Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

A Hybrid Navigation and Image Processing Model with Dynamic Mobile Manipulation

Shang Gao[1], Biqiao Wang[1], Peiqi Wang[1], Bangzhi Yan[1], Chuting Wang[1], Chaomin Luo[1] and Mariam Faied[1]

***Abstract*— A mobile manipulator is widely used for many scenarios such as warehouse management, and disaster rescue missions. The autonomy of robots operating in an unknown environment is critical. In this paper, a dynamic mobile manipulation robot platform based on the Pioneer 3-AT robot, equipped with one 6-DOF manipulator and multiple sensors is designed and tested. This autonomous mobile manipulator robot platform functions with mapping, path-planning, obstacle avoidance, image processing, color-based object detecting and manipulation algorithms running in the Matlab-ROS environment. In addition, Arduino-based devices are used to enable manipulation control and other future add-on devices. The platform is operating in an unknown dynamic environment using a local path planning algorithm for map exploration, color-based object detection, and manipulation tasks. The project aims to build a platform that is universally suitable for various task situations, as well as easy and efficient to modify and test algorithms with the advantage of data processing capability provided by Matlab. The modification for integrating Pioneer 3-AT robot and Trossen Widow X robotics arm is introduced on both hardware and software levels.**

Fig. 1. Dynamic Matlab-ROS mobile manipulator platform

I. Introduction

Recently, the desired working environment for robots has become various. In most of the robotics assisting missions, a mobile manipulator suits the task best because of its combination of locomotion and manipulation capabilities. The mobile manipulator robot is widely used in tasks such as rescue missions, where the working space is difficult or dangerous for humans to enter due to radiation, toxic fumes or unstable environment. Deploying mobile manipulator robots reduces risk. A typical mobile manipulator for search and rescue tasks is usually teleoperated by human operators. The robot has a higher degree of freedom for performing tasks in the complex environment. This increases the difficulty of control and mental stress for human operators. The teleoperation of a high number of degrees of freedom system is slow and requires a high-quality communication system. It is especially difficult in an unknown environment for search and rescue tasks. However, full autonomy for search and rescue robots is difficult and dangerous with current technology, especially in unknown environments.

Semi-autonomy is proposed which provides robot autonomy on controlling lower-level tasks such as joint space control. And human operators could focus on making high-level decisions. There are lots of researches conducted in this field. The project by Miyasaka et al. introduces a pipeline to protect robot end-effector based on force sensing during a mobile manipulator robot performing manipulation [1]. Another research done by Schnaubelt et.at focuses on autonomous assistance for versatile grasping of rescue mobile manipulator robot [2]. There are also researches on mobile base autonomy with both global and local path planning algorithm, such as A* [3], Vector Field Histogram (VFH) [4], and Dynamic Window Approach (DWA) [5]. Yang and Luo [6] proposed a neural network model for robot path planning in non-stationary environments. However, the current knowledge of the non-stationary environment was assumed to be completely known [7]. Nowadays, some bio-inspired intelligence algorithms are applied to robot mapping and navigation [8], [9], [10].

There are also some researchers working on semi-autonomy between locomotion and manipulation. Nagatani et al. proposed a motion planning algorithm that keeps the manipulability of the manipulator by adjusting mobile base motion accordingly [11]. Okabe et al. introduced a semi-autonomy control pipeline for mobile manipulator rescue robots, which allows human operators to input a desired grasping pose in Cartesian space. The robot could plan the path with obstacle avoidance, drive the mobile with moving the manipulator along the trajectory [12]. However, this approach requires the knowledge of the position of desired targets.

As an objective of this research, functions of unknown environment navigation, object detection, and manipulation

[1] Department of Electrical and Computer Engineering and Computer Science, University of Detroit Mercy, Detroit, United States, (gaosh1, wangbi1, wangpe1, yanba, wangch25, luoch, faiedma)@udmercy.edu

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

are necessary for an autonomous mobile manipulator robot. It allows the operators to input a desired searching area, and the robot searches and detects the object autonomously. Meanwhile, it is desired that the robot platform is operating in a universal environment with flexibility for modification on both hardware and software levels. The flexibility allows the increasing demands on the tasks for the robot, for which a specific algorithm development is challenging [13], [14]. Robot Operating System (ROS) provides an operating system like platform, organizes the control parameters into messages under a structure called topic. ROS made the development and controlling of the robotic system easier and standard for various platforms in both academics and industry [15].

The primary supportive coding languages by ROS is C++ and Python. However, Matlab is widely used for designing and testing prototypes with its advantage of data processing. The Robotics System Toolbox is a new toolbox that released after Matlab R2015A, which allows efficient communication between Matlab and ROS. A Matlab-ROS system based on path planning simulation is introduced by M. Galli et al. [16]. Also, a Pioneer 3-DX (P3-DX) robot controlled with Matlab-ROS system for navigation task is discussed [17], and a project using a similar hardware setup for ROS based mobile robot navigation is introduced very recently [18].

In this paper, an autonomous mobile manipulator robot platform that functioning with mapping, path-planning, obstacle avoidance, image processing, color-based object detecting and manipulation algorithms running in the Matlab-ROS environment is introduced as shown in Fig.1. The system is build based on Pioneer 3-AT (P3-AT) robot and WidowX 6-DOF manipulator with multiple sensors on board. External devices that based on the Arduino system are also available for accessing by MATLAB. The benefit of this Matlab-ROS environment robot platform allows future developers to have an easier way to implement modification and debug prototype algorithms on it.

There are some challenges during this project, specifically due to the relatively less number of research on the P3-AT robot comparing to P3-DX. The P3-DX is the differential drive version of the Pioneer mobile robot. Most of the currently available control packages are designed for P3-DX robot, which made it necessary to modify the those packages. This will be discussed in Section III.

There are two algorithms design and test on this platform. The first one is a local path planner for unknown map environment navigation and map exploration. It combines obstacle avoidance algorithm from dynamic window approach [5] and VFH (Vector Field Histogram) [4] with fuzzy base [19]. This will be discussed in Section IV. Color-based object detection [20], localization, and manipulation algorithm is implemented through the integration of Matlab, ROS and Arduino device, and will be discussed in Section V.

II. System Architecture

Dynamic mobile manipulation robot platform is built based on the Pioneer 3-AT robot, equipped with Trossen Widow X robot arm, Hokuyo LiDAR UST-10LX, Microsoft

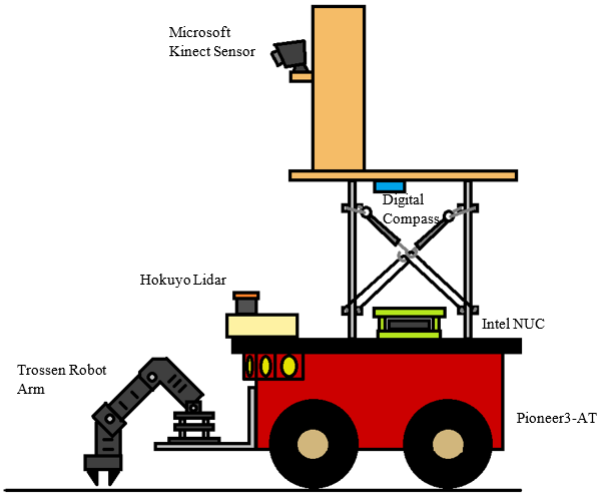

Fig. 2. Hardware architecture

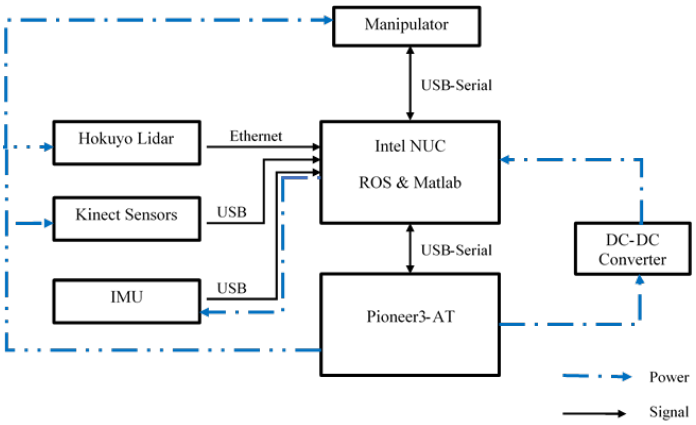

Fig. 3. Hardware architecture showing the signal flowing and powering

XBOX ONE Kinect Sensor and Sparton IMU (Inertial Measurement Unit) as shown in Fig. 2. The platform is controlled by the Intel NUC computer running Linux Ubuntu 14.04 LTS operating system, which is compatible with ROS Indigo. The communication and powering chart is shown in Fig. 3

For the actuators of the system, P3-AT is a highly versatile four-wheel-drive robotic platform, powerful yet easy to use, reliable yet flexible. P3-AT is a popular team performer for outdoor or rough-terrain projects that has powerful motors and four knobby wheels can reach speeds of 0.8 meters per second and carry a payload of up to 12 kg [21].

The WidowX Robot Arm Mark II is an updated version of Interbotix Labs' arm offered for the MX series of DYNAMIXEL Servos. It has streamlined the arm with servos provide full 360-degree freedom of movement, high resolution of 4096 positions, and smooth interpolation. The WidowX Robot Arm has up to a 41cm horizontal reach and 55cm of vertical reach as shown in Fig. 4. At a 10cm reach, it can lift up to 800 gram, and at 30 cm up to 400 gram. The gripper itself has a rated holding strength of up to 500 gram, while the wrist itself can lift up to 500 gram horizontally

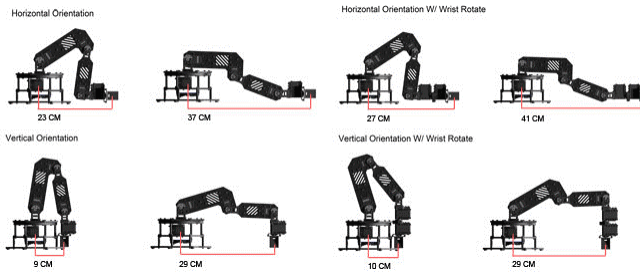

Fig. 4. WidowX Robot Arm working range

(400 gram if using wrist rotate) [22].

For the sensors, the Hokuyo LiDAR has 270 degrees view of range at 10 meters scanning range, which is used for mapping the surrounding environment and obstacle avoidance on this platform. The Kinect sensor is performing as visual input device. Image data from the Kinect sensor is used in the image processing algorithm to find desired objects. IMU is used to increase the accuracy of robot's localization and reduce the error caused by wheel-encoders.

III. ROS Packages Modification

As mentioned, the Pioneer 3-DX support packages needed to be modified for Pioneer 3-AT. P3-DX robot is a dual-wheel mobile robot product developed by the same company. The control system and hardware architecture are similar between the two robots. However, the kinematic is different between two robots, which requires some custom modifications to the existing resources.

On the input side, the odometry data is calculated and processed by the P3-AT robot onboard controller. The raw data from wheel-encoders make no difference in how a pose message is shown in ROS. Also, P3-AT has four wheel-encoder, which gives the robot more accurate odometry data compared to the P3-DX that has only two. On the actuating aspect, the mechanical driving structure for P3-AT robot is actually similar to P3-DX. There are two DC-motors instead of four in P3-AT. The two wheels on the same side are driven by a rack that receives power from single DC-motor. The desired output is processing by the built-in controller for driving motors.

In this case, it is logical to assume that the mapping and navigation packages designed for P3-DX are compatible with P3-AT robot. The input and output data are interchangeable between two. While this assumption is correct in general, the next subsections will discuss some configuration modifications in detail that have been done in this project.

A. LiDAR Configuration Changes

There is an available *SLAM-Gmapping* package on the ROS library designed for P3-DX robot using SICK LiDAR. For this project, changing all configuration to the Hokuyo LiDAR is required. The ROS package *urg_node* is the driver for the Hokuyo LiDAR in the ROS environment. It is necessary to replace all the SICK LiDAR configuration in all launch files to the one for Hokuyo LiDAR so that the sensor works correctly.

B. Package Modification for Pioneer 3-AT

Although the Pioneer 3-AT robot is supported by the P3-DX mapping and navigation packages, some detailed configurations are necessary to be modified.

1) Robot's size for path-planning: There is a configuration file defining the footprint robot in the *p3dx_navigation* package: costmap_commom_params. In case of the P3-AT robot, the robot has a larger scale than the P3-DX robot. The data of robot footprint need to be increased accordingly for better navigation.

2) The maximum running velocity: The maximum velocity for the P3-AT robot is faster than P3-DX. The P3-AT wheels have larger width comparing to P3-DX. It makes the robot more stable to be operated in all-terrain. While this change of max velocity definition is not necessary because it is able and safer to run P3-AT robot following P3-DX's maximum speed limit. But it will eliminate the speed advantage of P3-AT robot.

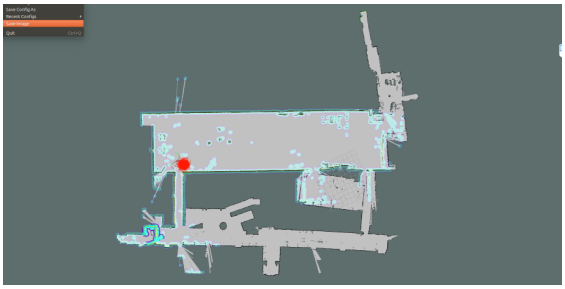

Fig. 5. Map of first floor of Engineering Building at University of Detroit Mercy made by *slam_gmapping* package, with running test for *P3-DX Navigation* package on P3-AT

3) Package Capability Test: Couple tests are done on Pioneer 3-AT after the packages were modified to make sure the compatibility of packages. While the robot is driving itself to the goal, the mapping should keep updating, for both obstacle avoidances and localization by comparing LiDAR data with the originally given map. This process is defined as simultaneous localization and mapping (SLAM). The result is shown in Fig. 5. And the result confirms the success of modification. While the result shows that the odometry data from wheel-encoders is not good enough. The map has some deviation. Utilizing IMU sensor will fix this error, and significantly increase the accuracy.

IV. Local Path Planning Algorithm

A fuzzy based local path planning algorithm [19], based on Dynamic Window Approach [5] and VFH (Vector Field Histogram) [4] is developed in Matlab. A local path planning algorithm is useful in two circumstances. The first one is for a robot to navigate between two global path planned waypoints. And the other is for the robot to navigate in unknown map environment, where global path planning is unavailable. The map exploring task is a good example. In this project, algorithm runs in Matlab, reading sensors data, processing and publishing the velocity command to

/move_base topic. Then the command will be executed by robot's wheel motor and move the robot.

In this approach, by setting the coordination of the target point, the robot should scan the map along the trajectory without hitting static or dynamic obstacles. As a common sense, the shortest path between two points is the line connecting in between. This idea works the same when there is no obstacle between two waypoints. The robot is following the line until obstacle is detected and a detour needs to be performed. By receiving real-time surrounding environment information from the LiDAR, the algorithm could get the obstacle density data on each surrounding sector and compare to the target point heading to find out the optimal direction. By using real-time data, algorithm will update the orientation to drive toward goal with relatively short trajectory as well as avoiding obstacles.

The Pseudo code for the path planning algorithm in Matlab is shown below.

Algorithm 1 Local Path Planning in Unknown Map Environment

Input: Odometry, LiDAR, IMU Data, Goal Position

1: **while** $\text{Dist}_{Goal} > Threshold$ **do**
2: $Sector_{RAW} \leftarrow LiDAR\_Data$
3: $Sector_{FILTED} = (\Sigma Sector_{RAW} \pm 2)/5$
4: $Sector_{Obstacle} = Obstacle_{thres} - Sector_{FILTED}$
5: $Open\_sector = list[Sector_{Obstacle} == Obstacle_{thres}]$
6: $\theta_{global} = atan2(goal - Odometry)[y, x]$
7: $\theta_{goal} = \theta_{global} - \theta_{IMU}$
8: $Sector_{Goal} = (\theta_{goal} - (Number_{sectors} + 1)/2) \times \theta_{sec_wid}$
9: $Sector_{go} = Open\_sector[min(|Open\_sector - Sector_{goal}|)]$
10: $[vel, rotate] = f(Sector_{go})$
11: $ROStopic/cmd\_vel \leftarrow [vel, rotate]$

For the robot to converge to the desired vector facing towards the goal or avoiding obstacles quickly without reducing the linear velocity, a controller is introduced to make the robot turn while maintaining linear velocity v_0. The robot dynamic model is defined when on a 2D plane as:

$$\begin{bmatrix} \dot{x} = v_0 \times cos(\theta) \\ \dot{y} = v_0 \times sin(\theta) \\ \dot{\theta} = \omega \end{bmatrix} \quad (1)$$

Symbols *x, y,* θ are the current status of the robot pose. e_X and e_θ indicate the error on distance and angle from the desired line trajectory. The a, b and c is the property of the desired trajectory line.

$$\begin{bmatrix} e_X = \frac{|ax+by+c|}{\sqrt{a^2+b^2}} \\ e_\theta = \theta_d - \theta \\ \dot{e_X} = v_0 sin(e_\theta) \\ \dot{e_\theta} = \omega \end{bmatrix} \quad (2)$$

$$\omega_d = K_p \times (e_X + e_\theta) + K_d \times (\dot{e_X} + \dot{e_\theta}) \quad (3)$$

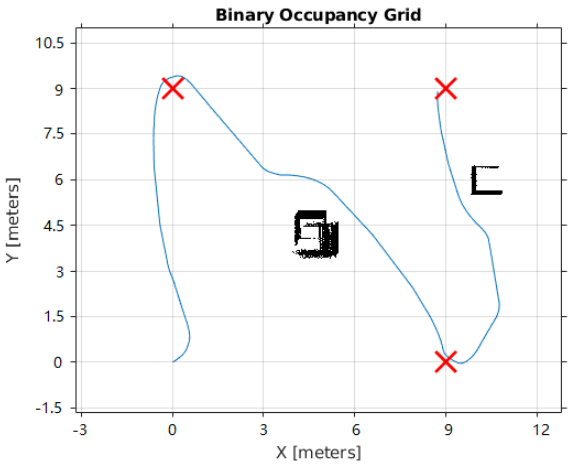

Fig. 6. Robot trajectory, marked waypoints and obstacle figure

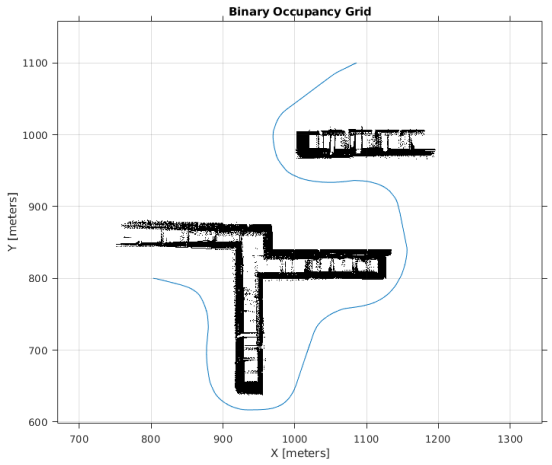

Fig. 7. Robot trajectory, navigated around the obstacle and mapped the environment

The desired velocity is calculated as (v_0, ω_d), in which v_0 is the maximum available linear velocity. In the first experiment, a sequence of waypoints: (9,0), (0,9), and (9,9) in the unit of meter, are sent to the robot as target points with respect to the robot's initial point. There are two obstacles on the playground lying along the path. The result of robot's actual path is shown in Fig. 6. The robot navigates through the desired waypoints, scanning the environment while avoiding obstacles. In the second experiment, the robot successfully navigates around the obstacles, reaches the goal with map scanned during the motion in Fig. 7.

V. Color-based Object Detection and Manipulation Algorithm

The color-based object detection and manipulation algorithm requires integration of ROS, Matlab, and Arduino. The image input device is the Microsoft Kinect sensor, which publishes an RGB image to the ROS topic /Kinect2. Matlab subscribes and processes the image to find out blue items in the sight view, as well as the pose of end-effector in the pixel

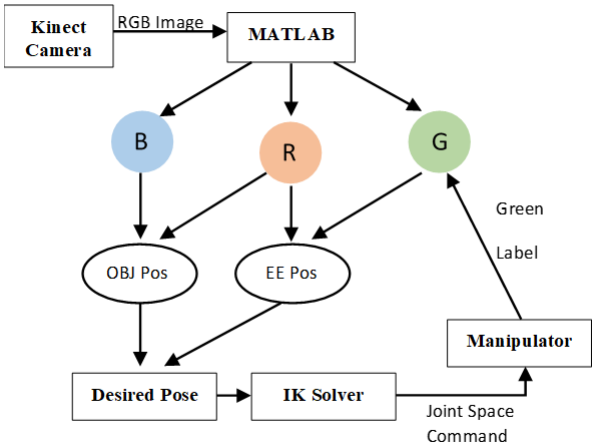

Fig. 8. Color based object detection and pick up flow

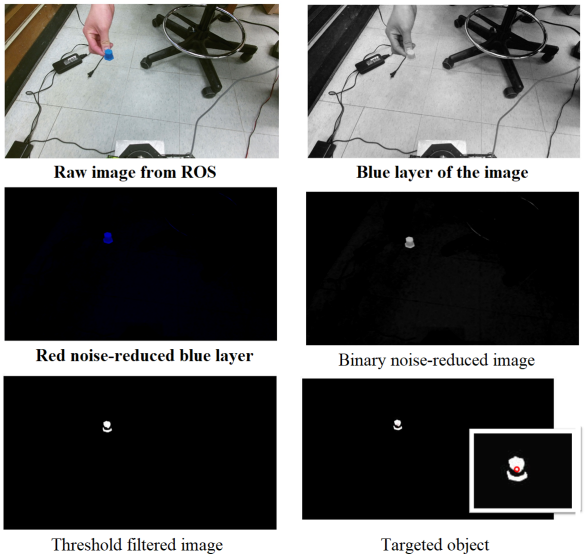

Fig. 9. Image processing process in Matlab

frame. Compare the position difference and solve the inverse kinematics for the robot arm. Return the joint command to the robot manipulator Arduino controller through USB-serial communication. The algorithm is shown in Fig. 8.

The method to pick up an object based on the color is performed by localizing colored object and color-labeled manipulator end-effector simultaneously. The end effector is labeled in greed and the object is in blue. The RGB pixel matrices are processed in which, the red pixel layer is set to be brightness reference to cancel out the noise in the other two layers [23]. By subtracting brightness from blue and green layers, pixels with a higher purity of the layer color is shown. Then the pixels are filtered by a threshold so that the desired pixels outstayed. The geometric center coordinates of those pixels are calculated as the localized position in image space. It indicates the position of object to be picked up and the pose of end effector. This process is shown in Fig. 9.

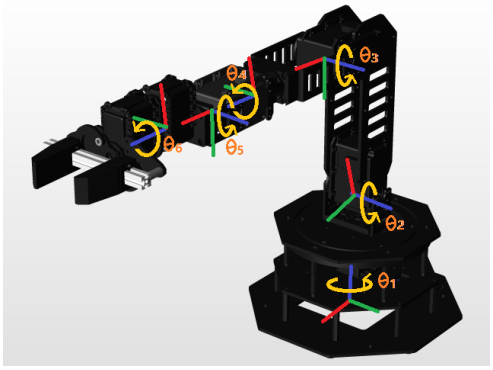

Fig. 10. Trossen WidowX Robot Arm frame assignment

By comparing the relative position between object detected and end effector in image space, a desired moving direction of the manipulator is calculated. The motion is discredited into steps for further localizing. Since the orientation of grasping is designed to be fixed at this project. The joint angle is calculated only based on the Cartesian position of the end effector with respect to the arm base frame by inverse kinematic. The arm base frame is related to the center of the mobile base with linear translation. For controlling the manipulator, the forward kinematics of the WidowX robot arm is solved as below for the end-effector position. The frames and variables $\theta 1$ to $\theta 6$ are assigned based on Denavit–Hartenberg convention as Fig. 10 shown. For simplicity, adding angle such as $\theta 2 + \theta 3$ will be represented as $\theta 23$.

$$x = cos\theta 1 \times (sin(\theta 234) + 146 \times cos(\theta 23) + 151 \times cos\theta 2) \quad (4)$$

$$y = sin\theta 1 \times (sin(\theta 234) + 146 \times cos(\theta 23) + 151 \times cos\theta 2) \quad (5)$$

$$z = 146 \times sin(\theta 23) - 80 \times cos(\theta 234) + 151 \times cos\theta 2 \quad (6)$$

Hence the inverse kinematics of the manipulator is calculated as below. The desired x, y, z Cartesian coordinates and the orientation of the gripper are known as it is fixed as discussed before. The orientation is $\theta 234$ in this specific kinematic.

$$R = \sqrt{x^2 + y^2} \quad (7)$$

$$\theta 1 = atan2(y, x); \theta 5 = 0 \quad (8)$$

$$A = 302 \times (R + 150 \times sin(\theta 234)) \quad (9)$$

$$B = 302 \times (z - 150 \times cos(\theta 234)) \quad (10)$$

$$C = (A/302)^2 + (B/203)^2 + 151^2 - 147^2 \quad (11)$$

$$D = A^2 + B^2 - C^2 \quad (12)$$

$$\theta 2 = atan2(B, A) + atan2(\sqrt{D}, C) \quad (13)$$

$$M = z - 151 \times sin(\theta 2) - 150 \times cos(\theta 234) \quad (14)$$

$$N = R - 151 \times cos(\theta 2) + 150 \times sin(\theta 234) \quad (15)$$

$$\theta 23 = atan2(M, N) \quad (16)$$

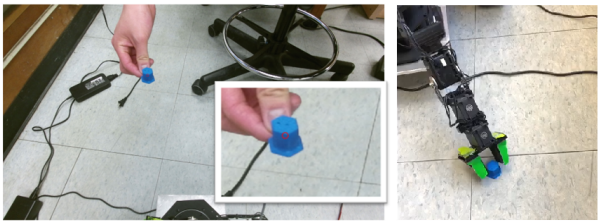

Fig. 11. The blue object is detected and localized. Robot manipulator picked up the blue object autonomously.

$$\theta 3 = \theta 23 - \theta 2 \tag{17}$$

$$\theta 4 = \theta 234 - \theta 23 \tag{18}$$

By this approach, a set of possible IK solutions are generated because of the kinematic redundancy of the 6-DOF robotic arm. Since the robotic arm is mounted in a specific way at the purposed platform, half of the workable range is blocked by the mobile base. This reduces the solutions of inverse kinematics and simplifies the question. After removing solutions with out-of-limit joint angles. A joint space motion command is sent to Trossen arm via Matlab-Arduino interface until the end-effector reaches the object. The result of localization and grasping shows in Fig. 11.

VI. Conclusion

The dynamic mobile manipulation robot platform is successfully built and tested under different tasks for autonomous navigation in the unknown-map environment, and color-based object detection and manipulation. However, there are issues existing during the testing.

As a current limitation, the position for mounting the LiDAR is critical. Since the Hokuyo LiDAR only scans in the 2D plane, obstacles without enough height or have wider lower part won't show up in the sensor. It will further increase the risk of collision. Another issue observed during the testing is that the collision between Pioneer 3-AT mobile base and the robotic arm. The mobile base blocks a part of the arm's working range. The current inverse kinematic solver does not check for collision. Future work could focus on adding boundaries on the robotic arm's working range by having the mobile base as obstacle in arm's joint space, and monitoring the joint position for collision avoidance. A better loco-manipulation that coordinates the mobile base and arm for maintaining manipulability or visibility could be investigated in the future.

References

[1] K. Miyasaka, M. Hatano, "Research on Rescue Robots with Force Sensors on the Fingertips for Rubble Withdrawal Works", The Society of Instrument and Control Engineers SICE Japan 2008.

[2] M. Schnaubelt S. Kohlbrecher, O. Stryk, "Autonomous Assistance for Versatile Grasping with Rescue Robots" in *2019 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, Würzburg, Germany, 2-4 Sept. 2019

[3] R. Dechter, J. Pearl, "Generalized best-first search strategies and the optimality of A*", *Journal of the ACM (JACM)* 32, 505-536. J. ACM. 32. 505-536. 10.1145/3828.3830.

[4] J. Borenstein, Y. Koren, "The vector field histogram – fast obstacle avoidance for mobile robots", *IEEE Journal on Robotics and Automation*, vol. 7, no. 3, pp. 278-288, 1991.

[5] D. Fox, W. Burgard, S. Thrun, "The dynamic window approach to collision avoidance", *IEEE Robotics and Automation Magazine*, vol. 4, no. 1, pp. 23-33, Mar. 1997.

[6] S. X. Yang and C. Luo, "A neural network approach to complete coverage path planning", *IEEE Transactions on Systems, Man, and Cybernetics*, Part B, 34(1) 2004. pp.718-725.

[7] C. Luo, S. X. Yang, "A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments", *IEEE Transactions on Neural Networks*, 19(7), 2008. pp. 1279-1298.

[8] C. Luo, S. X. Yang, M. Krishnan, M. Paulik and Y. Chen, "A Hybrid System for Multi-goal Navigation and Map Building of an Autonomous Vehicle in Unknown Environments", in *Proc. of IEEE International Conference on Robotics and Biomimetics (ROBIO'2013)*, pp. 1288-1233, Dec 2013.

[9] Chaomin Luo, S. X. Yang, X. Li, and M. Q.-H. Meng, "Neural Dynamics Driven Complete Area Coverage Navigation Through Cooperation of Multiple Mobile Robots", *IEEE Transactions on Industrial Electronics*, DOI 10.1109/TIE.2016.2609838. vol. 64, no. 1, pp. 750-760, Jan. 2017.

[10] S. X. Yang, C. Luo, H. Li, J. Ni, and J. Zhang, "Theory and Applications of Bioinspired Neural Intelligence for Robotics and Control", *Computational Intelligence and Neuroscience*, Vol. 2016 (2016), Article ID 5089767, 2 pages, http://dx.doi.org/10.1155/2016/5089767.

[11] K. Nagatani T. Hirayama A. Gofuku and Y. Tanaka, "Motion planning for mobile manipulator with keeping manipulability", in *2002 IEEE/RSJ International Conference on Intelligent Robots and Systems* vol. 2 Lausanne 2002 pp. 1663-1668.

[12] D. Okabe N. Sato Y. Morita "Tele-operation system for rescue robot by inputting target position of end-effector", in *System Integration (SII) 2013 IEEE/SICE International Symposium*, pp. 861-866 2013.

[13] H. Utz, S. Sablatnög, S. Enderle, G. K. Kraetzsehmar, "Miro - middleware for mobile robot applications," *IEEE Transactions on Robotics and Automation*, Special Issue on Object-Oriented Distributed Control Architectures, vol.18, pp. 493-497, 2002.

[14] S. Masoud Sadjadi, Philip K. McKinley, "Transparent autonomization in CORBA", *Computer Networks*, Volume 53, Issue 10, 14 July 2009, Pages 1570-1586, ISSN 1389-1286.

[15] S. Zaman, N. U. Haq, M. I. Gul, A. Habib, "Robotic navigation based on logic-based planning", in *2017 International Conference on Communication Computing and Digital Systems (IEEE-CCODE)*, pp. 396-401, 2017.

[16] M. Galli, R. Barber, S. Garrido, L. Moreno, "Path planning using Matlab-ROS integration applied to mobile robots", in *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pp. 98-103, 2017.

[17] S. Zaman, W. Slany, G. Steinbauer, "ROS-based mapping localization and autonomous navigation using a pioneer 3-DX robot and their relevant issues", *IEEE Saudi International Electronics Communications and Photonics Conference*, pp. 1-5, 2011.

[18] S. Gatesichapakorn, J. Takamatsu, M. Ruchanurucks, "ROS based Autonomous Mobile Robot Navigation using 2D LiDAR and RGB-D Camera", in *2019 First International Symposium on Instrumentation, Control, Artificial Intelligence, and Robotics (ICA-SYMP)*, 2019

[19] Karthika Balan, Melvin P Manuel, Mariam Faied, Mohan Krishnan and Michael Santora, "A Fuzzy Based Accessibility Model for Disaster Environment," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019, pp. 2304-2310.

[20] A. N. Fitriana, K. Mutijarsa, W. Adiprawita, "Color-based segmentation and feature detection for ball and goal post on mobile soccer robot game field", in *IEEE-International Conference on Information Technology Systems and Innovation (ICITSI)*, pp. 1-4, October 2016.

[21] Pioneer 3 Operations Manual, Omron Adept Mobile Robots, Amherst, NH, USA, 2017.

[22] WidowX Robot Arm Kit Datasheet, Trossen Robotics, Downers Grove, IL, USA, 2017.

[23] M. Abdellatif, "Effect of color pre-processing on color-based object detection", in *2008 Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, Aug 2008.