Autonomous Mobile Manipulator

Deep Saran Masanam
Electrical Engineering Department
RIT
Rochester, USA
dm7079@g.rit.edu

Sriparvathi Shaji Bhattathiri
Electrical Engineering Department
RIT
Rochester, USA
ssb6096@g.rit.edu

Jaji Bala Jyothika Pamarthi
Electrical Engineering Department
RIT
Rochester, USA
jp8250@g.rit.edu

Abstract— Mobile manipulators have a prominent role in different industries such as military, space, health care, industries. SLAM and the operation of the arm are two significant parts of it. In this paper, we discuss in detail about the localization and mapping technique that is ORB SLAM2 which uses depth sensors to analyze the environment around the robot and perform pseudo pick and place, using ALB5 arm robot using inverse kinematics.

Index Terms: AmigoBot, AL5B Arm Robot, Mobile manipulator, ORB SLAM 2, Inverse Kinematics.

I. INTRODUCTION

Mobile manipulator is an umbrella term used to refer to all robotic systems that are encompassed of a mobile base with an arm / manipulator attached to it. There are innumerable mobile manipulators employed in various fields. Having autonomous mobile manipulators can be helpful in carrying out work in hazardous environments, for space exploration and even in health care.

There are many ways in which automation of the mobile manipulator is achieved which includes some path planning method. SLAM i.e. Simultaneous Localization and Mapping is a popular method for exploring an unknown terrain by constructing and updating the map of the terrain. It also involves simultaneously keeping track of the location of the robot. It generally estimates the position and reconstructs the environment. Most the algorithms are based on monocular cameras, stereo cameras and RGB-D cameras.

The arm manipulator also must be automated. The path of the arm can be planned according to the task to be performed. Usually, in industries arms perform certain set of repetitive tasks while a mobile manipulator used in domestic environment may perform a variety of simple and complex tasks.

II. PROBLEM STATEMENT

When robot is moving from one point to other to pick and place an object in dynamic environment like space, sea, industries. In these cases, programming the robot to move in fixed path is not possible. To make this operation feasible in dynamic environment, mobile manipulator with SLAM is used. SLAM can be used to Simulate, Localize and Map the surroundings of the robot continuously. With SLAM, the robot can move to location specified by avoiding obstacles. Using path planning algorithm for the pick and place of the object using ALB5 helps to optimizing the task.

III. LITERATURE REVIEW

In dynamic environment, continuous localization, mapping and path planning are major challenges which have innumerable applications in the field of robotics. SLAM is technique which uses sensor devices such as cameras to get the images of the environment and get the details such as the distance from the different objects and position of the object. For the manipulator, path planning is important to avoid collision with other objects in the workspace. These methods are explained in detail in below sections.

A. SLAM

Simultaneous Localization and Mapping (SLAM) is the problem of using a mobile robot to build a map of an unknown environment and at the same time localizing the robot within the map [1]. In most of the SLAM algorithms are based on static world assumption i.e. these require that the objects in the environment are non-moving. RGB-D cameras are widely used as they are of low cost, appropriate size and weight. They can provide depth measurement and visual information with RGB images. SLAM generally uses Iterative Closest Point (ICP) algorithm for point-cloud registration with a two-step mechanism. The first step includes Random Sample Consensus (RANSAC) algorithm, which is used to roughly estimate the 3-D transformation by finding an optimum model and in the second step, ICP is used to refine the transformation with the initial guess from the first step [2].

RGB-D SLAM are classified into two types, Dense and Spare SLAM. Sparse SLAM algorithms are mainly based on visual odometry i.e. using visual features for pose to pose motion [3]. This algorithm is fast and determines the pose based on spare point features, but the imaging quality is poor and there are many repeated/ redundant points in the map as stated in [3]. Dense SLAM algorithm was first introduced in Kinect Fusion which can obtain real time depth measurements and a map simultaneously, but it is limited to small workspaces, consuming high memory [3].

The data associations are of two types in SLAM front end, short term data and long term. Where short term data association refers to pose estimations and long-term data association impacts the loop detection [2]. The main disadvantage is that the data associations are easily failed in dynamic environments where the environments keeps changing i.e. the objects keep moving. Generally, motion removal is used to solve this problem in dynamic environments which can be divided into two parts namely

offline and online approaches as stated in the [2]. In offline approach the processing starts after receiving the complete data due to which they cannot produce the motion removal immediately. In contrast, the online methods segment moving objects simultaneously with the SLAM as moving the moving objects must be removed before data association [2]. Meng [4] used multiple RGB-D cameras to do accurate wide filed view and to improve pose tracking where he did an extrinsic calibration between the different cameras which mainly improve the map build using the dense model in real time efficiently.

ORB-SLAM2, is SLAM technique which can be implemented using monocular, stereo and RGB-D cameras that helps in map reusing, loop closing and re-localization. With monocular camera, SLAM has few drawbacks like scale drift and low performance pure rotations as there is no proper depth information with one camera. With stereo or RGB-D camera, the above-mentioned issues can be solved with help of depth information. ORB-SLAM2 for stereo and RGB-D cameras is works on principle of monocular feature-based ORB-SLAM. The system has three main components according to Ra´ul Mur-Artal and Juan D. Tard´os [5] : 1)the tracking to localize the camera with every frame by finding feature matches to the local map; 2) the local mapping to manage the local map and optimize it and 3) the loop closing to detect large loops and correct the drift by performing a pose-graph optimization. ORB-SLAM2 is a feature-based method so, it preprocesses the input to extract features for key point locations. The input images are discarded after feature extraction and system classifies the key points as close and far based on distance. For RGB-D cameras, extracted features on the RGB image and, as proposed by Strasdat et al. [6], for each feature with coordinates (uL, vL) and the depth value is formed into virtual right coordinate.

$$\mathrm{uR} = \mathrm{uL} - fx\ b/d$$

where fx is horizontal focal length and b is baseline between structured light projector and the infrared camera, which is approximately 8 cm for Kinect and Asus Xtion. The close key points are triangulated from one frame as depth and scale, translation and rotation information are provided. Far points provide rotation information but weak scale and translation information. Monocular key points are defined by two coordinate xm= (uL, vL) on left image and correspond to all ORB that have improper depth value in RGB-D case. [5]

B. Path Planning

Path planning problems are major challenges in motion planning which finds innumerable applications in the field of robotics. [7] Robots being an integral part of our everyday life need to navigate uncertain and complex terrains. Path planning becomes very crucial. [8] Path planning entails finding an optimal path between two points commonly referred to as start point or source and end point or destination in a workspace with obstacles. The word optimal is rather ambiguous since different robots have different abilities and constraints as well as different applications. Yet, in a general sense it can be said that path planning provides a path which optimizes the energy consumption, time taken to converse the path and improves safety. The safety aspect of path planning is of great importance as it ensures that both the robot and the obstacle are safe. The energy efficiency of the robot depends on the ability for sensing, moving and rotating of the different parts of the robot and hence on it physical and mechanical structure also. [7]

There are various algorithms available to tackle the path planning problems. Some of these algorithms are heuristic, some are exact, and some are even approximation algorithms. Minimization of the path length is the goal of most of these algorithms like cell decomposition, road map, potential field and heuristic algorithms. [7]

Cell decomposition algorithm, road map algorithms and potential field method are classical methods just like sub goal methods and sampling space methods. [9]

Cell decomposition algorithm essentially decomposes the workspace and converts the path planning problem into a graph search problem. It finds a feasible path which is collision free and there is very little guaranty of other aspects of optimality. [7] Sub goal method utilizes a listing of possible configurations that avoid all the obstacles while traversing from starting point to the end. Planning schemes based on sampling-based motion planning are probabilistic road-map and rapidly exploring random trees. [9] Visibility graphs and Voronoi diagrams are the examples of road maps. Road maps work by constructing collision free paths and connecting the source and destination through these paths. Visibility graphs are only semi collision free while Voronoi diagrams give collision free paths, but they may be too long. Potential field approach is fast but may get trapped in the local minima. [7] The classical methods have many drawbacks and heuristic methods evolved to overcome these. [9]

Heuristic based algorithms are neural networks, fuzzy logic, nature inspired algorithms and hybrid algorithms. Heuristic methods like ant colony optimization and particle swarm optimization are nature inspired and can be used for path planning, but the most popular approach is using genetic algorithm (another nature inspired algorithm) which is robust and can be used in both the discrete and continuous space. [7]

C. Inverse Kinematics for the arm robot

Manipulator robots, especially those with multiple degrees of freedom are required in many fields. They were initially employed in manufacturing industry and have been rapidly developing since as early as 1950s. Due to the need of high level of intelligence for autonomy and versatility, they have become a focus area for intensive research. [10]

For a robot manipulator which is usually also referred to as a robotic arm, if the joint angles are given then the position of the end effector and its orientation can be determined by using co-ordinate transformations. Using the DH convention, the frame of each joint can be assigned. Then the transformation matrix can be obtained. The Inverse kinematics as the name suggests is the derivation of the

kinematics of a model from the end-effector position and orientation i.e., depending on the goal position the joint angles of the arm robot are decided. To compute the inverse kinematics equations total transformation matrix will be used. In comparison to the forward kinematics, inverse kinematics is rather complex and there is no one analytical solution. Different manipulators will have different methods of solution depending on the unique structure of the robot as well as its constraints. [11] Further there can be more than one solution to reach a position and orientation even for the same robot. [10] So a solution will have to be chosen from many possible solutions. Many at times the solution chosen will depend on the particular initial position or even on the
Inverse kinematics can be solved in various ways. The geometric approach uses the cartesian coordinates, length of the links and trigonometry to decide the angle to which each joint will move. [10] This method is very tedious and often even more complicated as the number of joints and degree of freedom increases. [12]

A common method employed to obtain the inverse kinematics is using the Arm Jacobian. It is a quality of significance when analyzing and controlling an arm robot. Depending upon the structure of the robot, Jacobian can be of varying sizes. For the AL5B robot arm the Jacobian matrix is a 6 by 5 matrix as it has 5 revolute joints. [11]

The analytic method would be to obtain the closed form equations. This will involve solution of non-linear equations with trigonometric functions. A better way is to use a numeric method which involves approximation using multiple iterations. Frames are attached to each joint and the DH table parameters are obtained. The transformation matrix between two successive joints can hence be formed. The transformation from the base to the end effector is thus available and can be written symbolically. We can construct numeric transformation matrix to the goal position. A solution for the equation can be obtained by equating the symbolic transformation matrix and the numeric transformation matrix and iterative approximation can be used to get the value of the angles. The root mean square error of each of the solutions obtained can be calculated using the forward kinematics and the suitable solution can be selected. [12]

To verify the inverse kinematics of the robot, the initial/ home position can be simulated using forward kinematics and then the co-ordinates of the end effector fed to the inverse kinematics calculator. The simulation can be observed to check if the robot has reached the desired location. The experimental results can be used to further verify the validity of the inverse kinematic solution. [13]
The inverse kinematics thus obtained can be used for path planning.

IV. Proposed Method

The AmigoBot forms the base of the mobile manipulator while the AL5B arm robot is the manipulator. Xtion camera is depth camera which is used to get the depth images of the environment around the AmigoBot that helps in localization and mapping of the environment. ORB SLAM2 is implemented with the help of Xtion depth camera which is placed on top of the AmigoBot and ALB5, an arm is also placed on top of the AmigoBot to perform pseudo pick and place as shown in the Figure 1.

Fig 1: AmigoBot interfaced with ALB5 and Xtion

There are two steps in SLAM:

1. Building a map
2. Locating the device

The first step in SLAM is to build a map of the environment based on which the robot can autonomously move in that place unless the environment is constantly changing. For environment which is constantly changing active SLAM with constant monitoring based on a prebuild map is used. For locating the robot in the environment, the algorithm compares the current frame with the previous frame of the camera such that it can assume that the robot is at that same location. To avoid damage to the robot obstacle avoidance behaviour is implemented using the eight sonars present in the AmigoBot such that it can avoid crashing in all directions.

The DH table of the arm robot AL5B which constitutes the manipulator part of the mobile manipulator is shown in Table 1.

Table 1: DH table of the AL5B ARM (Manipulator)

i	alpha	a	d	theta
1	0	0	d1	Theta1
2	90	0	0	Theta2
3	0	a3	0	Theta3
4	0	a4	0	Theta4-90
5	-90	0	d5	Theta5
6	0	0	0	Gripper

For the arm robot, which was part of the setup, the length parameters in the DH table corresponded to the following.

d1=4.5;

a3=13.5;

a4=12.5;

d5=4.5;

Servo Position /Joint name	Center Value	Angle Limits			
		Min		Max	
		Servo Value	Degree	Servo Value	Degree
Base	1450	800	-70	2200	90
Shoulder	1500	800	-80	2150	80
Elbow	1400	900	-60	2000	70
Wrist	1475	800	-80	2100	90
Wrist Twist	1480	800	-80	2100	90
Gripper	1575	2200	Closed	800	Open

It was observed that the minimum and maximum angle limits the joints in AL5B does not equal the theoretical limits. From the limits the limits of the servo motor and the angle as well as setting safe limits for the servos which are less than the limits as mentioned in the table, the degree values are mapped to the servo values.

The formula used is as follows -

Qreach_servo_goal=qreach_goal_degrees * 10 * 180

Then it is simulated in MATLAB and integrated with ROS such that when we reach the goal point then it performs pseudo pick and place.

V. Results

For building the map the robot is made to move around the entire room such that it maps the environment using Xtion. For the ORB SLAM2 to work and map correctly the robot must move slowly as it tries to re-localize to track properly.

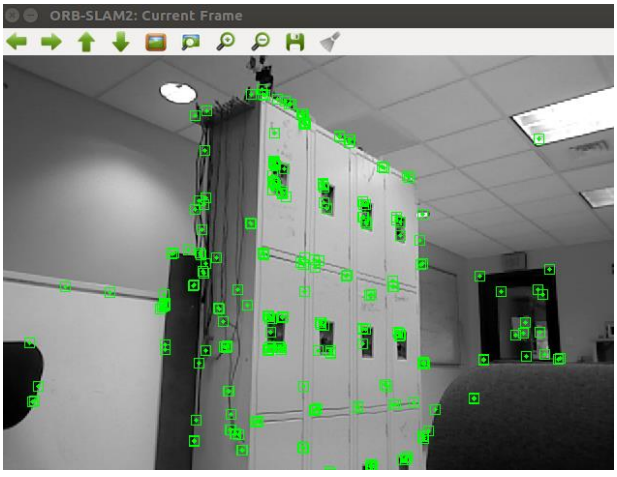

Fig 2: Xtion tacking the environment

The figure 2 shows the tracking of the environment of environment i.e. green boxes shows the tracked keypoints.

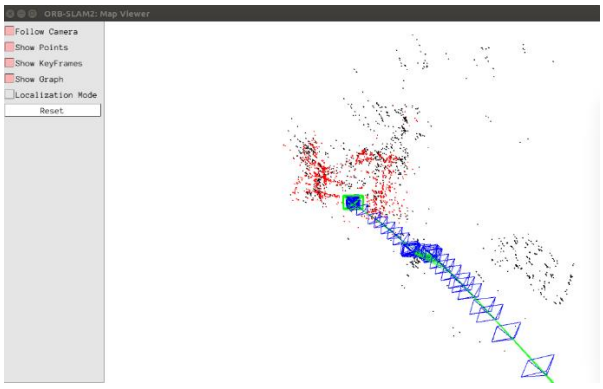

Fig 3: Map viewer

The Figure 3 shows the tracked keypoints onto a map i.e. blue boxes shows the keypoints in map frame, red dots shows the local map points. By propagating the coordinate system across the map from the current location to previous position with which a new map is updated. Octomap is used to visualize the 3D occupancy grid. It is displays both occupied cells as well as free cells as shown in the Figure 4.

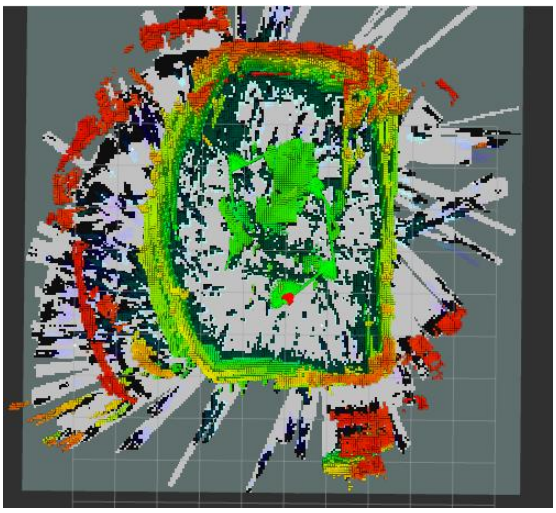

Fig 4: Octomap of the room

It is used to see the depth map of the environment which cannot be viewed by the normal cost map, local or global map. Where cost, local and global maps are 2D maps shown below.

Fig 5: Local Map of the room

The arm robot has been simulated in MATLAB using the Robotics Toolbox available in MATLAB. It is a third party toolbox which provides numerous functionalities including

serialLink() used for the simulation. The equivalent of home position is obtained in the following figure.

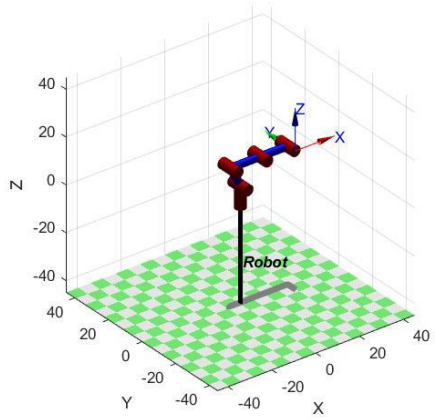

Figure 6: Robot in Home Position

The co-ordinates of the end effector at the home position is fed to the ikcon() function available in the MATLAB Robotics Toolbox. It can be observed that the robots end effector reaches the same position and orientation but using different joint angles.

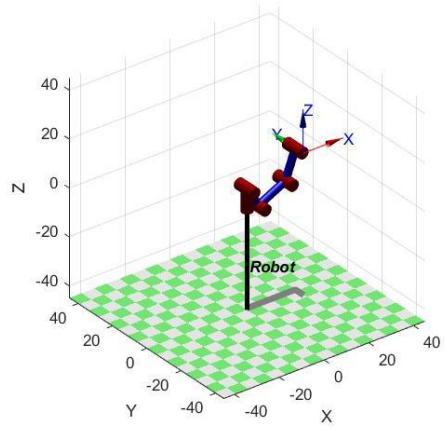

Figure 6: Robot reaching a point in space

The ikcon() function uses a method similar to the one explained in the literature survey where in a numerical iterative method is used and error is minimizing the error between the joint angle solution and the forward kinematics. It essentially works like an optimization problem and requires the optimization toolbox also. Fmincon() function from the optimization toolbox is used to perform the following –

sumsqr((inv(T)*robot.fkine(q) - eye(4)) * omega)

where omega is an arbitrary matrix

omega = diag([1 1 1 3/reach])

The angle values obtained are in radians and have to be converted to corresponding servo values. Using this the arm is moved and the pick and place is facilitated. While the inverse kinematics is efficient, the angles used to reach the same position and orientation may not be the same in all the different tries. Further since optimization is used in the solution of the inverse kinematics, the initial position and orientation of the frame plays an important role.

The height of the object that is picked up remains the same in this experimental setup and so does the distance of the object from the mobile manipulator. The object detection is done using an Xtion and it cannot detect objects closer than 8 cm. Hence the robot stops 8 cm away requiring the arm manipulator to extend to 8 cm in the x-axis.

VI. Conclusion

The autonomous manipulator does ORB SLAM2 , pick and place .But it take a lot of time to reach the goal as it keeps on checking the environment i.e. keeps sensing. The inverse kinematics used for pick and place reaches the desired position after reaching the goal point set from the map of the place.

References

[1] Yuxiang Sun a, Ming Liu a, Max Q.-H. Meng b, “Motion removal for reliable RGB-D SLAM in dynamic environments”, Robotics and Autonomous Systems 108 (2018) 115–128

[2] Yuxiang Sun a, Ming Liu b, Max Q.-H. Menga, “Improving RGB-D SLAM in dynamic environments: A motion removal approach” Robotics and Autonomous Systems 89 (2017) 110–122

[3] Liang Wang, and Zhiqiu Wu, “RGB-D SLAM with Manhattan Frame Estimation Using Orientation Relevance” Sensors 2019, 19, 1050; doi:10.3390/s19051050

[4] Xinrui Meng, Wei Gao and Zhanyi Hu “Dense RGB-D SLAM with Multiple Cameras” Sensors 2018, 18, 2118; doi:10.3390/s18072118

[5] Ra´ul Mur-Artal and Juan D. Tard´os, “ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo,and RGB-D Cameras” IEEE TRANSACTIONS ON ROBOTICS, VOL. 33, NO. 5, OCTOBER 2017

[6] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, “Double window optimization for constant time visual SLAM,” in Proc. IEEE Int. Conf. Comput. Vis., 2011, pp. 2352–2359.

[7] Davoodi, Mansoor, et al. "Clear and smooth path planning." Applied Soft Computing 32 (2015): 568-579.

[8] Le, A., Prabakaran, V., Sivanantham, V., & Mohan, R. (2018). “Modified a-star algorithm for efficient coverage path planning in tetris inspired self-reconfigurable robot with integrated laser sensor.” Sensors, 18(8), 2585.

[9] Mac, T. T., Copot, C., Tran, D. T., & De Keyser, R. (2016). “Heuristic approaches in robot path planning: A survey”. Robotics and Autonomous Systems, 86, 13-28.

[10] Kim, H. W., Chen, H., & Lee, J. M. (2014, July). Path planning of 5-DOF manipulator. In 2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (pp. 877-881). IEEE.

[11] Qassem, M. A., Abuhadrous, I., & Elaydi, H. (2010, March). Modeling and Simulation of 5 DOF educational robot arm. In 2010 2nd International Conference on

Advanced Computer Control (Vol. 5, pp. 569-574). IEEE.
[12] Huang, G. S., Tung, C. K., Lin, H. C., & Hsiao, S. H. (2011, May). Inverse kinematics analysis trajectory planning for a robot arm. In 2011 8th Asian Control Conference (ASCC) (pp. 965-970). IEEE.
[13] Zhao, L., Huang, S., & Dissanayake, G. (2019). "Linear SLAM: Linearising the SLAM problems using submap joining". Automatica, 100, 231-246.