Comparison of ROS Local Planners with Differential Drive Heavy Robotic System

Isira Naotunna
dept. of Mechanical Engineering
Chiang Mai University
Chiang Mai, Thailand
naotunna_isira@cmu.ac.th

Theeraphong Wongratanaphisan
dept. of Mechanical Engineering
(Center of Mechatronics System and Innovation)
Chiang Mai University
Chiang Mai, Thailand
wong@dome.eng.cmu.ac.th

***Abstract*—This work aims to analyze the performance of ROS local planners with a differential drive heavy robotic system. Intel Realsense D435i depth camera and T265 tracking camera are used as the primary sensor source for navigation with real-time appearance-based mapping and localization technique. This work has studied the performance of DWA, TEB, and EBand local planners under two experiments. Obtained results are used to analyze the local planners based on how well it follows the global planners, their obstacle avoidance capability, time consumption, goal-reaching accuracy, and the quality of motion generation. Obtained results indicate that DWA and TEB local planners are more suitable for the large differential drive robot setup.**

Keywords—Dynamic-Window, Elastic Band, Timed Elastic Band, Robot Operating System

I. INTRODUCTION

Within the past few decades, the interest to develop robotic systems that are dedicated to autonomous navigation has been rapidly growing. The fast-moving advancements in communication, sensors, and computing technologies have provided significant progress in the field of autonomous navigation in robotics. The concept of autonomous navigation covers many areas, such as map development, localization, obstacle avoidance, and path planning. The navigation algorithms use sensory information to determine a suitable trajectory and move accurately within its environment without collision. However, the development of a custom navigation algorithm for a robot from scratch is quite a challenge. Therefore, the Robot Operating System (ROS) has become a popular open-source software platform among researchers around the world for robot development. Navigation of mobile robots is well documented in ROS with the developed software libraries to address various tasks in robot navigation. ROS official packages are adequate in everyday robotic tasks, and ROS provides an Application Programming Interface (API) to build custom packages or to communicate with external systems and equipment.

ROS navigation stack is a popular ROS library which is used to perform navigation tasks in mobile robot development. In the ROS navigation stack, the path planning strategy is divided into two sections; global planner and local planner. Global planning provides the optimal path based on the provided map, and local planner recalculates the path to avoid dynamic obstacles. Generally, the ROS navigation stack consists of inbuilt local and global planner libraries which can be customized according to the robot configuration. Dynamic-Window Approach (DWA) local planner, Timed Elastic Band (TEB) local planner, and classical Elastic Band (EBand) local planner are the most commonly used local planner libraries with ROS navigation stack. The purpose of this study is to analyze the performance of these three local planners using the differential drive heavy robotic system. In this study, the navigation stack is developed based on the data provided by Intel Realsense D435i depth camera and T265 tracking camera setup.

The outline of this paper is divided into five main sections. Sections II consist of the studies that have been previously conducted to analyze the performance of the ROS navigation stack and a brief explanation about the local planner algorithms. Then, the experimental setup is explained in section III before discussing the experiment and obtained results in section IV. The conclusion and future works in section V.

II. BACKGROUND STUDY

A. Related Work

Even though ROS is a popular platform to develop autonomous navigation tasks in mobile robotics, only a handful number of studies have been done to investigate the path planning techniques of ROS. These studies address the performances of various areas of path planning techniques such as local planners, global planners, goal-reaching accuracy, control velocity generation, and repeatability.

A systematic analysis of global and local planners for different scenarios in simulation and real-world applications is presented in [1]. The study is done using a Turtlebot platform, and the results show that the TEB local planner had the best performance. A similar ROS local planner study is presented in [2]. This experiment is done in a simulated environment using a small scale differential drive robotic system. The impact of the local planner on the accuracy of reaching the goal is presented in [3]. In this work, TEB, DWA, and EBand local planners are analyzed using a TurtleBot platform in both simulated and real-world environments, and the test results indicate that the DWA performed the best repeatability, EBand had the highest tolerance, and TEB local planner had the best speed of action. As per the above literature, a majority of the studies on path planners are done using small scale robots or simulated robots. A slightly different study when compared to those above, can be found in [4]. This study analyzes the performance of the TEB local planner using an actual Ackremen type vehicle.

According to the existing studies on the path planning strategies of mobile robotic systems, it is not easy to find a performance analysis of a ROS-based path planning techniques with a 3D vision system. Also, most of the existing studies are done using the Turtlebot research platform or in a simulated environment. Therefore, this study will be useful to identify the behaviour of ROS-based local planners with a 3D vision-based localization system and a heavy robotic system.

978-1-7281-6530-1/20/$31.00/ ©2020 IEEE

B. Local Planner Algorithms

The algorithms of the local planner techniques which are discussed in this study are briefly explained below.

a) DWA Local Planner: A dynamic window approach of 1997 is presented by [5]. According to their work, DWA performs a sample-based optimization to produce a pair of linear and angular velocity values which represents a robot's optimal circular trajectory. The trajectories are simulated according to the specified horizon lengths based on the motion model of the robot. Because of the constant control action along the defined horizon, the DWA planner is unable to produce reverse motion. The principle of operation of DWA local planner consists of five main steps. The first step is the velocity sampling in the velocity space. Secondly, for a short time, a forward simulation is performed to every velocity pair to identify the possible robot's motion. As a third step, the trajectory scores are calculated from each forward simulation using the cost function. Finally, the best trajectory is selected, and the associated velocity value is sent to the base controller. This algorithm is implemented in the dwa_local_planner package in ROS [6].

b) EBand Local Planner: EBand local planner is developed based on the elastic band theory [7]. This algorithm imitates the elastic behaviour of a rubber band, and the algorithm generates a collision-free path based on two essential components; contraction force and repulsion force. The contraction force is responsible for generating the shortest path between the start point and goal point while the repulsion force repels the path from the obstacles. This algorithm is developed in ROS as the eband_local_planner package [8].

c) TEB Local Planner: TEB local planner is implemented using the Timed Elastic Band Algorithm, which is a modified algorithm of the elastic band approach [9]. This method also has similar characteristics but, instead of using contraction and repulsion forces, this algorithm optimizes every moment of trajectory deformation and minimizes the target cost function. This algorithm requires information about kinematics, dynamics, geometric shape, acceleration, and velocity limits. TEB local planner is available in ROS as the Teb_local_planner package [10].

III. Method

A. Robot Description

The experiment is performed using a differential drive mobile robotic system which is developed to carry a Baxter manipulator, as shown in Fig. 1. Since this experiment focuses only on the local planner study, the manipulator system is not used. The approximate weight of the entire system is about 100 kg, and it is 0.75 m long, 0.7 m wide and 1.5 m tall. Nvidia TX1 development board is used as the main computer in the system, and an Arduino MEGA 2560 which communicates with NVidia TX1 via ROS Serial package is used to control the mobile robot that is driven by two brushed DC motors. AMS's AS5147P magnetic encoders are used to get the velocity feedback from the motors. Two serially connected 12V lead-acid batteries power the system. Also, the setup consists of a MeanWell TS100 inverter to generate the 230V AC voltage required to operate the Baxter manipulator system.

Fig. 1. Robot setup

The Intel Realsense D435i depth camera and the T265 tracking camera are the primary sensor sources for autonomous navigation. Both cameras are connected to the Nvidia TX1 via a USB hub, and the communication is established via Realsense 2.3.5 SDK. Since all tasks related to autonomous navigation is done in the ROS environment, the Realsense ROS wrapper library is used to collaborate Realsense cameras with ROS.

B. Experiment Description

The experiment is performed in a selected area on the first floor of the main engineering building at Chiang Mai University. A 2D map of the selected area is shown in Fig. 2. The selected floor area is approximately 145 $m^{2,}$ and it consists of a room, a doorway, a corridor, and a vacant space. The performance of the local planners is investigated by navigating the robot between four defined waypoints which are namely O1, O2, O3, and O4. The 2D coordinates (in meters) of each point with respect to the map frame are (0.0, 0.0), (5.0, 2.0), (13.2, 2.5), and (22.5, 7.0) respectively. The robot starts to move from point O1, and then it reaches points O2, O3, and O4, respectively. Finally, the robot returns to its initial position O1 from the point O4. During the testing of each local planner, the robot performs 20 trials on this route. These 20 trials are also divided into two sections. The first ten trials run without any additional obstacles, and the next ten trials run with obstacles. During each trial, the data is recorded into a rosbag file.

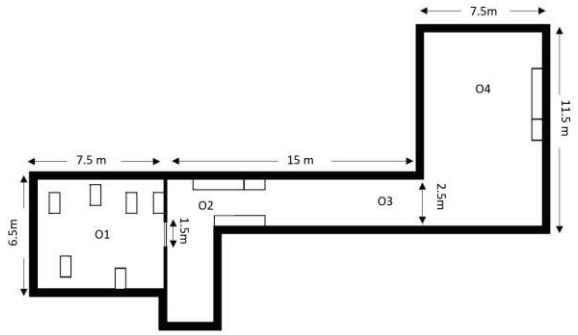

Fig. 2. 2D map of the experimental area

C. ROS Navigation Stack Setup of the Robot

In order to perform the navigation tasks, the ROS navigation stack needs to be appropriately set up according to the robot's requirement. In the developed system, the primary sensor sources of navigation tasks are the Realsense D435i depth camera and the T265 tracking camera. This section briefly explains the robot transformation (TF) tree setup, simultaneous mapping and localization (SLAM) method, and the move_base node setup.

a) Robot TF Tree: Generally, a REP 105 coordinate frame setup is used when setting up the robot's TF tree [11]. In the traditional REP 105 setup, the odometry frame is always directly linked to the base frame of the robot. In that scenario, the odometry data is provided from the wheel encoder or an Internal Measurement Unit (IMU). However, in this system, the robot's base frame is connected to the odometry frame of the T265 camera, as shown in Fig. 3. It helps to provide accurate odometry readings from T265 tracking camera rather than getting the odometry readings from the wheel encoders.

b) Map Generation: Map generation is done using the Realtime Appearance Based Mapping (RTAB-Map) package [12]. RTAB-Map is an RGB-D, Stereo, and Lidar Graph-Based SLAM approach based on an incremental appearance-based loop closure detector. RGB-D data from the D345i camera is used to construct the map using RTAB-Map. The constructed 2D occupancy grid map of the experimental area is shown in Fig. 4.

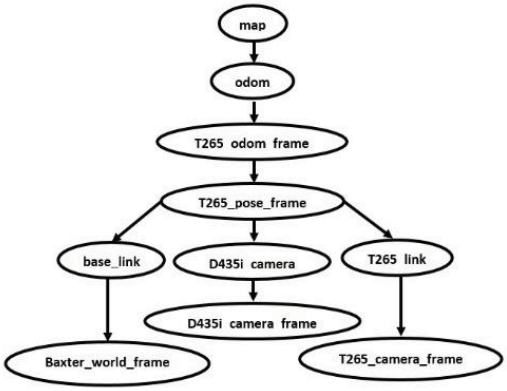

Fig. 3. Robot TF Tree

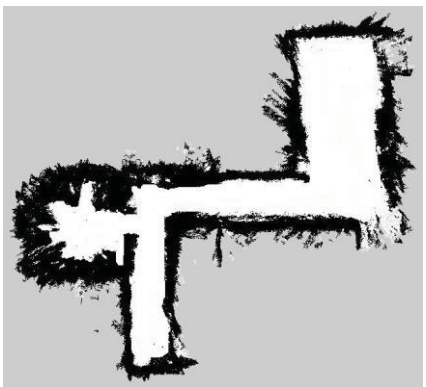

Fig. 4. Occupancy grid map generated by RTAB-Map

c) Navigation Stack Configuration: The navigation stack has to be appropriately configured to perform autonomous navigation. Localization and path planning are the two main sections of autonomous navigation. According to the navigation stack setup shown in Fig. 5, the robot localization is also performed using the RTAB-Map. RTAB-Map provides robot localization based on the database generated during the mapping process, and the T265 tracking camera provides the odometry data. The D435i camera is the primary observation source of the ROS navigation stack. The point-cloud data, and the fake laser scan data that is generated from the depth images are used to update local and global costmaps. Based on the costmap data, local and global paths are calculated for collision-free motion. In path planning, the Dijkestra algorithm is used as a global planner that is available in the navfn ROS package [13].

d) Navigation Stack Parameters: All three local planners are tested with the same global and local parameters. Since the robot setup is a heavy system, the acceleration and velocity limits are set to low values to ensure better safety. The essential local planner parameters used in this experiment are shown in Table I. With this robot setup, slightly higher goal tolerances have to be set to avoid unnecessary oscillations from the robot. In DWA local planner, the sim time parameter provides the time interval for trajectory optimization. The required velocity samples for trajectory generation can be set using vx_samples and v_theta parameters. The path_distance_bias and goal_distance_bias parameters define how much local planner stay close to the global path and the local goal.

The proportional and damping gain of the Eband local planner parameters need to be tuned to obtain better control of the robot. Otherwise, the robot oscillates along the path instead of reaching the goal location. When setting the TEB local planner parameters, the most important thing is to select the proper footprint model. The footprint model of the robot is defined as a circular footprint, even though the actual footprint of the robot is rectangular. Since the wheels of the robot are located along the centre axis of the robot's base frame, selecting a circular footprint model helps to maintain better control of the robot.

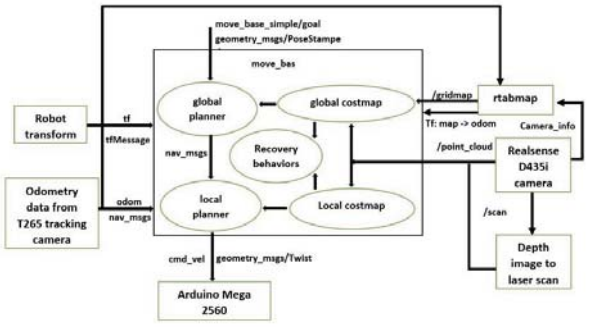

Fig. 5. General setup of the robot's navigation stack

TABLE I. LOCAL PLANNER TUNING PARAMETERS

Parameter Category	Parameters
Velocity and acceleration parameters	Maximum linear velocity:0.4ms^{-1} Maximum angular velocity:0.4 rads^{-1} Linear acceleration limit:1.5 ms^{-2} Angular Velocity limit:1.5 rads^{-2}
Goal tolerances	xy goal tolerance: 0.22 m yaw goal tolerance: 0.1 rad
DWA planner parameters	Sim time: 4 vx_samples:20 v_theta samples:40 path distance bias: 32 goal distance bias: 24
EBand planner parameters	Proportional gain: 4.0 Damping gain: 3.5 Control rate: 10 Hz
TEB planner parameters	Footprint radius: 0.35m Minimum obstacle distance: 0.6m Maximum backward velocity:0.1ms^{-1}

IV. EXPERIMENT AND RESULTS

As explained earlier, this experiment is categorized into two sections. The first experiment is done without any random obstacles in the experimental area, and the second experiment is done by adding obstacles at different locations on the map. All the local planners are tested with ten trials in each experiment, and the results are analyzed offline using the recorded data.

With this data, the local planners are analyzed based on the following methods. First, the performance of the local planners is analyzed based on the Root Mean Square Deviation (RMSD) between global and local planners. It can be used to identify how accurately the local planner follows the global planner. If the coordinates of the global planner and the local planner are given by (X_g, Y_g) and (X_l, Y_l) respectively, the RMSD value can be calculated for n number of trials as shown in (1). In this experiment, the target reaching accuracy is measured using only the data taken at the point O1. Once the robot completes the task by travelling on the path O1→O2→O3→O4→O1, the robot's final position and the orientation is measured based on the odometry data provided by the Intel Realsense T265 camera. Additionally, the generated velocity commands and the time taken to complete the task are also studied to presents a better comparison.

$$\mathrm{RMSD} = \sqrt{\frac{1}{n}\sum_{1}^{n}(X_g - X_l)^2 + \frac{1}{n}\sum_{1}^{n}(Y_g - Y_l)^2} \qquad (1)$$

A. Performance of the Local Planners without Obstacles

Fig. 6 shows a sample set of generated local planners during the experiment. The Figure includes the local planner generated during the path O1→O2→O3→O4 and path O4→O1 separately. According to Fig. 6, it can be seen that all the local planners offset to a certain degree from the exact goal points. One of the main reasons for that is the slightly higher goal tolerances which are used with this system to avoid unnecessary oscillations. Other than that, the localization errors also affect the accuracy of reaching the goal. However, errors that occur when reaching the goal are not more than 0.4 m in any of the trials performed during the experiment. It can be considered as an acceptable range for this robot setup.

EBand local planner generates the shortest path compared to the other two. Even though the path is short, sometimes the robot has to travel through the paths which are very close to the walls. Therefore, a larger inflation radius is more suitable for EBand approach to increase the safety of the navigation. Table II shows the overall performance of each local planner during obstacle-less navigation.

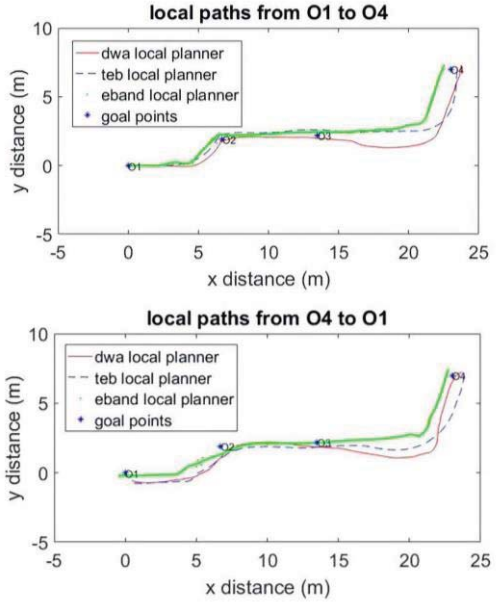

Fig. 6. Generated local planners without obstacles

Based on the average RMSD values obtained from the experimental data, the global planner following accuracy of DWA and TEB local planners are high compare to the EBand local planner. By considering the average time to complete the path, it can be observed that DWA local planner takes less time to complete the compared to the TEB and EBand local planners. Even though EBand local planner generates the shortest path, the generated maximum velocity from the Eband local planner is 0.25 ms^{-1} which is less than the maximum set velocity of 0.3 ms^{-1}. The velocity can be increased by increasing the proportional and derivative gains in the EBand local planner parameters. However, for higher gains, the robot oscillates along the robot trajectory. Therefore, the existing Eband local planner parameters are described in Table I are used to generating the optimum motion under the defined speed limits.

TABLE II. OVERALL PERFORMANCE OF LOCAL PLANNERS WITHOUT OBSTACLES

Performance Measurement	Local planner		
	DWA	***EBAND***	***TEB***
Average RMSD value	0.47	0.74	0.41
The average time is taken to complete the task	252 s	285 s	257 s
Minimum distance from final goal point O1	0.19m	0.12m	0.25m
Maximum distance from final goal point O1	0.41m	0.35m	0.48m
Minimum angular error from desired orientation at goal point O1	0.01 rad	0.01rad	0.02rad
Maximum angular error from desired orientation at goal point O1	1.3rad	1.1rad	1.5rad
Number of successfully completed trials out of 10 trials	10	10	10

According to the position and orientation errors measured at the final goal location O1, it can be seen that Eband has a comparatively higher accuracy of goal-reaching, and TEB local planner has the maximum error deviation. The position and orientation errors can also be improved by reducing the goal tolerances and reducing the footprint size. However, as explained before, reducing the goal tolerance to a lower value creates unwanted oscillations when reaching the goal points. Also, to increase safety, the footprint size of the navigation stack is set to 0.8 m X 0.8 m which is higher than the actual robot's footprint size, which is 0.75 m X 0.7 m. That is also has a slight effect on the accuracy of reaching the goal. However, when compared with the size of the robot, the position errors and orientation errors of the existing setup is acceptable.

B. Performance of the Local Planners with Obstacles

Fig. 7 shows the linear velocity commands generated from each local planner during the experiment. The DWA and TEB local planners have better control of the robot compared to the EBand local planner. That is clearly understood by observing the velocity plot shown in Fig. 7. Since the proportional and the derivative gains of the EBand controller has to be manually tuned, parameter tuning of EBand local planner is slightly tricky when compared to that of the TEB and DWA local planners.

The second experiment is performed by placing two obstacles in random locations on the map. An example set of local paths generated with obstacles is shown in Fig. 8. During the trials, the same measurements are taken for each local planner. Out of the ten trials performed by each local planner, the TEB local planner was able to complete all the trials successfully while DWA local planner completed eight trials, and EBand local planner completed seven trials. From this experiment, it is observed that at some stages, DWA and EBand local planners got stuck in recovery behaviours while avoiding the obstacles.

When looking at the generated path, it can be observed that the TEB local planner has travelled a longer distance when compared to the EBand and the DWA local planners. Generally, the TEB local planner always maintains a minimum distance from the obstacles, and the path is planned accordingly. A minimum distance can be set during parameter tuning, and for this robot setup, it is 0.6 m. Also, TEB local planner can generate backward motions when it cannot generate the forward paths. Therefore, the TEB local planner has the best capability to avoid obstacles without getting stuck in the recovery state. DWA local planner also performed well except on a few occasions where the robot got stuck in recovery behaviour. The main problem of using the DWA local planner with differential drive robots is its inability to

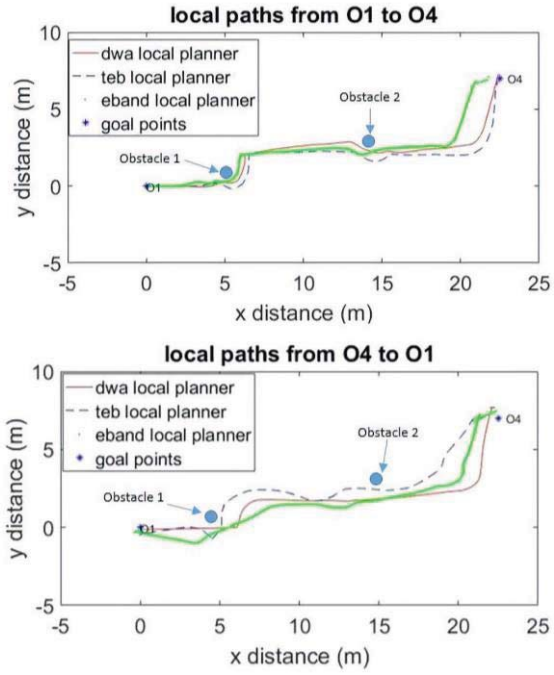

Fig. 8. Generated local planners with obstacles

generate backward motions. Generally, the DWA local planner avoids the obstacles by making a quick turn when it detects the obstacles. However, especially in narrow areas, this quick turn would not be enough to create the path to avoid the obstacle. EBand local planner follows a concept similar to TEB local planner in obstacle avoiding. However, the reactive time of EBand local planner is comparatively low. Thus, it is understood that the EBand local planner also has limitations in avoiding obstacles. When observing the paths O4→O1, all the local planners have avoided the obstacles before the robot reaches the obstacle. That is because the global costmap gets updated after detecting each obstacle when the robot travels from O1 to O4. Therefore, the global plan from O4 to O1 does not go through the previously detected obstacle locations, and this helps the robot to avoid the obstacles before it reaches it.

Table III shows the overall performance of the local planners with the obstacles. The RMSD values are comparatively higher than that of the previous experiment due to the continuous change of path during obstacle avoidance. When considering the overall time taken to complete the task, the DWA local planner is the quickest. However, DWA local planner can only complete 7 trials out of 10. Therefore, TEB local planner can be considered as the best, even if it takes more time to complete the task.

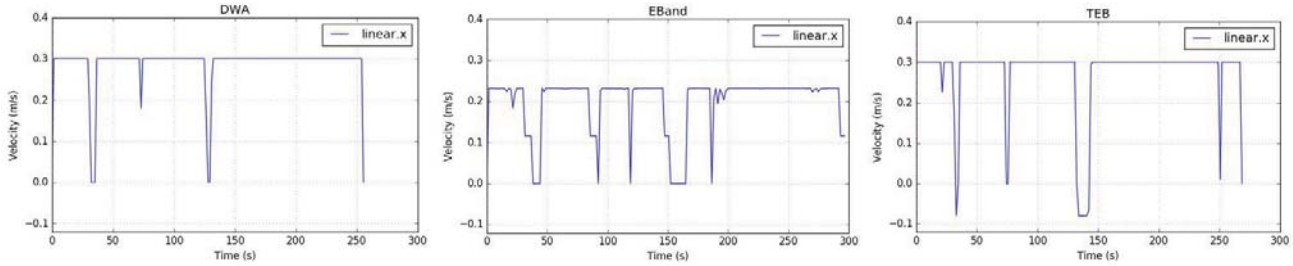

Fig. 7. Linear velocity generation of local planners

TABLE III. OVERALL PERFORMANCE OF LOCAL PLANNERS WITH OBSTACLES

Performance Measurement	Local planner		
	DWA	*EBAND*	*TEB*
Average RMSD value	0.72	0.98	0.93
The average time is taken to complete the task	285 s	302 s	322 s
Minimum distance from final goal point O1	0.18m	0.15m	0.28m
Maximum distance from final goal point O1	0.52m	0.38m	0.54m
Minimum angular error from desired orientation at goal point O1	0.01 rad	0.01rad	-0.01rad
Maximum angular error from desired orientation at goal point O1	1.5rad	1.3rad	1.2rad
Number of successfully completed trials out of 10 trials	8	7	10

V. CONCLUSION AND FUTURE WORK

The main objective of this study is to analyze the performance of the ROS local planners with a differential drive heavy robot. After conducting the tests and analyzing the results, it can be concluded that both TEB and DWA local planners are an effective path planning strategy for a differential drive heavy robotic system. However, when comparing the TEB and DWA local planners, it is hard to indicate the specific comparison winner. Each of these planners should be considered under different criteria. When considering the smoothness of the motion, repeatability, and time consumption, the DWA local planner is significant. The goal-reaching accuracy of the EBand local planner is comparatively higher than the others. However, when it comes to the obstacle avoidance capability and the higher reactivity, the TEB local planner can be distinguished as the best option. Therefore, TEB local planner can be considered as a better option for a large differential drive robot.

This experiment is conducted with the same navigation parameters for each local planner. It might affect some of the results of local planners alongside with the localization errors. In this experiment, the navigation is completely done based on the data from the Realsense D435i camera and the T265 tracking camera. The results would be more accurate if the localization technique is improved by adding a LIDAR sensor.

For future work, local planners can be studied in a dynamic environment with a large robot setup. Also, the experiment can be improved by adding an external motion capturing setup to measure the goal-reaching tolerances. Furthermore, the performances of local planners with different global planning technologies can also be studied to make a better comparison.

ACKNOWLEDGEMENT

This research is supported by the Department of Mechanical Engineering at Chiang Mai University in Thailand.

REFERENCES

[1] M. Pittner, M. Hiller, F. Particke, L. Patino-Studencki and J. Thielecke, "Systematic Analysis of Global and Local Planners for Optimal Trajectory Planning," ISR 2018; 50th International Symposium on Robotics, Munich, Germany, 2018, pp. 1-4.

[2] M.S.Kim, R.Delgado, and B.W.Choi, "Comparative Study of ROS on Embedded System for a Mobile Robot," Journal of Automation, Mobile Robotics & Intelligent Systems, vol.12,2018.

[3] B. Cybulski, A. Wegierska and G. Granosik, "Accuracy comparison of navigation local planners on ROS-based mobile robot," 2019 12th International Workshop on Robot Motion and Control (RoMoCo), Poznań, Poland, 2019, pp. 104-111, doi: 10.1109/RoMoCo.2019.8787346.

[4] P. M. Plaza, A.Hussein, D.Martin, and A. D. Escalera, "Global and Local Path Planning Study in a ROS-Based Research Platform for Autonomous Vehicles," Journal of Advanced Transportation, 2018.

[5] D. Fox, W. Burgard and S. Thrun, "The dynamic window approach to collision avoidance," in IEEE Robotics & Automation Magazine, vol. 4, no. 1, pp. 23-33, March 1997. doi: 10.1109/100.580977

[6] " Wiki.ros.org.(2020) .dwa_local_planner - ROS Wiki", Wiki.ros.org, 2020. [Online]. Available: http://wiki.ros.org/dwa_local_planner. [Accessed: 9- Nov- 2020].

[7] S. Quinlan and O. Khatib," Elastic bands: connecting path planning and control," [1993] Proceedings IEEE International Conference on Robotics and Automation, Atlanta, GA, USA, 1993, pp. 802-807 vol.2. doi: 10.1109/ROBOT.1993.291936.

[8] "Wiki.ros.org. (2020). eband_local_planner - ROS Wiki". [online] Available at: http://wiki.ros.org/eband local planner [Access: 9- Nov- 2020].

[9] C. Roesmann, W. Feiten, T. Woesch, F. Hoffmann and T. Bertram," Trajectory modification considering dynamic constraints of autonomous robots," ROBOTIK 2012; 7th German Conference on Robotics, Munich, Germany, 2012, pp. 1-6.

[10] "Wiki.ros.org. (2020). teb_local_planner" - ROS Wiki. [online] Available at: http://wiki.ros.org/teb local planner [Access: 9-Nov- 2020].

[11] " Wiki.ros.org. (2020). REP 105 -- Coordinate Frames for Mobile Platforms (ROS.org)", Ros.org, 2020. [Online]. Available: https://www.ros.org/reps/rep-0105.html. [Accessed: 10- Nov- 2020].

[12] " Wiki.ros.org. (2020). rtabmap_ros - ROS Wiki", Wiki.ros.org, 2020. [Online]. Available: http://wiki.ros.org/rtabmap_ros. [Accessed: 10- Nov- 2020].

[13] " Wiki.ros.org. (2020). navfn - ROS Wiki", Wiki.ros.org, 2020. [Online]. Available: http://wiki.ros.org/navfn. [Accessed: 10- Nov- 2020.