Semantic Mapping and Motion Planning with Turtlebot Roomba

Rizwan Aslam Butt *, Syed M Usman Ali*
*Electronic Engineering Department, NED University of Engineering. & Technology, Karachi, Pakistan
rizwan.aslam@neduet.edu.pk, uashah68@neduet.edu.pk

Abstract. In this paper, we have successfully demonstrated the semantic mapping and motion planning experiments on Turtlebot Robot using Microsoft Kinect in ROS environment. Moreover, we have also performed the comparative studies on various sampling based motion planning algorithms with Turtlebot in Open Motion Planning Library. Our comparative analysis revealed that Expansive Space Trees (EST) surmounted all other approaches with respect to memory occupation and processing time. We have also tried to summarize the related concepts of autonomous robotics which we hope would be helpful for beginners.

1. Introduction
Motion planning is the core of autonomous robotics. The idea is fascinating that a robot manipulator is asked to reach an assigned position (goal) from its current position (start) in the workspace with the help of a motion planning algorithm, while avoiding hitting the obstacles in the way. Robot operating system (ROS) [1] is a quite popular open source development environment for robotic application development. Drivers and supporting packages are available for many robots in ROS like PR2 [2], Turtlebot [3] etc. Due to integration of Point Cloud Library (PCL) [4] and OpenCV [5] in ROS it is easier to develop visualization and navigation applications for ROS supported Robots. In this paper we have described our motion planning experiments on Turtlebot in our University Lab. We used Microsoft Kinect for vision and particle filter based Simultaneous and Localization And Mapping (SLAM) technique for generating grid based environmental map. Later on, we utilized this generated map for solving motion planning queries in ROS environment which were successfully solved by our robot. During the investigations, it has been observed that the performance of Robot was not consistent in solving the motion planning queries and the reason for such behavior was expected due to heavy computational load for working with complete map of the environment. Thus, based on our observed results, we decided to test the performance of our robot with sampling based motion planning approaches using Open Motion Planning Library (OMPL) by Kavraki lab [6]. This library provides the provision for testing as well as visualizing a motion planning algorithm. The state space and the robot model can be defined in collada file format. A motion planning query can be assigned using Scripting, Coding or through Graphical User Interface. We developed our environmental and robot models in collada file formats using Google Sketchup.

2. Related work
Autonomous and Assistive robotics is currently an active area of research. Especially ROS has gained wide popularity as a common open source platform for the development of perception and navigation
activities using ROS supported robots like PR2, TOM ROSSIE [7] and Turtlebot. For example, Intelligent autonomous systems Group (IAS) has demonstrated autonomous robotic activities in kitchen environment in [8] [9] using PR2 and TUM Rosie. Similarly the Kivriaki labs group is dedicatedly working on motion planning algorithms. Since it is hard to directly implement a Motion planning algorithms so it is often suitable to first test the algorithm in a simulation environment like OpenRave [10] and OMPL. As pointed out by Mark Moll at el. in [11] OMPL is a recent addition to such simulators through which it is easier to simulate and demonstrate motion planning scenarios without the need of going to complex and low level programming. The OMPL has now gained quite wide popularity in analyzing complex motion planning tasks. For example Christos Fragkopoulos et al has presented an analysis of various sampling based motion planning algorithms on 7 Degrees of Freedom (DOF) robotic arm mounted on the rehabilitation system FRIEND in a SE(3) configuration space. One more advantage of OMPL is that it can also be integrated with ROS via OMPL ROS interface. Therefore, we have chosen OMPL for analyzing various sampling based motion planning algorithms on Turtlebot robot.

3. Motion Planning with Turtlebot

Motion planning is the task for a robot of finding a collision free path from a starting point to a goal point and then actually following that path. This first requires some sort of semantic map of the environment like Grid based maps or Roadmaps and then locating its current position within that map. Both these tasks are pre-requisite for each other so also termed as chicken and egg problem. The solution to this problem is termed as SLAM. There were different SLAM solutions available to resolve such issue like [12][13] DP-SLAM, FastSlam , GridSlam , GMapping etc [14] . Each of these approaches is based on some of probabilistic estimator like Kalman filter, extended Kalman filter and particle filter. We used Gmapping package available in ROS which is a particle filter based SLAM technique.

3.1. Simultaneous Localization and Mapping (SLAM)

It is a famous technique used by autonomous vehicles and robots to develop not only a map of an anonymous environment but also to be able to keep track of their current position. If only we rely on the wheel encoder information for measurement of the current robot position in the map then the measured distance and direction travelled might be slightly inaccurate and thus the mapping process will be erroneous. Thus, It is required to improvise the erroneous measurements for the correct mapping otherwise earlier accumulate erroneous measurements will distort the whole map. Which means our measurements of robot’s relative poses will also be erroneous. The solution for such issues is to have some static landmarks in the environment to improve our position measurements. SLAM approach provides us exactly the same mechanism. In SLAM process, we extract landmarks from the environment and improve our robot’s position estimates with the help of some estimator. The typical example of such landmarks is walls in a room. For discovering landmarks in the environment we need a vision system like Laser Scanner, Ultrasonic sensors or Microsoft Kinect etc.

3.2. Semantic Mapping

is defined as the task of generating models of a robot’s environment from sensor’s data in which we have the information about the free space and obstacles present in this environment. In the perspective of indoor systems, three map concepts prevail namely; topological, geometric, and grids. Topological representations try to represent environments with structures like graphs, where nodes correspond to "something distinct" and edges represent an adjacency relationship between nodes. In cases of geometric models various geometric primitives are used for representing the environment. From the observation data the mapping is then responsible for estimating the best fit parameters of the primitives. Different representations, up till now, have been used with great success. One most successful primitive has been line segments used by many researchers as in [15] and [16] to represent parts of the environment. In Grid based approaches, the environment is divided in fine resolution grids.
with each marked as free or occupied. Occupancy grids are one of such popular approach which uses grid structures. In this approach the grid cells may be considered as pixels and the value of each pixel corresponds to the likelihood that its corresponding portion of workspace or configuration space is occupied [17]. Occupancy maps have been built with different approaches like stereo vision [19], Laser Scanners, and ultrasonic range-finders [18, 19, and 20].

3.3. Motion Planning Algorithms
There are different kinds of motion planning algorithms. Bug1 and Bug2 algorithms [21] are the oldest and simplest. These approaches treat the robot as a point in a plane with a contact sensor (or a zero range sensor) for detecting the obstacles. These techniques do not build any map of the environment and rather move along the obstacles like a bug to reach their destination. Their improved form is Tangential Bug [22] which used range sensor and thus has a global knowledge of the obstacles around and thus considered more robust. Another real-time method of obstacle avoidance and path planning was presented by Ossama Khatib from Stanford University in [23]. It directs a robot from a start point to the end point as if it was a charged particle moving in a vector field gradient. Positive charge is assigned to the obstacles forming a repulsive force which directs the robot away from obstacles. The combination of attractive and repulsive forces hopefully guides the robot from the start location to the goal location while at the same time avoiding collision with obstacles. Another class of motion planning algorithms is mapping based planning. These methods require map of the environment in the configuration space and then considering the free spaces in the map as the nodes employ the graph theory concept for finding the shortest path from the robot’s current pose to the destination pose. These methods often have two phases. In first phase, also termed as Global Planning, a shortest path algorithm like Dijkstra or A* is used for finding collision free poses to the goal point from the start point. In second phase a Local Planner algorithm is used to compute the control signals (velocity, acceleration etc.) suitable to drive the robot on the obtained collision free poses. As these map based methods require a lot of memory and computations therefore, it is more convenient to only consider some samples of free space instead of considering the whole leading to the class of algorithms termed as Sampling based Motion Planning Algorithms. These approaches can be broadly classified in three types; PRM (Probabilistic Roadmaps), EST (Expansive Space Trees) and RRT (Rapidly Exploring Random Trees).

4. Experimental Setup
Our experimental setup consisted of a Turtlebot Roomba with an iRobot Roomba base. For vision we had Microsoft Kinect mounted on it and an Acer Aspire1 notebook, with 1.66 GHZ atom processor and 2GB RAM, connected to it and placed within its racks as shown in Fig. 1. For driving the robot we used available ROS driver for Turtlebot. For teleportation we used the ROS keyboard package and kinect package to remotely control and view our robot.
4.1. Occupancy Grid Map of Radio Engineering Lab at our University

We used the Laser based SLAM implementation in ROS named as Gmapping. It is a particle filter based SLAM implementation. We configured it with 300 particles. Since Laser Scanner is quite costly and not available at NED University, we used Microsoft Kinect instead and used the ROS package Pointcloud_To_Laserscan [24] to convert the depth information of Kinect to equivalent laser scan data. The idea was to read point cloud data of Kinect at a certain height (10cm in our case) and read it at a certain spacing/linear angle. The intensity of the read pixels gave us distance information like a Laser beam in case of a Laser Scanner. We passed this data to PointCloud_Throttle nodelet which republished it to match the Laser Scanner publish frequency. Using this simple technique we were able to use our previous ROS packages and reproduce the real time semantic mapping of an indoor room environment as shown in Fig. 5. The developed map was being viewed with the help of RVIZ [25] at the ROS master node, another laptop in our case. We saved this map using ROS map_saver [26] package to hard disk and is shown in Fig. 2.

![Fig. 2. Occupancy Grid Map of our University Lab](image)

4.2. Motion Planning Experiments on Turtlebot

We used the obtained map to solve a motion query problem. We assigned a goal pose to our robot, used the ROS amcl [27] package to compute the current position in the map, navfn [27] package to obtain a global path in the map, which used Dijkstra’s algorithm for finding shortest collision free path and used Trajectory rollout algorithm as a local planner available in nav_core [27] package. The performance of our robot was not consistent as expected and it failed sometimes. The reason for it was mainly heavy computational load on the notebook for considering the whole workspace.

4.3. Analysis with Sampling Based Approaches

As discussed that Sampling based motion planning approaches are faster and require lesser memory thus become more attractive for the researchers. To obtain best results, we have applied several approaches to our investigations in order to decide the best approach for our Turtlebot robot. We have performed the comparative study of EST, RRT, RRTConnect, LazyRRT, PRM, ERT, KPICE, LBKPICE, and SBL sampling based motion planning algorithms [30] using OMPL in an indoor closed environment. We developed required collada file format models of robot and environment in Google Sketchup. We selected a SE(3) environmental model from OMPL library resources and modified it. Fig. 3 shows our experimental setup in the OMPL GUI. We used all the above algorithms one by one to solve this motion query with OMPL GUI by restricting bounds to allow the movement and rotation in x and y plane only, thus converting the problem to SE(2) instead of SE(3). The solved motion planning query with KPICE showing the explored sample points is shown in Fig. 4.
To elaborate the comparative study further we used the benchmarking utility of OMPL on the same experimental setup to compare all algorithm performance. We restricted the maximum memory usage for a planner to 2000MB. Our investigated results are shown in Fig. 5 (a) and (b). It can be seen clearly from these results that the PRM algorithm with uniform sampling approach consumed the minimal time to solve the motion planning query for our scenario but consumed a bit more memory but EST on the other hand had a slightly more simplification time but consumed the minimal memory. Therefore, EST proved to be the best motion planning algorithm for this scenario.
4.4. Analysis with thin corridor

To analyze the complicated environment with our proposed system for solving motion planning queries, we introduced a thin corridor between the start and goal positions of our robot as shown in Fig. 6. The obtained results are shown in Fig. 7 (a) and (b) showing that the EST algorithm showed consistent and reproducible results with simple and complicated scenarios during several experiments.
5. Conclusion

The proposed developed system for implementing the EST based Motion Planning algorithm has successfully demonstrated the less memory consumption and processing time as compared to its other counterparts. The outcome from several experiments for simple to complicated scenarios during investigations produces the repeatable and reproducible results. Based on all these advantageous aspects of EST algorithm we conclude that it is most suitable sampling based motion planning algorithm for Turtlebot Roomba Robot that can also be suitable for any other planar robotic systems. Moreover, it is also possible to practically implement this algorithm on Turtlebot using OMPL_ROS interface.

References

[1] www.ros.org
[2] www.willowgarage.com/pages/pr2/overview
[3] http://turtlebot.com/
[4] http://pointclouds.org/
[5] http://opencv.org/
[6] http://ompl.kavrakilab.org/
[7] http://ias.cs.tum.edu/robots/tum-rosie
[8] Michael Beetz, Ingo Kresse, Ulrich Klang, Alexis Maldonado and others, “Robotic Roommates Making Pancakes “, 2011, IEEE 11th International Conference on Humanoid Robots.
[9] Michael Beetz, Lucian, Nico Blodow, Cosmin Goron and others, “Autonomous Semantic Mapping for Robots Performing Everyday Manipulation Tasks in Kitchen Environments” , RSJ/IEEE International Conference on Intelligent Robots and Systems, 25-30 Sep 2011, San Francisco, USA.
[10] www.openrave.org
[11] Mark Moll, Janice Bordeaux, Ioan A. Sucan, Lydia E. Kavraki, “Teaching Motion Planning Concepts to Undergraduate Students “, IEEE Workshop on Advanced Robotics and Its Social Impacts (ARSO), OCT 2-4, 2011.
[12] Gamini Dissanayake, Shoudong Huang, Zhan Wang and Ravinda Ranasinghe, “A Review of Recent Developments in Simultaneous Localization and Mapping “, 6th International Conference on Industrial and Information Systems (ICIIS), Aug. 16-19, Sri Lanka.
[13] Durrant-whyte, Hugh, “Simultaneous localization and mapping: part I “, Robotics &
Björn T. Jensen, Kai O. Arras, Nicola Tomatis and Roland Siegwart, “Multisensor on the fly localization: Precision and reliability for applications. Robotics and Autonomous Systems”, vol 34, issue 2-3, page no 131–143, 2001.

J. Crowley, “World modeling and position estimation for a mobile robot using ultrasound ranging “, IEEE International Conference on Robotics and Automation, 1989.

A. Elfes, “Sonar-based real-world mapping and navigation. IEEE Journal of Robotics and Automation “, vol 3, issue 3, page no. 249–265, June 1987.

R.C. Gonzales (Eds.), Mongi A. Abidi, “ Data Fusion in Robotics and Machine Intelligence “, Chapter 3, Academic Press, New York, 1992.

D.R. Murray, J.J. Little, “Using real-time stereo vision for mobile robot navigation,” Autonomous Robots, vol-8, issue 2, page no.161–17, 2000.

G. Oriolo, M. Vendittelli, G. Ulivi, “Real-time map building and navigation for autonomous robots in unknown environments,” IEEE Transactions on Systems, Man, and Cybernetics, June 1998.

V. Lumelsky and A. Stepanov, “Path planning strategies for point mobile automaton moving amidst unknown obstacles of arbitrary shape “, Algorithmica, vol-2, page no 403–430, 1987.

Kamon, E. Rivlin and E. Rimon, ”Tangent bug: A range-sensor based navigation algorithm”, International Journal of Robotics Research, vol-17, issue 9, page no. 934–953, 1998.

Oussama Khatib, “Real Time Obstacle Avoidance for Manipulators and Mobile Robots” International Journal of Robotics Research,vol-5, 1986.

http://www.ros.org/wiki/pointcloud_to_laserscan

www.ros.org/wiki/rviz

ros.org/wiki/map_server

www.ros.org

Ioan A. Sucan and Lydia E. Kavraki, “On the Implementation of Single Query Sampling Based Motion Planners “, IEEE International Conference on Robotics and Automation 3 to 7 May 2010