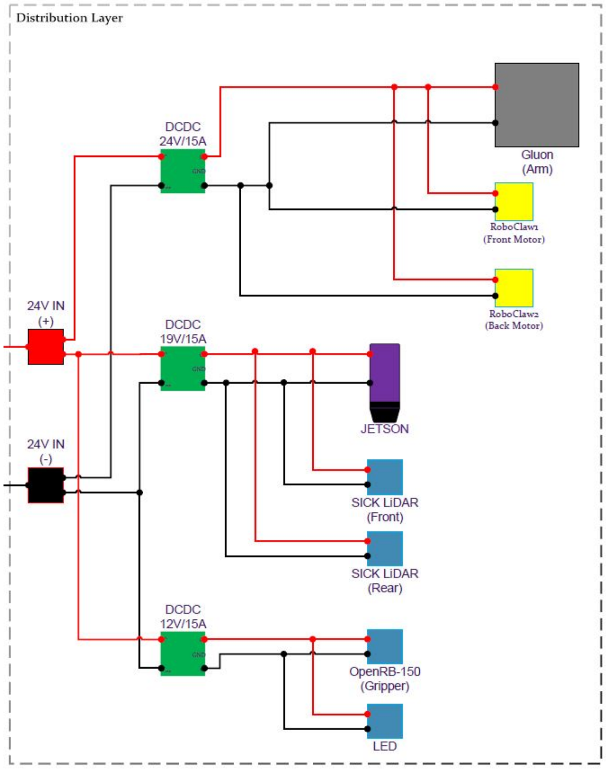


Fig. 3. Distribution Layer

3.2 Mechanical Design

SPlackie has a base dimension of 500 x 380 x 205mm (L x W x H) with 4-inch mecanum wheels added, which makes the robot's outer dimension of 500 x 443 x 225 mm (L x W x H). Refer to Figure 4 below for robot dimension drawing and 3D drawing of SPlackie.

The mechanical architecture consists of three layers mainly as Base Layer, Middle Layer and Top Layer. The Base Layer holds most of the electrical components of the robot such as DC Brushed Motor, Motor Controller, Arduino Mega, NVIDIA Jetson AGX Xavier Developer Kit, Laser Scanners and three-DCDC Converters. Next, Middle Layer consists of the main robot's supply such

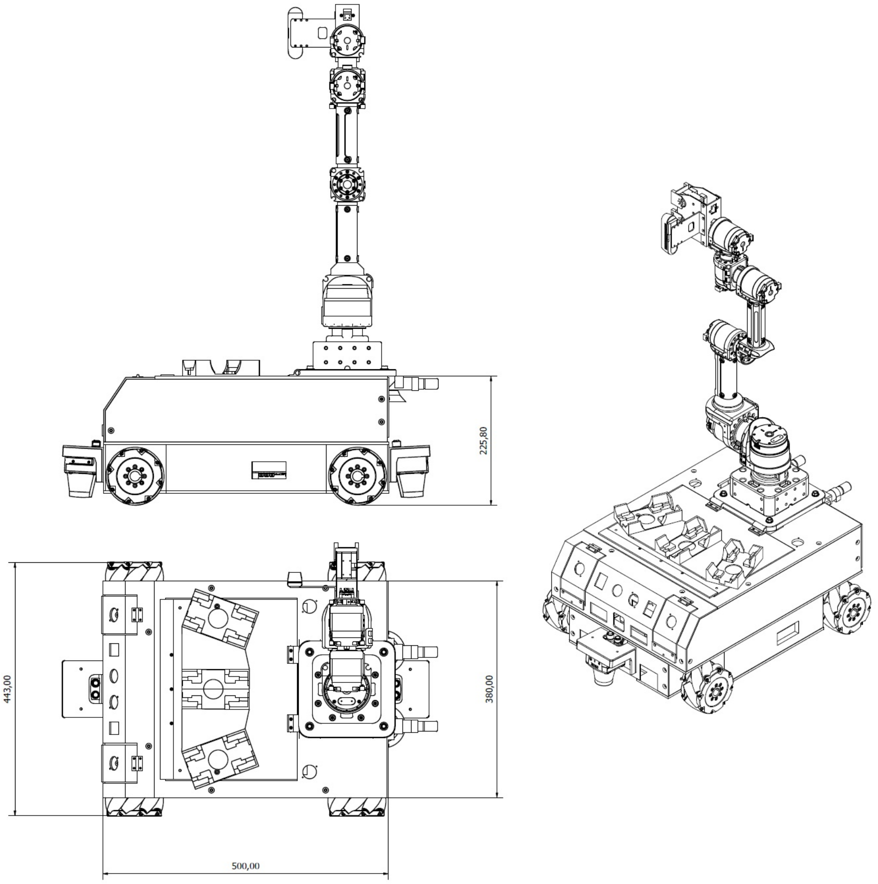


Fig. 4. Dimension Drawing of SPlackie

as Batteries and AC Power and it is where the main processor located. Last, arm manipulator and the tray where objects are placed is located at the Top Layer.

4 Software

SPlackie's software architecture is categorized into four major frameworks: Navigation, Perception (Vision), Manipulation, and Task Sorter. The overall software architecture is illustrated in Figure 5. Competition tasks originate from **atwork_commander** and are directed to the initial framework, **Task Sorter**. This framework manages and organizes the task sequence into a state machine. The **State Machine** assumes a pivotal role in the software architecture. It dispatches the sequence of goals and receives feedback from other frameworks

(**Navigation, Perception (Vision), and Manipulation**). Each framework becomes activated upon receiving goals from the State Machine.

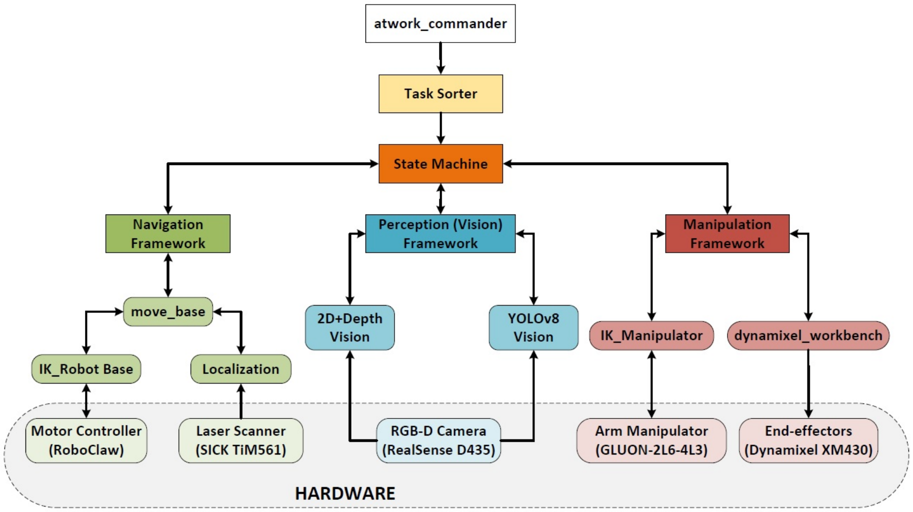


Fig. 5. Robo-Erectus@Work Software Architecture

4.1 Navigation Framework (Obstacle Avoidance)

The navigation framework is responsible for determining how the robot can safely reach its destination without colliding with obstacles in the shortest time possible. The robot's path planning involves a global planner for a static pre-recorded map, generated as the robot navigates through the arena using front and rear laser scanners. Additionally, there's a local planner designed for dynamic maps with moving obstacles.

SPlackie employs the Dynamic Window Approach (DWA) Planner [3] for local navigation and obstacle avoidance. This algorithm takes into account the robot's current state, including position and velocity, and continually generates a window of feasible angular velocities. Subsequently, the planner predicts all potential trajectories to avoid collisions with obstacles in the dynamic map. The optimal trajectory is then selected to guide the robot's movement, enabling it to steer away from obstacles effectively. Figure 6 below shows the example of the robot's trajectory avoiding obstacles in RVIZ on ROS environment.

4.2 Perception Framework (Empty Space Detection)

The perception framework empowers the robot to visually perceive, identify, and locate objects and various components within the arena, utilizing a vision system

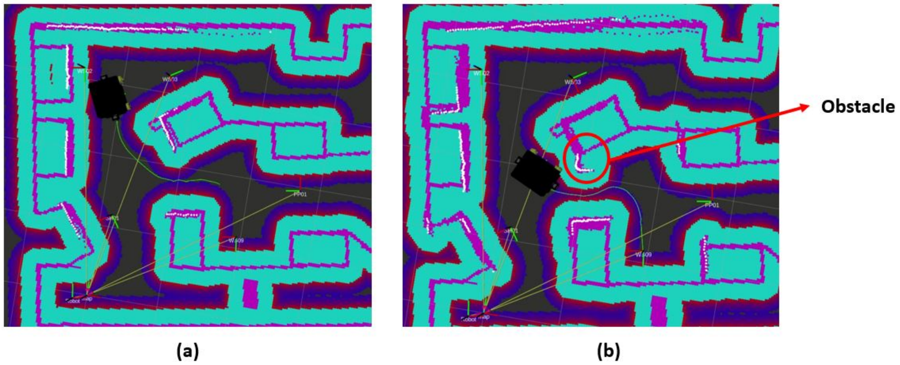


Fig. 6. (a) Normal Trajectory (b) Obstacle Avoidance Trajectory

and deep learning capabilities. In the face of dynamic conditions such as changing lighting and complex environments characterized by arbitrary surfaces and virtual walls, the task of enabling the robot to effectively perceive its surroundings becomes challenging. SPlackie has incorporated a deep learning algorithm employing the **YOLOv8** [4] [5] model to enhance the perception framework. This model supports object detection, cavity detection, container detection, and barrier tape detection. Recognized for its exceptional speed, accuracy, and learning capabilities, YOLOv8 is considered one of the fastest approaches to object detection.

One additional aspect of the perception framework involves detecting empty space. This functionality enables the arm manipulator to safely unload objects into the workstation without colliding with other objects. This action is crucial for the robot to avoid penalties during the unloading process. The detection of empty space employs the K-Dimensional Tree (K-D Tree) [6] to search for the nearest neighbors within the 3D point cloud plane generated by the vision system. For each point in the plane, it identifies the nearest neighbor within the same plane. Subsequently, based on a predefined threshold, the point is evaluated, and a radius search is conducted. If the number of points found exceeds the threshold, it is recognized as empty space; otherwise, the system continues the point iteration to search for empty space within the specified time frame. Figure 7 below illustrates the view from the RGB-D Camera.

4.3 Manipulation Framework

The proficiency of the arm manipulator in handling objects is of paramount importance for the robot's successful task completion. Given the diversity in workstation height, shelf configurations, and rotating tables, the development of the manipulation framework necessitates meticulous planning and close collaboration with both the perception and navigation frameworks.

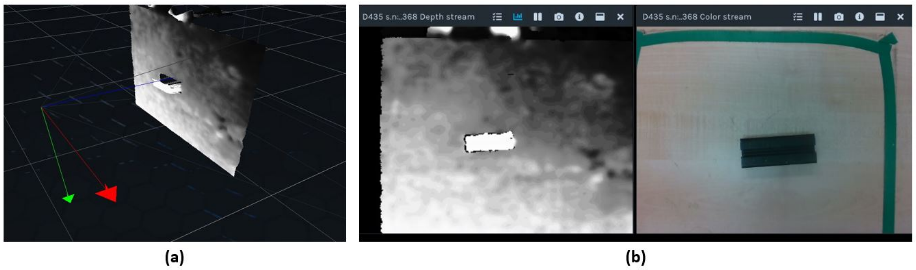
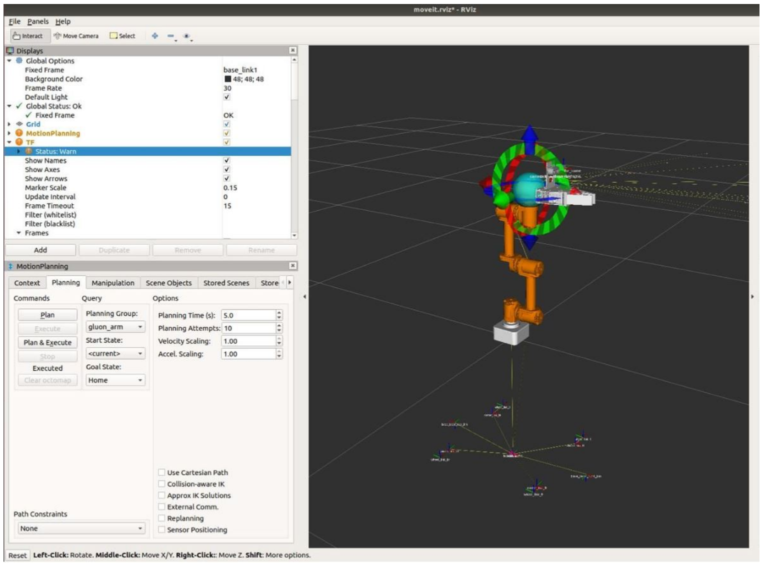


Fig. 7. (a) 3D View (b) 2D Depth View and 2D RGB View

In preparation for RoboCup 2024, SPlackie employs a new arm manipulator, the GLUON-2L6-4L3, and capitalizes on the MoveIt [7] framework for streamlined motion planning and execution. Refer to Figure 8 for a detailed illustration of the manipulation architecture. The manipulation framework acquires object information from the perception framework, and the TF2 library facilitates the transformation of the object's pose from camera coordinates to arm base coordinates. This transformation ensures that MoveIt can meticulously plan and execute precise movements, thereby enhancing the overall functionality and efficiency of the manipulation framework. Refer to Figure 9 for a detailed illustration of the manipulation architecture.



Global Options
Fixed Frame base_link1
Background Color 48; 48; 48
Frame Rate 30
Default Light ✓
Global Status: Ok
Fixed Frame OK
Grid ✓
MotionPlanning ✓
TF ✓
Status: Warn
Show Names ✓
Show Axes ✓
Show Arrows ✓
Marker Scale 0.15
Update Interval 0
Frame Timeout 15
Filter (whitelist)
Filter (blacklist)
Frames

Executed

Planning Group: gluon_arm
Start State: <current>
Goal State: Home

Planning Time (s): 5.0
Planning Attempts: 10
Velocity Scaling: 1.00
Accel. Scaling: 1.00

□ Use Cartesian Path
□ Collision-aware IK
□ Approx IK Solutions
□ External Comm.
□ Replanning
□ Sensor Positioning

Path Constraints
None

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click::** Move Z. **Shift**: More options.

Fig. 8. MoveIt framework with GLUON Arm Manipulator

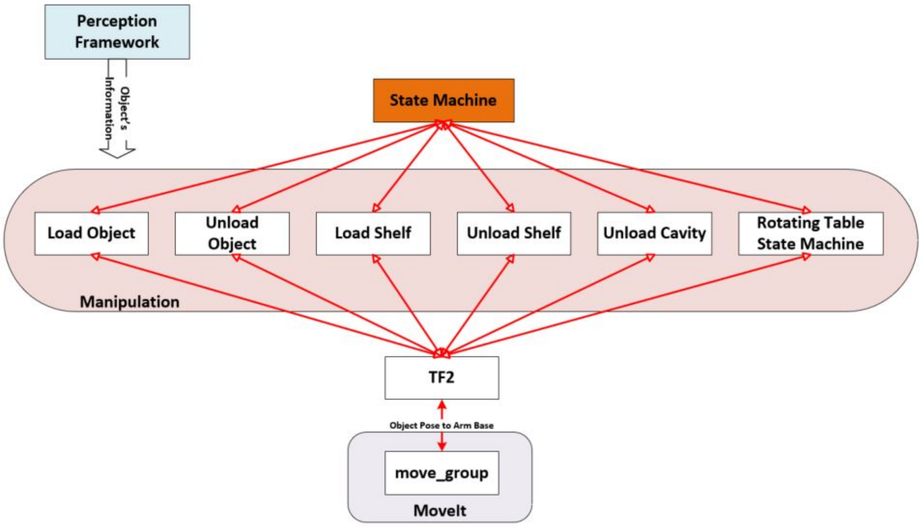


Fig. 9. Manipulation Architecture

5 Other Research Topic

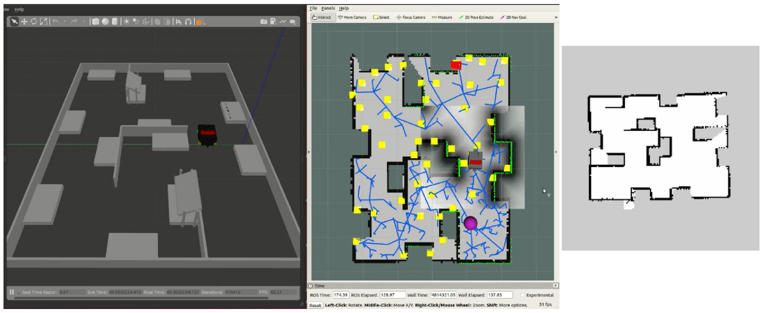
The team also explore few research topic in the area of Autonomous Mobile Robots (AMRs) in order to make the project relevant to Industrial environment, such as autonomous mapping system and fleet management system.

5.1 Autonomous Mapping System

To mapped the arena or unknown environment, the robot employs LiDAR sensor and moves around the arena to acquire information about the arena. This process requires human operator to manually control the robot and meticulously plan the robot movement with appropriate velocity so that the LiDAR sensor can provide data and feedback back to the robot's main system. It is a tedious process and require long time to achieved good quality map.

To ensure effective mapping, the team had worked on path planning algorithm, Rapidly Exploring Random Tree (RRT) [8] that can be integrated with SPlackie's navigation framework to perform autonomous mapping. The RRT is a data structure algorithm that efficiently searches for non-convex high-dimensional spaces, it constructs incrementally to quickly reduce the expected distance of a random chosen point to the tree. It can creates a path plan for the robot within the LiDAR sensor's vicinity to all reachable distance endpoints which are called frontier points. This allows the robot to map new unexplored region continuously as it moves towards the different frontier points.

Figure 10 below shows the operation of RRT Algorithm in simulation environment based on the arena of RoboCup @Work. As showed, the robot able to navigate and map the simulated arena and produce the map that is close to actual layout of the arena.



ROS Time: 174.39 ROS Elapsed: 128.97 Wall Time: '4814321.05 Wall Elapsed: 137.85 □ Experimental

Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options. 31 fps

Fig. 10. Autonomous Mapping in Simulation

5.2 Fleet Management System

With multiple AMRs robots operating within factory, the robots have to coordinate to ensure effective and efficient running of the operation in the factory. Fleet Management System [9] allows the coordination of the robots' movements and tasks, which ensure that the robots are moving in the most efficient ways possible, avoiding collisions and making sure that the robots are working on the most important tasks at any given time. The coordination typically managed through a central control system, which can be programmed with parameters such as tasks priority, robots' locations, robots' status, etc.

Figure 11 below shows multiple robots in the arena performing loading and unloading tasks, which controlled by central control system to effectively and efficiently performing the tasks.

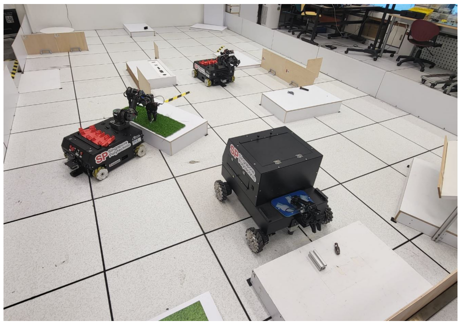


Fig. 11. Fleet Management System in Arena

6 Conclusion

SPlackie boasts an enhanced robotic foundation featuring a stable mechanical structure, hardware architecture, and a meticulously designed low-level motion controller. The robot facilitates a reliable and precise coordination between its motion control and artificial intelligence system. Employed in research and development collaborations with various technologies in a semi-outdoor setting, the robot also serves educational purposes by imparting knowledge in the fields of vision systems, artificial intelligence, deep learning, manipulation systems, and the ROS environment.

The team anticipates the SPlackie's stellar performance at the upcoming RoboCup 2024 in Eindhoven, Netherlands. Additionally, they eagerly anticipate the opportunity to exchange knowledge, experiences, and technology with participating teams from around the world.

7 Acknowledgement

The team extends gratitude to both past and present project team members responsible for the development of the SPlackie robot, as well as the current students and Technical Staff from the School of Electrical and Electronic Engineering at Singapore Polytechnic. A special acknowledgment is reserved for the higher management of Singapore Polytechnic, whose unwavering support has been instrumental in the continuous progress and development of Team Robo-Erectus@Work.

References

1. Willow Garage, ”Robot Operating System”, https://www.ros.org, 2007
2. GLUON-6L3, http://wiki.mintasca.com/, 2019
3. Dieter Fox, Wolfram Burgard and Sebastian Thrun, ”The Dynamic Window Approach to Collision Avoidance”, vol. 1, p.8, 1997.
4. Glenn Jocher, ”YOLOv8 in PyTorch”, https://github.com/ultralytics/ultralytics, 2023
5. Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi, ”You Only Look Once: Unified, Real-Time Object Detection”, p. 779-780, 2016
6. Aashish Barnwal, ”Search and Insertion in K Dimensional Tree”, https://www.geeksforgeeks.org/search-and-insertion-in-k-dimensional-tree/, 2020
7. Ioan A. Sucan and Sachin Chitta, ”MoveIt - Moving robots into the future”, https://moveit.ros.org/, 2014
8. Michael Otte and Emilio Frazzoli, ”RRTX: Real-Time Motion Planning/Replanning for Environment with Unpredictable Obstacles”, p.1-4, 2017
9. Vecna Robotics, ”What is Mobile Robot Fleet Management?”, https://www.vecnarobotics.com/resources/mobile-robot-fleet-management/, 2023