Development of an Autonomous Mobile Manipulator for Pick and Place Operation Using 3D Point Cloud

Sheng-Wei Sim
Department of Mechatronics and Biomedical Engineering, Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Sungai Long, Selangor, Malaysia
username7117@1utar.my

Ban-Hoe Kwan
Department of Mechatronics and Biomedical Engineering, Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Sungai Long, Selangor, Malaysia
kwanbh@utar.edu.my

Wun-She Yap
Department of Electrical and Electronics Engineering, Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Sungai Long, Selangor, Malaysia
yapws@utar.edu.my

Danny Wee-Kiat Ng
Department of Mechatronics and Biomedical Engineering, Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman
Sungai Long, Selangor, Malaysia
ngwk@utar.edu.my

2022 IEEE 10th Conference on Systems, Process & Control (ICSPC) | 978-1-6654-7098-8/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICSPC55597.2022.10001817

***Abstract*—Automated Guided Vehicles (AGV) are commonly used for tedious processes such as materials transportation and sorting. However, the function of an AGV is quite limited without a manipulator such as a robotic arm. A mobile manipulator can be formed by pairing up both the robotic arm and AGV, which has higher flexibility to conduct different types of tasks. The navigation of AGV is subjected to a certain degree of inaccuracy that needs to be overcome by applying specific compensation techniques such as 3D point clouds for object segmentation. Therefore, this paper will illustrate a solution to build a ROS2-based autonomous mobile manipulator with autonomous navigation, workpiece detection and robotic arm path planning and manipulation.**

***Keywords*—Robot Operating System 2, ROS2, AGV, mobile manipulator, 3D points clouds**

I. INTRODUCTION

As industrial robots are widely adopted in manufacturing sectors, their control techniques are essential to ensure the efficiency and safety of the machines. The presence of industrial robots helps to liberate humans from high-risk or repetitive tasks. An industrial robot is defined by ISO 8373:2012 as follows: "An automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes, which can be either fixed in place or mobile for use in industrial automation applications [1]. The mechanical structure of industrial robots can be categorised into a fixed-base system or mobile platform, enabling robots to adapt to different tasks and working scenarios. The preliminary part of an industrial robot refers to the robot's actuators and respective control techniques [2].

Compared to a traditional fixed-base robotic system, a mobile platform robotic system has higher heterogeneity and mobility in applications. A mobile platform robotics system is commonly built by pairing up a mobile robot with a manipulator or robotic arm. The movement of the manipulator is complex due to the high degree of freedom (DoF), and the localisation error on the mobile base also happened because of imperfection on odometry sensors and actuators [3]. Therefore, specific compensation or corrective measures were needed to ensure the accuracy of the robot's motions.

However, some concerns need to be considered when developing a mobile manipulator, such as accurate robot localisation and navigation, accurate object manipulation for tasks, and system coordination. All topics mentioned above are essential to developing a fully autonomous mobile manipulator with high accuracy and efficiency [4].

Therefore, this paper emphasises the method to develop a functional autonomous mobile manipulator for application in factory and warehouse automation. A 6 DOFs robotics arm was installed on Autonomous Guided Vehicle (AGV) as a mobile base to form a mobile manipulator system. Technology such as autonomous navigation, machine vision, and motion planning algorithms were implemented in the system. A pick-and-place task was tested with the developed robotic system.

II. LITERATURE REVIEW

A. Mobile Manipulator

A mobile manipulator is represented by a robotic system that pairs up an industrial robotic arm and manipulator with an autonomous mobile robot as a mobile base. Compared to a traditional fixed-based robotic system, a mobile manipulator provides higher flexibility, benefiting from the mobility of an autonomous mobile robot [6]. So, the ability of the robot to interact with the environment can be gradually improved [5].

In order to achieve a fully autonomous mobile manipulator system, the core principle of "see-think-act" should be implemented and followed by the robotic system [2]. Therefore, some main functions cannot be neglected when building up a mobile manipulator, which are:

- Perception

- High-level instruction and context-aware task execution
- Knowledge acquisition and generalization
- Adaptive planning

Perception is the eye that "sees" through sensory data processing, representation, and interpretation. Key elements within a working environment can be determined with the perception to carry out the desired motion. Besides, robots need the capability to acquire new knowledge and generalise them to enable them to perform a new task based on the history of the task and decision. Meanwhile, the high-level instruction and context-aware task execution will define the specific constraint for task execution based on the application scenario. Adaptive planning gets the knowledge to perform motion planning for the manipulator. It will predict the events and prepare to deal with them in advance, as well as deal with any unforeseen situations. By linking up these modules, robots can determine a series of actions and choose the best solution to ensure the effectiveness of the system. These three components will form a mobile manipulator system's "think" function. "Act" refers to continuous control and monitoring of the manipulator to ensure a correct motion is performed according to the plan.

An excellent industrial mobile manipulator should be able to navigate autonomously in a dynamic environment. From the first mobile manipulator in 1984, mobile manipulators started to show their potential in logistics and assistive tasks in industrial environments. From simple tasks such as pick and place, mobile manipulators nowadays can perform complex tasks such as logistics sorting, part feeding, large-scale inspection, etc. Recent projects focused on perception, motion planning and grasp pose detection to improve the performance of mobile manipulators in a known or unknown environment [7].

The final goal of the study on mobile manipulators is to develop a fully autonomous mobile manipulator system that can carry out their tasks without human supervision or instruction.

B. Sensing

A mobile manipulator needs to detect the environment, as physical interaction between the manipulator and the surrounding environment is required. Therefore, specific sensors are needed for the manipulator to perceive the surroundings correctly and accurately [8].

1) Elevated sensors: Similar to human eyes, an elevated sensor is located on top of the body to get the best possible impression of the surroundings. It serves for strategic ahead path planning or obstacle avoidance. Light detection and ranging (LiDAR), stereo cameras and RGBD cameras are commonly used in this aspect.

2) Touch sensors: It used to compensate for the limitation of cameras such as point of view, occlusion, and light condition. It also promotes the safety issue of the mobile manipulator. A physical sensor is preferable when dealing with highly dynamic processes

3) Base sensors: Commonly used for map building in an unknown environment or navigating the robot. This process is commonly known as Simultaneous Localisation and Mapping (SLAM). The standard sensor used in this aspect is laser rangefinders, LiDAR or ultrasonic-based. The accuracy of this sensor also directly affects the accuracy of SLAM.

C. Perception

Robotic perception is essential for a robot to operate correctly in a real-world environment. In an autonomous robot, robot perception is needed for obstacle detection, object recognition, 3D environment description, object tracking, plane segmentation, environment change detection, etc. The data for building robotic perception can be obtained from onboard sensors or other infrastructure. It can also consist of more than one sensor data source.

Robotic perception is commonly used to compile the environment representation or map for mobile robot applications. It enables the robot to localise and navigate autonomously in the environment while avoiding obstacles. Besides, object detection and segmentation also help in tasks such as pick-and-place, where robots need to identify the location of the workpiece relative to the end effector [9].

Sensor representation is vital for a robotic perception system. The output of the perception system will directly affect the behaviour of the mobile manipulator in the real world. In short, robotic perception translates sensor data or images into manipulator actions [10].

D. RGBD Camera

An RGBD camera is a depth camera that provides depth (D) and colour (RGB) data as the output in real-time. Depth information is retrievable through a depth map or image created by a 3D depth sensor such as a stereo sensor or time of flight sensor. RGBD cameras can merge RGB data and depth information to deliver both in a single frame.

In mobile manipulator applications, the RGBD camera is mainly used for workpiece detection coupled with a suitable computer vision algorithm. In project [11], a Microsoft Kinect Depth camera was used to detect and segment the workpiece by applying Point Cloud Library (PCL) for the welding process that joined two workpieces with a 6 DOF robotic arm. Besides, the project [12] also uses a Microsoft Kinect Depth Camera to perform box detection and segmentation. The position of boxes is predicted through a machine learning algorithm and passed to the robotic arm to perform sorting action

E. MoveIt! and Motion Planning

MoveIt! is a robotic package integrated with Robot Operating System (ROS) designed for mobile manipulation. Environment representation is built based on the robot's three-dimensional or other sensor data. After that, the motion of the mobile manipulator is generated and executed with constant monitoring for changes in the environment.

The environment can be represented in 2 formats: a voxel grid representing obstacles in the environment and geometric primitives and a mesh model representing the object that can be recognised through an object detection algorithm. The environment model is the main input to the motion planner to generate a collision-free path in the environment provided. It can couple with multiple motion planners such as randomised planners from Open Motion Planning Library (OMPL), search-based planners from the Search-Based

Planning Library (SBPL), trajectory optimisation libraries and stochastic trajectory optimisation for motion planning (STOMP). Therefore, MoveIt! can control a robotic arm in either a joint space goal or a cartesian goal [13].

Motion Planning refers to the process of determining a continuous path that relates the starting state and the goal pose in a robotic system with certain constraints added. A good motion planner provides a short planning process and optimal solutions are essential for mobile manipulators. OMPL is built for sampling-based motion planning. This library enables users to solve a series of complex motions quickly with minimal input required. It is widely implemented in robotic applications such as search and rescue robots, pick and place and service robots [14].

III. METHODOLOGY

In this paper, a mobile manipulator was assembled by combining an AGV as a mobile base and a robotic arm as a manipulator. The setup aims to transport magazines between the loading machine, unloading machine and storing racks in a semiconductor manufacturing factory. The AGV has the capability to SLAM and navigates with static and dynamic obstacle avoidance to prevent any collisions occur. A custom end effector was built on the manipulator, and an RGBD camera was installed to detect the workpiece.

The whole robotic system will operate based on Robot Operating System 2 (ROS2) hosted by Intel NUC. Users can send the task command remotely to the mobile manipulator, and the mobile manipulator should conduct the task accordingly without error.

A. Hardware Setup

Zalpha AGV from DF Automations and Robotics Sdn Bhd is used as a mobile platform, while robotic arm UR10e from Universal Robot is used as the main manipulator. An Intel NUC is used as a host to coordinate and control the motion for both the mobile base and the manipulator. All these three devices are located under the same Local Area Network (LAN). With the capabilities of ROS2 that all the ROS2 nodes can discover each other under the same network, the communication between the Zalpha AGV, UR10e and Intel NUC can be achieved through ROS2 nodes.

Fig. 1. Mobile manipulator combining Zalpha AGV and UR10e.

B. Point Cloud Library

Point Cloud Library (PCL) is built for 3D point cloud processing, including filtering, feature estimation, surface reconstruction, model fitting, segmentation, registration, and so on [15]. As mentioned above, the RGBD camera, Intel Realsense D435, is used to track the workpiece to compensate for the navigation error. As the RGBD camera can produce 3D point clouds, point cloud processing is needed to segment the workpiece for an accurate detection result.

Under many libraries available under PCL, pcl_sample_consensus is applied for this project. The pcl_sample_concensus library holds SAmple Consensus (SAC) methods like RANSAC for model detection such as planes and cylinders. These can be combined to detect specific models and their parameters in point clouds. This library's models include lines, planes, cylinders, and spheres. Plane fitting is often applied to detect joint indoor surfaces, such as walls, floors, and tabletops. SACMODEL_PLANE model under this library will be adopted as it can be used to determine plane models and return four plane coefficients to their Hessian Normal form.

C. Software Architecture

ROS2 is a server to host the process and coordinates the motion between UR10e and Zalpha AGV to form a functional mobile manipulator. The process of task execution, manipulator motion planning and execution is emphasised as Zalpha AGV comes with the capability of SLAM and autonomous navigation.

Firstly, a ROS2 wrapper is built for Zalpha AGV to control the AGV through the provided API. The wrapper should be able to start a task execution and monitor the progress. When the AGV reaches its destination, it needs to signal the robotic arm to start its motion.

According to the documentation provided by the manufacturer, the Zalpha AGV has a stopping accuracy of ±10mm, ±2º on magnetic tape and ±100mm, ±2º on visual navigation. This is a lot for mobile manipulation as the workpiece must load and unload from the machine has a fixed position. Therefore, an RGBD camera is needed for workpiece recognition and positioning to compensate for the navigation error.

MoveIt2! It was used for path planning and execution for UR10e. After receiving the signal from Zalpha AGV, the arm will move to a fixed position to scan and track the workpiece. If the detection is successful, the coordination of the workpiece related to the camera position will be sent to MoveIt2! action server, and the path will be planned and executed accordingly. As the arm is attached to a mobile platform, it is crucial to ensure the path generated is valid and not colliding with the AGV body. A URDF robot model needs to be compiled and fed into MoveIt2! action server to ensure that the path generated by MoveIt2! is valid.

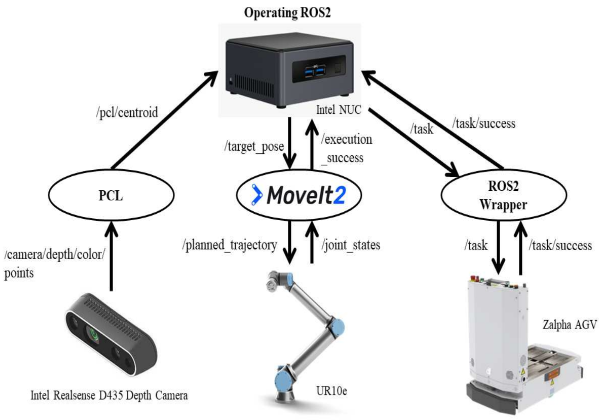


Fig. 2. Overview of proposed framework.

IV. Result and Discussion

An autonomous mobile manipulator is successfully developed by applying the proposed framework. Creating a relevant task template inside the AGV navigation system allows AGV to navigate to the destination point autonomously. After reaching the destination, the robotic arm will extend to a fixed position to detect the workpiece. The detection and segmentation of the workpiece are performed, and the centroid of the detected workpiece will be sent to MoveIt2! action server. The end effector will grab the workpiece, retract the arm, and navigate to the drop-off station. Through this, the inaccuracy of the navigation of AGV can be compensated.

A. Robot Model and MoveIt2!

A URDF was generated to represent the whole robot model for robotic arm collision detection while doing path planning. The AGV and end effector model was added to simulate the mobile manipulator's actual setup. The generated robot model is fed into MoveIt2! for joint manipulation and collision awareness, as shown in Fig 4.

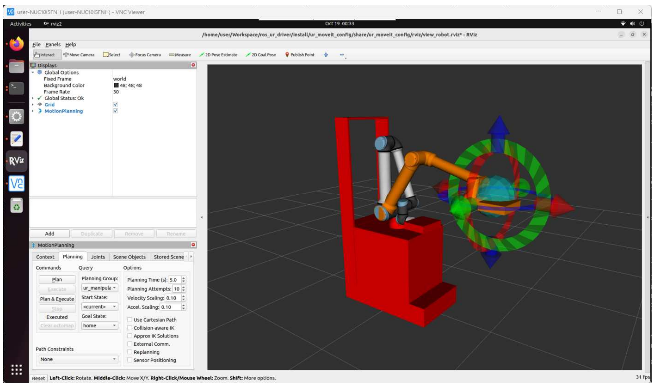


Fig. 3. Rviz window for robot model and MoveIt2! graphical interface. (Goal pose request shown in orange colour)

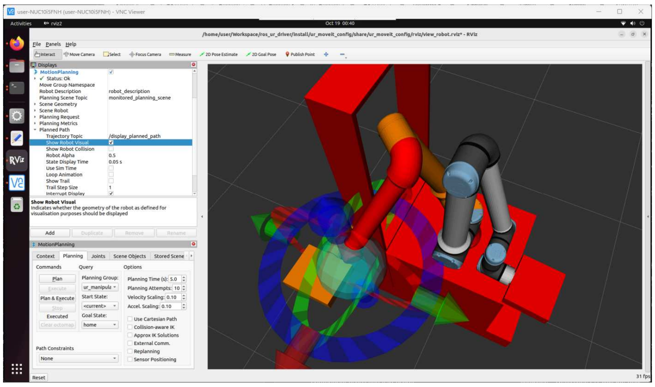


Fig. 4. Rviz window shows collision detection between UR10e and Zalpha AGV. (The red colour parts show collision occur)

Two types of goal requests are being applied in this project: setJointValueTarget () for extending the arm to a fixed position and computeCartesianPath() for cartesian path planning to approach the workpiece. With a proper robot model provided, the path planning and execution process successfully ran on MoveIt2! Fig 5 shows the result of the path planning process.

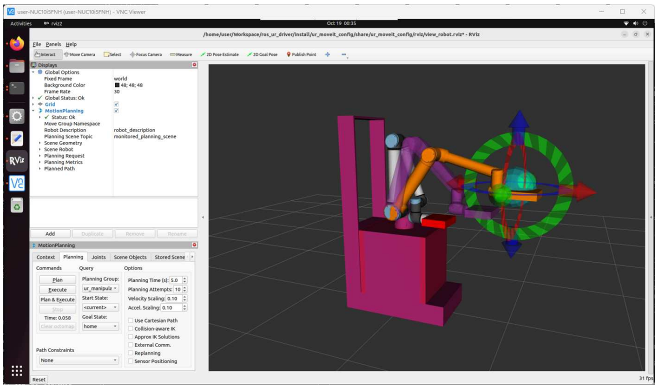


Fig. 5. Rviz window shows the result of path planning. (The planned path shows as animation in purple colour)

B. PCL Plane Segmentation

A workpiece detector node was developed with PCL by taking in input from the Intel Realsense D435 depth camera. A rectangular workpiece can be detected with the corner and centroid of the workpiece being identified. Fig 7 shows the result for PCL segmentation with corners and centroid of the workpiece being identified. Intel Realsense D435 can be applied in both indoor and outdoor applications with a sensing range from 0.3 meters to 3 meters with a depth accuracy of <2% at 2 meters. The maximum resolution of the output depth stream is up to 1280 × 720 with 90 frames per second. This provides high-quality input depth point clouds for the segmentation process.

A size filter was implemented in the detector function to prevent the wrong object from being identified. The detector will only detect an object with specific size and height. Besides, a ROS2 node was created, which subscribes to depth camera output, perform the PCL segmentation, and publish out the centroid of the detected workpiece.

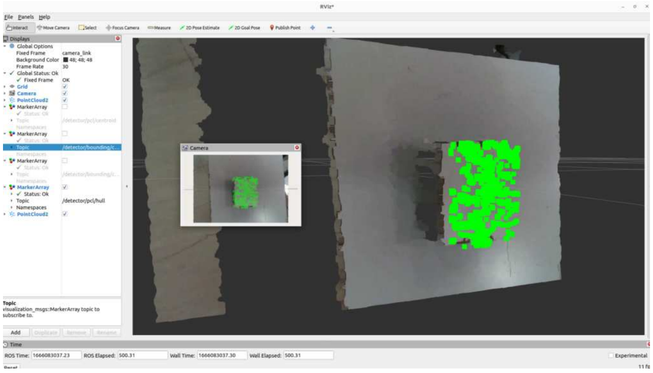
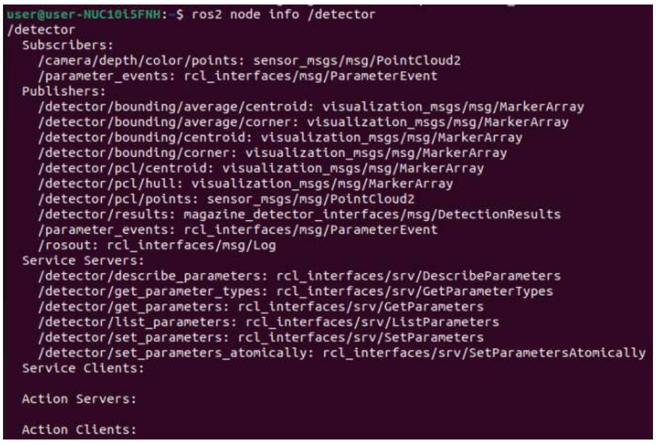


Fig. 6. Rviz window shows the result of PCL 3D plane segmentation.

Fig. 7. Result of PCL segmentation that able to identify workpiece corners and centroid.



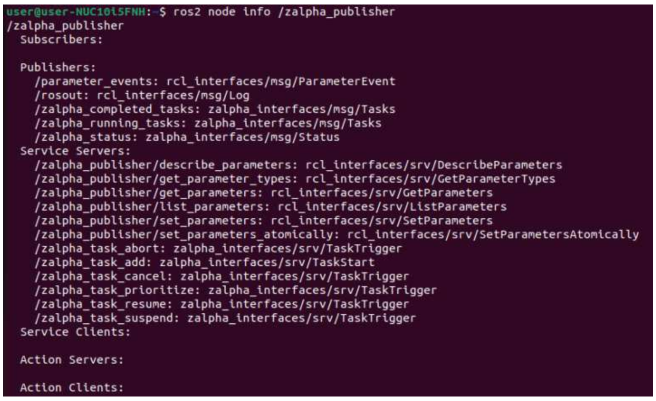
```
user@user-NUC10i5FNH:~$ ros2 node info /detector
/detector
  Subscribers:
    /camera/depth/color/points: sensor_msgs/msg/PointCloud2
    /parameter_events: rcl_interfaces/msg/ParameterEvent
  Publishers:
    /detector/bounding/average/centroid: visualization_msgs/msg/MarkerArray
    /detector/bounding/average/corner: visualization_msgs/msg/MarkerArray
    /detector/bounding/centroid: visualization_msgs/msg/MarkerArray
    /detector/bounding/corner: visualization_msgs/msg/MarkerArray
    /detector/pcl/centroid: visualization_msgs/msg/MarkerArray
    /detector/pcl/hull: visualization_msgs/msg/MarkerArray
    /detector/pcl/points: sensor_msgs/msg/PointCloud2
    /detector/results: magazine_detector_interfaces/msg/DetectionResults
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
  Service Servers:
    /detector/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /detector/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /detector/get_parameters: rcl_interfaces/srv/GetParameters
    /detector/list_parameters: rcl_interfaces/srv/ListParameters
    /detector/set_parameters: rcl_interfaces/srv/SetParameters
    /detector/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
  Service Clients:

  Action Servers:

  Action Clients:
```

Fig. 8. ROS2 node info for /detector node created for workpiece detection.

An experiment had been set up with a white colour workpiece placed on the white colour table. Figures 6 and 7 show that the segmentation process is carried out successfully, which can detect the workpiece. The corner and centroid of the detected plane can also be identified correctly. In order to have a more accurate result, an average of 10 detection results is gathered to obtain an average centroid reading and sent to Moveit2! Action server for cartesian path planning and execution.

C. Zalpha AGV to UR10e communication and coordiantion

The communication between Zalpha AGV and UR10e was done through the ROS2 node named /zalpha_publisher. This node is responsible for tracking the task execution process of Zalpha AGV. Topic named /zalpha_status will be the reference status for UR10e; the motion on UR10e will start only after the task for Zalpha is completed.



```
user@user-NUC10i5FNH:~$ ros2 node info /zalpha_publisher
/zalpha_publisher
  Subscribers:

  Publishers:
    /parameter_events: rcl_interfaces/msg/ParameterEvent
    /rosout: rcl_interfaces/msg/Log
    /zalpha_completed_tasks: zalpha_interfaces/msg/Tasks
    /zalpha_running_tasks: zalpha_interfaces/msg/Tasks
    /zalpha_status: zalpha_interfaces/msg/Status
  Service Servers:
    /zalpha_publisher/describe_parameters: rcl_interfaces/srv/DescribeParameters
    /zalpha_publisher/get_parameter_types: rcl_interfaces/srv/GetParameterTypes
    /zalpha_publisher/get_parameters: rcl_interfaces/srv/GetParameters
    /zalpha_publisher/list_parameters: rcl_interfaces/srv/ListParameters
    /zalpha_publisher/set_parameters: rcl_interfaces/srv/SetParameters
    /zalpha_publisher/set_parameters_atomically: rcl_interfaces/srv/SetParametersAtomically
    /zalpha_task_abort: zalpha_interfaces/srv/TaskTrigger
    /zalpha_task_add: zalpha_interfaces/srv/TaskStart
    /zalpha_task_cancel: zalpha_interfaces/srv/TaskTrigger
    /zalpha_task_prioritize: zalpha_interfaces/srv/TaskTrigger
    /zalpha_task_resume: zalpha_interfaces/srv/TaskTrigger
    /zalpha_task_suspend: zalpha_interfaces/srv/TaskTrigger
  Service Clients:

  Action Servers:

  Action Clients:
```

Fig. 9. ROS2 node info for /zalpha_publisher

Besides, a node, /zalpha_ur_controller, was created to gather the information from UR10e, PCL segmentation and Zalpha AGV. It is a simple state machine controller to determine the motion of the mobile manipulator depending on the task status of each part of the machine. It signals the robotic arm to start its motion once the AGV reach the destination, and it will also get the result from PCL segmentation to determine the relative position of the workpiece with the end effector and plan and valid cartesian path to approach the workpiece. Fig 10 shows the overall simplified ROS2 nodes connection for the developed framework.

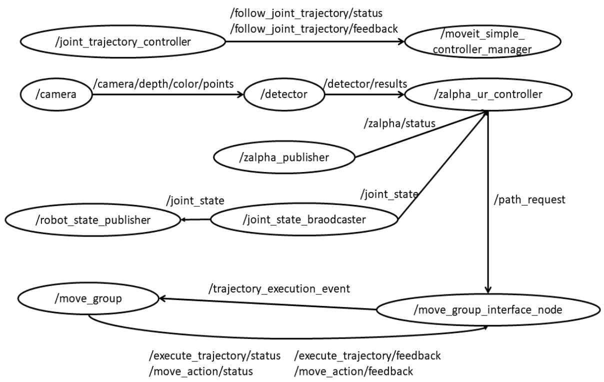


Fig. 10. Overall ROS2 node graph of the framework

D. Discussion

With the proposed framework, the user can start a task by sending relevant task details needed by AGV through ROS2 topics. After receiving a task, AGV will navigate to the designated place, and the robotic arm will be extended to a fixed position. Workpiece detection will be carried out using a depth camera, and segmentation will be done through PCL to get the centroid of the workpiece relative to the camera position. This process aims to compensate for the navigation inaccuracy of AGV. After that, a valid cartesian path will be planned, and the arm will approach the workpiece, which is controlled by MoveIt2! The workpiece can be grabbed, and the arm will retract, move to the drop-off station, and release the workpiece.

However, the current segmentation result still can be improved to produce a more stable and accurate detection for the centroid of the workpiece. Tasks needed to be keyed in manually by humans to trigger the motion currently, which also can be improved through on-station work progress detection.

V. Conclusion

A mobile manipulator is created by installing a UR10e robotic arm on Zalpha AGV. As the AGV has particular navigation inaccuracy, which will cause the robotic arm fails to grab the workpiece, this error is compensated by applying PCL segmentation to determine the center of the workpiece relative to the robotics arm's end effector. The path is planned through OMPL and will be executed accordingly through MoveIt2! The mobile manipulator can perform this once it receives a task request from the user.

In future work, a multi-robot coordination strategy can be introduced into the framework, as there should be multiple mobile manipulators that will work together in a factory environment. The accuracy of workpiece detection can also be improved by applying filters of object detection and semantic segmentation neural networks.

Acknowledgement

This study was funded by UTAR Research Fund from the Universiti of Tunku Abdul Rahman (UTAR) (IPSR/RMC/UTARRF/2021-C1/K03) and supported by Centre for Healthcare Science & Technology, UTAR (Vote account: 6297/001).

References

[1] ISO 8373:2012; Robots and Robotic Devices—Vocabulary; International Organization for Standardization: Geneva, Switzerland, 2012.

[2] J. Arents, and M. Greitans, "Smart Industrial Robot Control Trends, Challenges and Opportunities Wirhin Manufacturing," Appl. Sci. 2022.

[3] I. Akli, B. Bouzouia, and N. Achour, "Motion analysis of mobile manipultor executing pick-up tasks," Computers and Electrical Engineering 45, pp 257-269, February 2015.

[4] Miškovi´c, D.; Mili´c, L.; Cilag, A.; Berisavljevi´c, T.; Gotshaber, ˇ A.; Rakovi´c, M. "Implementation of Robots Integration in Scaled Laboratory Environment for Factory Automation," Appl. Sci, December 2022.

[5] R. Xu, L. Qian and X.W. Zhao, "Development of dual-arm mobile robot platform based on ROS," Cobot 2022, July 2022 DOI: https://doi.org/10.12688/cobot.17457.1

[6] H. Engemann, S.Z. Du, S. Kallweit, P. Conen, and H. Dawar, "OMNIVIL-An Autonomous Mobile Manipulator for flexible Production," Sensors 2022, 20,7249, December 2022.

[7] Rofalis, N.; Nalpantidis, L.; Andersen, N.A.; Krüger, and V, "Vision-based robotic system for object agnostic placing operations," In Proceedings of the International Conference on Computer Vision Theory and Applications, Rome, Italy, 27–29 February 2016, pp. 465–473.

[8] M. Sereinig, W.Werth, and L.M. Faller, "A review of the challenges in mobile manipulation: systems design and RoboCup challenges," Elektrotechnik & Informationstechnik (2020) 137/6: pp. 297–308. DOI: https://doi.org/10.1007/s00502-020-00823-8

[9] C. Premebida, R. Ambrus, and Z.C. Marton, "Intelligent Robotic Perception Systems," Intech Open, Mobile Robots, Volume 1, 2018.

[10] R. Galin, and R. Mesheryakov, "Collaborative Robots: Development of Robotic Perception System, Safety Issues, and Integration of AI to Imitate Human Behavior," September 2020, DOI: 10.1007/978-981-15-5580-0_14

[11] V. Patil , I. Patil, V. Kalaichelvi, and R. Karthikeyan, "Extraction of Weld Seam in 3D Point Clouds for Real Time Welding Using 5 DOF Robotic Arm," 2019 5th International Conference on Control, Automation and Robotics.

[12] J. Medrano, F. Yumbla, G. Lee, et al, "Box segmentation, position and size estimation for robotic box handling applications," Jeju, Korea, July 2022 [19th International Conference on Ubiquitous Robots (UR)]

[13] S. Chitta, I. Sucan, and S. Cousins, "Moveit!," EEE Robotics & Automation Magazine, March 2012, DOI: 10.1109/MRA.2011.2181749

[14] I. A. Sucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," IEEE Robotics and Automation Magazine, December 2012.

[15] R. B. Rusu, and S. Cousins, "3D is here: Point Cloud Library (PCL)," 2011 IEEE International Conference on Robotics and Automation (ICRA).