Comparison of ROS Local Planners for a Holonomic Robot in Gazebo Simulator

Artem Apurin[1](✉), Bulat Abbyasov[1], Edgar A. Martínez-García[2], and Evgeni Magid[1,3]

[1] Intelligent Robotics Department, Institute of Information Technology and Intelligent Systems, Kazan Federal University, 35, Kremlyovskaya street, Kazan 420111, Russia
aaa@it.kfu.ru

[2] Institute of Engineering and Technology, Department of Industrial Engineering and Manufacturing, Autonomous University of Ciudad Juarez, Manuel Díaz H. No. 518-B Zona Pronaf Condominio, Chihuahua, 32315 Cd Juárez, Mexico

[3] Tikhonov Moscow Institute of Electronics and Mathematics, HSE University, 34, Tallinn street, Moscow 123458, Russia

Abstract. A safe robot navigation in a dynamic environment is an essential part of an autonomous exploration path planning. A path planning part of a navigation involves global and local planners. While a global planner finds an optimal path with a prior knowledge of an environment and static obstacles, a local planner recalculates the path to avoid dynamic obstacles. The main goal of a local planning is adjusting an initial plan produced by a global planner in an online fashion. It is a crucial step to ensure a robot operation in dynamic environments because in real world scenarios an environment usually contains people and thus, a dynamic obstacles avoidance must respond quickly and recalculate an actual route. Holonomic robotic platforms are robotic vehicles that use omni-wheels to move in any direction, at any angle, without an additional rotation. These robotic platforms are ideal for working zones with a limited space access. This paper provides a comparison of ROS local planners that support omni-wheel mobile robots: Trajectory Rollout, DWA, EBand, and TEB. The algorithms were compared using a path length, a travelling time and a number of obstacle collisions. Gazebo simulator was used for modeling virtual scenes with dynamic obstacles.

Keywords: Mobile Robot · Mecanum Wheel · Local Planner · ROS · Gazebo

1 Introduction

Nowadays, mobile robotics provides new opportunities for developing novel robotic systems. A wide range of wheels of various sizes, different design types and materials used allow to integrate mobile robotic platforms into many areas of a human life. Common mobile robot applications include industrial automation [1], transportation [2], medical care [3], emergency rescue operations [4], and other areas. Mobile robots are featured by different motion systems.

There are two different types of mobile robots drive systems: a holonomic and a non-holonomic. For a wheeled robot, a non-holonomic drive system is a robot configuration limited by a number of wheels or their orientation. Holonomic drive systems have more than two degrees of freedom, which provide more freedom and flexibility of motion. The main benefit of a holonomic drive system is an ability to travel in any desired direction at any specified orientation without additional rotations with regard to Z-axis (yaw) of a series of intermediate motions (e.g., a typical car parking procedure). To perform such locomotion a robotic platform uses a special design of wheels called mecanum or omnidirectional [5]. Omnidirectional wheels increase a robot mobility and are used in tasks where a high maneuverability is required. Omnidirectional robotic platforms are ideal for working zones with a limited space access and cluttered environments, e.g., for scheduling pick-up and delivery tasks in hospitals [6].

Performing safe robot navigation is a general issue faced by a robot operating in a real environment [7]. Real world environments usually contain people and other dynamic obstacles. A real-time path planning is an essential part of an autonomous exploration. An obstacle avoidance capability used by a path planning approach must detect obstacles quickly and replan an actual route [8]. A path planning part of a robot navigation involves global and local planners [9]. While a global planner finds an optimal path with a prior knowledge of an environment and static obstacles, a local planner recalculates the path to avoid dynamic obstacles. The main goal of a local planning approach is adjusting a plan produced by a global planner in an online fashion.

This paper presents a comparison of ROS local planners supporting a holonomic drive system: Trajectory Rollout [10], DWA [11], EBand [12] and TEB [13]. These local planner algorithms were selected because they are most popular for ROS environment, easily pluggable and support a holonomic motion. The main contribution of the paper is a benchmark to discover the most suitable ROS local planner for a holonomic system used within a dynamic environment. Virtual experiments were conducted in Gazebo simulator [14] using static and dynamic obstacles.

2 System Setup

2.1 Mecanum Wheel Robot

A virtual model of a modular multifunctional robotic omni-wheeled mobile platform ArtBul [15] was used for experiments. Mecanum wheel models were created in Blender software [16]. To reduce complex collision calculations and increase a real time factor (RTF), low-poly models were used for a roller collision part. A 3D model of the mecanum wheel is shown in Fig. 1. Each roller has its own joint and can be freely rotated along the Z-axis of its frame.

A ROS plugin was developed to control the robot in Gazebo simulation [17] by publishing messages with linear velocities along the X and Y axes and an angular velocity along the Z-axis to a robot command topic. To detect collisions *gazebo_ros_bumper* plugin [18] was used. The mobile platform with a laser range finder (LRF) and an enabled bumper plugin is depicted in Fig. 2.

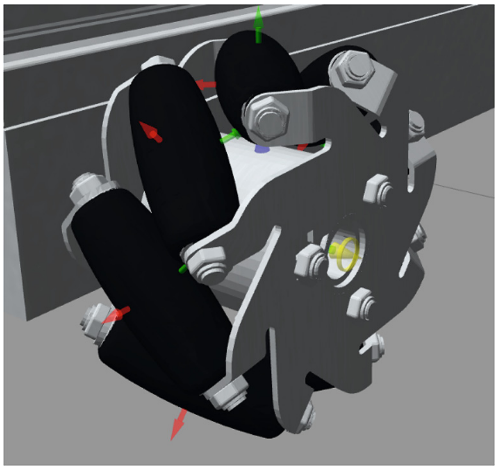

Fig. 1. 3D model of a mecanum wheel in Gazebo: a red, green and blue arrows' set denotes a coordinate frame of each roller.

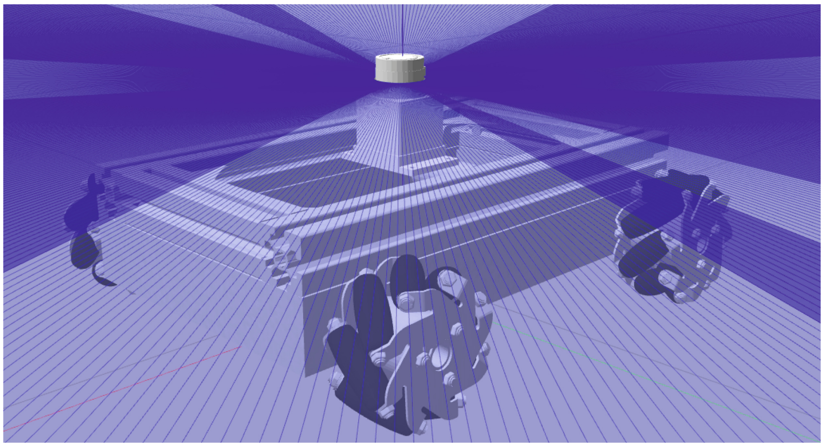

Fig. 2. ArtBul mobile robot in Gazebo: blue rays visualize LRF beams.

2.2 Virtual Environments

Simulation provides a significant support in early stage testing. A 3D modeling can be used to produce a necessary 3D digital representation of real objects with a varying difficulty. Modern modeling tools are often used for designing virtual environments [19]. Testing local planners requires a special navigation map called an occupancy grid map (OGM). The OGM is a 2D binary map that consists of cells. The OGM encodes occupancy data where white pixels represent free cells, black pixels are occupied cells and gray pixels are not yet explored. In our test cases, OGMs should not contain any information about obstacles because a goal of a local planner is a real-time path planning processing.

Two different virtual worlds in Gazebo were created to benchmark ROS local planner algorithms. The first world had 20 × 6 × 3 m dimensions and contained static obstacles (Fig. 3, top). An OGM of the first virtual environment is shown in Fig. 3, bottom. The second world of 10 × 10 × 3 m dimensions contained a single dynamic obstacle – a cube with a 1 m length side (Fig. 4). We created two motion patterns that the cube uses while moving: along X-axis (Fig. 4, left top) and Y-axis (Fig. 4, left bottom). An OGM of the second virtual world is shown in Fig. 4, right.

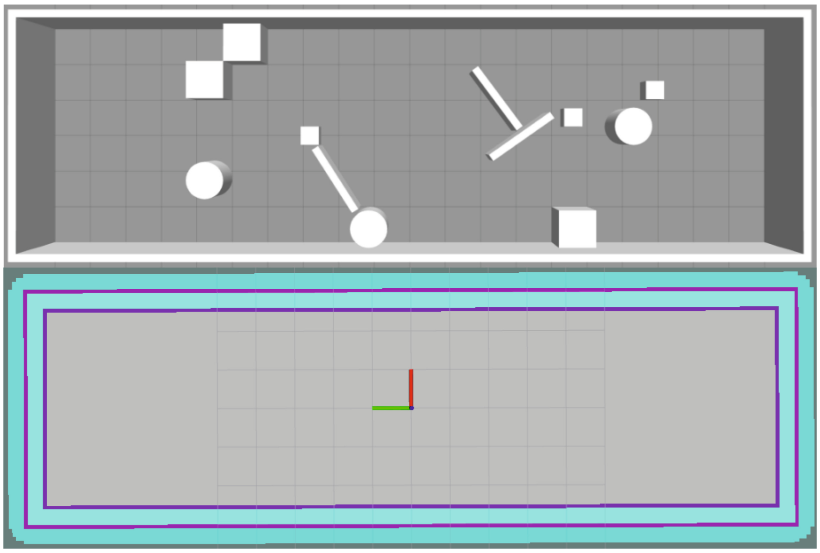
Fig. 3. A 3D virtual environment filled with static obstacles: cubes of varying sizes, cuboids and cylinders (top). The corresponding 2D OGM with static obstacles excluded (bottom).

3 ROS Local Planners

A motion control plays an important role in an autonomous navigation. ROS local planners use sensory information to perceive a current robot state and generates feasible trajectories that the robot is allowed to follow. Avoiding any dynamic or static obstacles that may (or may not) be included in a given global map is a responsibility of a ROS local planner. ROS local planners use sensory data from various sensors such as LRF sensors or ultrasound sensors, and various devices to plan an optimal trajectory [20].

All local planners use the same values for coinciding parameters and the same global and local costmap configurations. For all local planners limitations were set as follows: a linear acceleration was limited to 2.5 m/s^2, an angular acceleration to 3.2 rad/s^2, a linear speed to 0.5 m/s, and an angular speed to 1 rad/s. A controller tolerance in yaw/rotation (*yaw_goal_tolerance*) was set as 0.05 rad, a controller tolerance in the X and Y distance (*xy_goal_tolerance*) as 0.1 m.

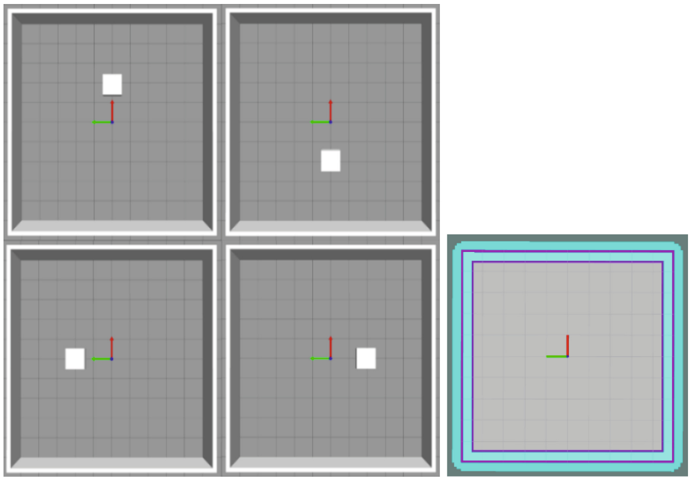

Fig. 4. (Left) 3D virtual environment with a dynamic obstacle: a cube moves along X-axis (top) and Y-axis (bottom). (Right) The corresponding 2D OGM with static obstacles excluded.

Trajectory Rollout. ROS package *base_local_planner* provides implementations of the Dynamic Window and Trajectory Rollout approaches to a local control. *Base_local_planner* is a basic ROS local planner that provides an application programming interface for other local planners. In order to use Algorithm Trajectory Rollout, the parameter *dwa* should be set to *false*. For mecanum wheel robots *holonomic_robot* parameter should be set to *true.*

DWA. ROS package *dwa_local_planner* is a modular DWA implementation with more flexible y-axis variables for holonomic robots than *base_local_planner*'s DWA. DWA discretely samples a robot's control space, performs a forward simulation for each sample, evaluates and filters each trajectory in the local costmap and finally selects a highest-scoring trajectory.

Eband. ROS package *eband_local_planner* implements the Elastic Band (EBand) method. An elastic band is a deformable collision free path generated by a global path incorporating information about obstacles proximity. A main drawback of *eband_local_planner* is that the ROS-based method implementation does not support an obstacle avoidance for moving obstacles.

TEB. ROS package *teb_local_planner* implements the Timed-Elastic-Band (TEB) method for an online trajectory optimization. A difference between TEB and EBand is that a local trajectory is optimized not by external forces, but by applying a cost function. For a holonomic robot *min_turning_radius* parameter should be set to 0 and *weight_kinematics_nh* parameter to 1.

4 Performance Comparison

Three experiments with different scenarios were conducted to identify a most suitable local planner for a mecanum wheeled robot. In the first experiment, a starting position of the robot was set to (0; −8) and a goal was set to (0; 8). A 2D occupancy grid map did not contain static obstacles. Next, several static obstacles were added to the world after a global map had been built. In the second experiment, the robot started at (−3.5; 0) and targeted to (3.5; 0). A cube with sides of 1 m moved linearly without an acceleration along a trajectory from point (2; 0) to point (−2; 0) and backwards, with 0.4 m/s linear velocity. In the third experiment, the cube moved from (0; 2) to (0; −2) and backwards. A reference trajectory depicted in Fig. 5 represents a suggested optimal path. Distance-optimal robot trajectories in the world with static objects are shown in Fig. 6.

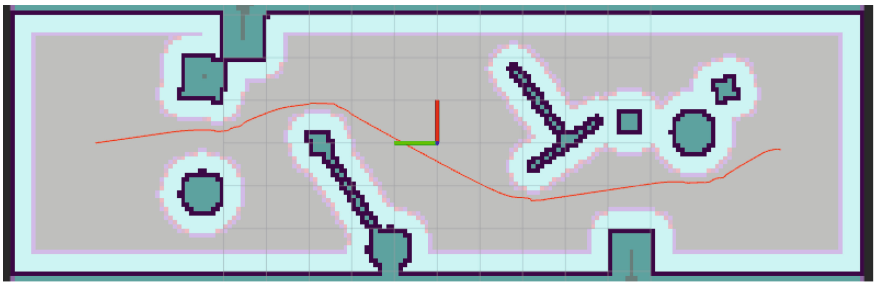
Fig. 5. Distance optimal robot trajectory.

Experiments showed that the Trajectory local planner never generated linear velocities along the Y-axis, and the omnidirectional robot moved as a differential wheeled robot with any planner settings. The Eband local planner trajectory was the smoothest, but this planner cannot handle dynamic obstacles and does not perform an online trajectory replanning. The drawbacks of the Eband are that the algorithm uses a global costmap updated dynamically, does not publish a response after reaching a target point and a task execution time is unmeasurable.

Table 1 demonstrates experimental results of local planners evaluated in the first world. Max T, Min T and Avg T stand for a maximum, minimum and average task execution time, respectively. Max D, Min D and Avg D denote a maximum, minimum and average path length, respectively. Success (Suc) column depicts how many times the robot reached the goal without obstacle collisions. Success with collision (SwC) column depicts how many times the robot reached the goal with at least one obstacle collision. Failed (F) column depicts how many times the robot failed to reach the goal. The TEB local planner showed the lowest minimum time and the lowest average time to complete the task. In one case, the robot with the TEB collided with an obstacle because the TEB heavily loaded the PC system and a frequency of publishing velocities to a command topic decreased. The DWA achieved the lowest minimum and the lowest average path length.

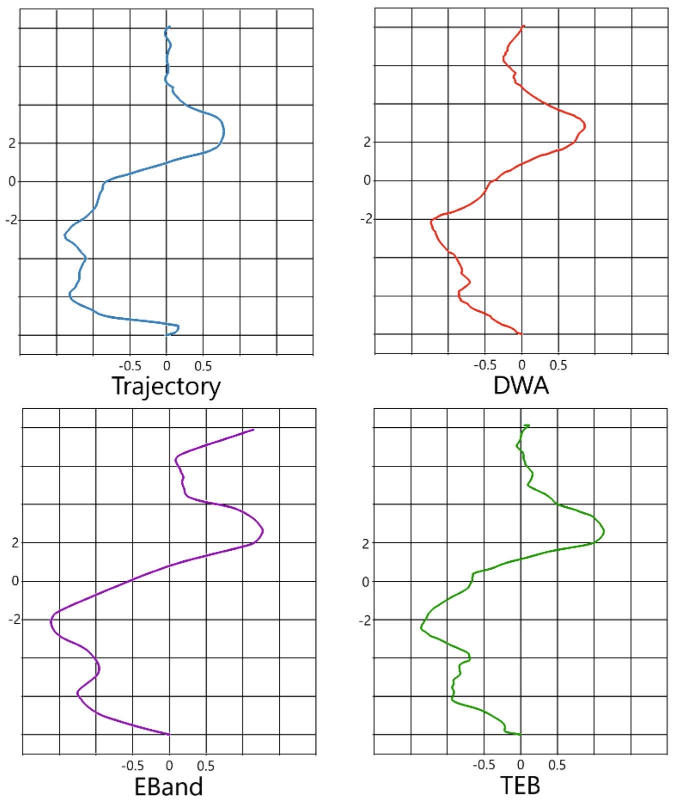

Fig. 6. Robot trajectories built by the local planners (in the world with static obstacles).

Table 1. ROS local planners' evaluation results (in the world with static obstacles).

Planner	Max T	Min T	Avg T	Max D	Min D	Avg D	Suc	SwC	F
Trajectory	188.6	64.8	100.9	23.2	17.9	19.6	50	0	0
DWA	67.1	39.6	43	18.9	17.2	17.7	50	0	0
EBand	-	-	-	19.2	18.1	18.9	50	0	0
TEB	91.6	32.4	38.3	43.4	17.9	19	49	1	0

In the experiment with the moving along the X-axis cube, all planners generated approximately the same trajectory (Fig. 7). Table 2 demonstrates experiments results of the local planners evaluated in the second world with the dynamic obstacle (a pattern of motion along X-axis). In this case, the TEB also showed the lowest minimum and the lowest average time required for a successful task completion. The Trajectory local planner showed the worst result within 50 experiments with only 7 successful and 15 (completely) failed. The EBand was successful in all 50 cases.

Table 2. ROS local planners evaluation results (the pattern of motion along X-axis).

Planner	Max T	Min T	Avg T	Max D	Min D	Avg D	Suc	SwC	F
Trajectory	222.5	23.4	40.8	15.4	2.7	8.6	7	28	15
DWA	41.2	18.6	22.1	13.3	1.7	8	24	23	3
EBand	-	-	-	9	8.3	8.6	50	0	0
TEB	31.41	16.2	20.8	16.9	8.5	11	47	2	1

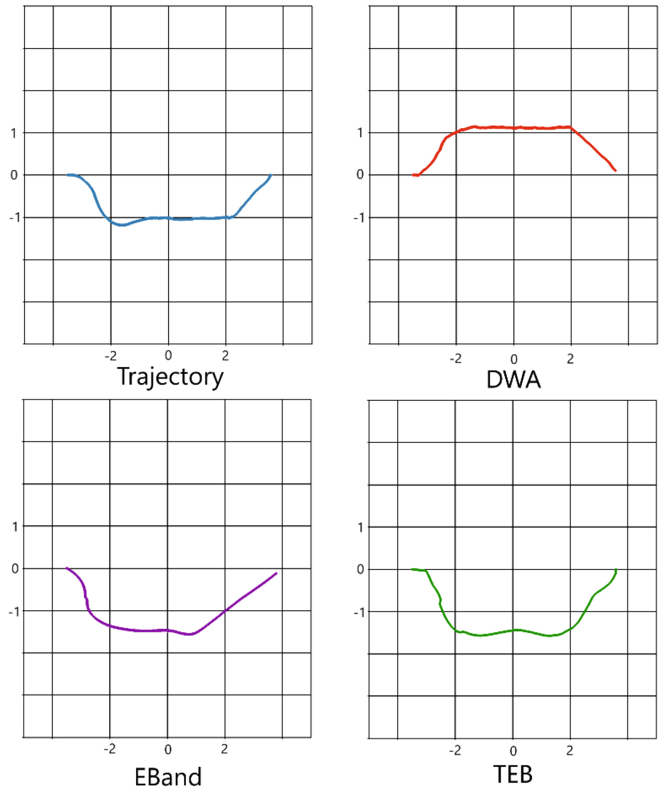

Fig. 7. Robot trajectories built by the local planners (the pattern of motion along X-axis).

In the case of the moving along the Y-axis cube, the DWA and the Trajectory Rollout showed a similar behavior (Fig. 8). When the cube appeared in the local costmap, the robot stopped and attempted to select an optimal movement trajectory. When the cube left the local costmap, the robot moved forward. The Eband and the TEB rebuilt the trajectory and continued the motion.

Table 3 demonstrates experimental results of the local planners evaluated in the second world with a dynamic obstacle (a pattern of motion along Y-axis). The TEB generated a maximum robot velocity. This method demonstrated the minimum task execution time and the least number of collisions with the obstacle. The worst result was shown by the DWA, which completely failed the task in 32 cases out of 50.

Table 3. ROS local planners evaluation results (the pattern of motion along Y-axis).

Planner	Max T	Min T	Avg T	Max D	Min D	Avg D	Suc	SwC	F
Trajectory	75	15.6	30.1	8.9	4.1	6.7	21	9	20
DWA	78	17.2	32.4	14.4	3.4	6.3	7	11	32
EBand	-	-	-	60	19.6	56.5	22	1	27
TEB	21.4	15	17.3	11	5.8	9	30	19	1

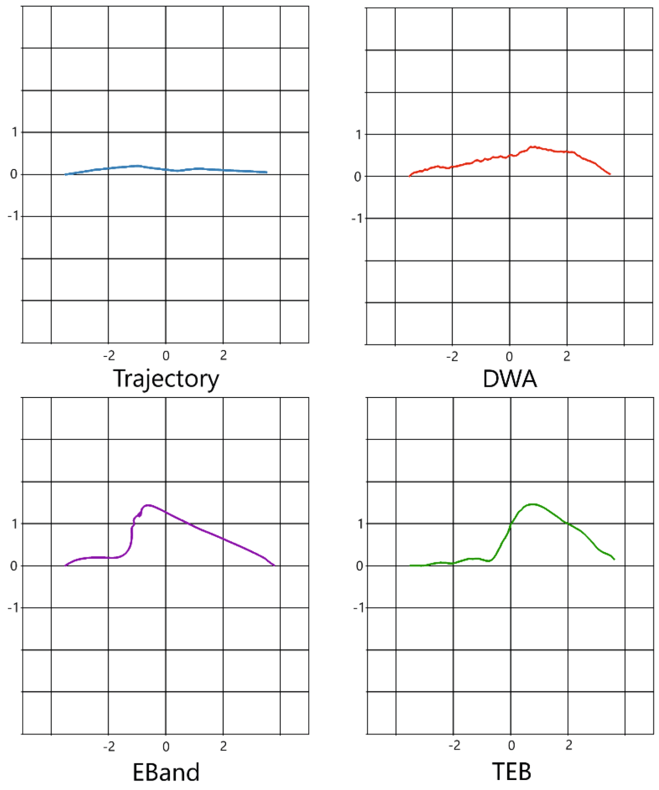

Fig. 8. Robot trajectories built by the local planners (the pattern of motion along Y-axis).

5 Conclusion

This paper presented a comparison of ROS local planners for mecanum wheeled robots (Trajectory Rollout, DWA, EBand, and TEB) and provided a benchmark that allowed to experimentally determine a recommended ROS local planner for a wheeled holonomic system. Virtual experiments were conducted in Gazebo simulator, which was used for modeling virtual environments with static and dynamic obstacles. Three types of virtual environments were employed with 50 virtual experiments within each environment. The algorithms were compared using a path length, a travelling time and a number of obstacle collisions.

The virtual experiments showed that the Trajectory Rollout local planner did not generate linear velocities along the Y-axis in all configurations; this planner demonstrated the worst task execution time and the longest trajectory path lengths. The DWA local planner handled static obstacles effectively, but performed poorly with dynamic obstacles. The EBand local planner did not rebuild a local motion trajectory when it worked on a non-renewable costmap of an explored environment; therefore, the use of this planner is possible only when a global costmap is dynamically updated, in which case the robot's trajectory will be rebuilt by a global planner. The TEB local planner achieved a significantly better performance in terms of a task execution time and showed the least number of obstacle collisions. Therefore, the TEB local planner could be recommended for dynamic scenes since it demonstrated the best results within a dynamic environment.

Acknowledgements. The reported study was funded by the Russian Science Foundation (RSF) and the Cabinet of Ministers of the Republic of Tatarstan according to the research project No. 22-21-20033.

References

1. Koubaa, A., et al.: Introduction to mobile robot path planning. In: Robot Path Planning and Cooperation. SCI, vol. 772, pp. 3–12. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-77042-0_1
2. Gul, F., Rahiman, W., Nazli Alhady, S.S.: A comprehensive study for robot navigation techniques. Cogent Eng. **6**(1), 1–25 (2019)
3. Hai, N.D.X., Nam, L.H.T., Thinh, N.T.: Remote healthcare for the elderly, patients by tele-presence robot. In: 2019 International Conference on System Science and Engineering (ICSSE), pp. 506–510 (2019)
4. Murphy, R.R.: Humans and robots in off-normal applications and emergencies. In: Chen, J. (ed.) Advances in Human Factors in Robots and Unmanned Systems: Proceedings of the AHFE 2019 International Conference on Human Factors in Robots and Unmanned Systems, July 24–28, 2019, Washington D.C., USA, pp. 171–180. Springer International Publishing, Cham (2020). https://doi.org/10.1007/978-3-030-20467-9_16
5. Taheri, H., Zhao, C.X.: Omnidirectional mobile robots, mechanisms and navigation approaches. Mech. Mach. Theor. **153**, 103958 (2020)
6. Safin, R., Lavrenov, R., Hsia, K.H., Maslak, E., Schiefermeier-Mach, N., Magid, E.: Modelling a turtlebot3 based delivery system for a smart hospital in gazebo. In: 2021 International Siberian Conference on Control and Communications (SIBCON), pp. 1–6 (2021)
7. Pimentel, F., Aquino, P.: Performance evaluation of ROS local trajectory planning algorithms to social navigation. In: 2019 Latin American Robotics Symposium (LARS), 2019 Brazilian Symposium on Robotics (SBR) and 2019 Workshop on Robotics in Education (WRE), pp. 156–161 (2019)

8. Vagale, A., Oucheikh, R., Bye, R.T., Osen, O.L., Fossen, T.I.: Path planning and collision avoidance for autonomous surface vehicles I: a review. J. Marine Sci. Technol. **26**(4), 1292–1306 (2021). https://doi.org/10.1007/s00773-020-00787-6
9. Cybulski, B., Wegierska, A., Granosik, G.: Accuracy comparison of navigation local planners on ROS-based mobile robot. In: 2019 12th International Workshop on Robot Motion and Control (RoMoCo), pp. 104–111 (2019)
10. Gerkey, B. P., Konolige, K.: Planning and control in unstructured terrain. In: ICRA Workshop on Path Planning on Costmaps (2008)
11. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robot. Autom. Mag. **4**(1), 23–33 (1997)
12. Gehrig, S.K., Stein, F.J.: Elastic bands to enhance vehicle following. In: 2001 IEEE Intelligent Transportation Systems, pp. 597–602 (2001)
13. Keller, M., Hoffmann, F., Hass, C., Bertram, T., Seewald, A.: Planning of optimal collision avoidance trajectories with timed elastic bands. IFAC Proc. Volum. **47**(3), 9822–9827 (2014)
14. Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2149–2154 (2004)
15. Apurin, A., et al.: LIRS-ArtBul: design, modelling and construction of an omnidirectional chassis for a modular multipurpose robotic platform. In: Interactive Collaborative Robotics: 7th International Conference (ICR 2022), pp. 16–18 (2022)
16. Gschwandtner, M., Kwitt, R., Uhl, A., Pree, W.: BlenSor: blender sensor simulation toolbox. In: Bebis, G., et al. (eds.) Advances in Visual Computing: 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26-28, 2011. Proceedings, Part II, pp. 199–208. Springer Berlin Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24031-7_20
17. Apurin, A., Abbyasov, B., Dobrokvashina, A., Bai, Y., Svinin, M., Magid, E.: Omniwheel chassis' model and plugin for gazebo simulator. Proc. Int. Conf. Artif. Life Robot. **28**, 170–173 (2023). https://doi.org/10.5954/ICAROB.2023.OS6-7
18. Takaya, K., Asai, T., Kroumov, V., Smarandache, F.: Simulation environment for mobile robots testing using ROS and Gazebo. In: 2016 20th International Conference on System Theory, Control and Computing (ICSTCC), pp. 96–101 (2016)
19. Iskhakova, A., Abbyasov, B., Mironchuk, T., Tsoy, T., Svinin, M., Magid, E.: LIRS-MazeGen: an easy-to-use blender extension for modeling maze-like environments for gazebo simulator. In: Ronzhin, A., Pshikhopov, V. (eds.) Frontiers in Robotics and Electromechanics, pp. 147–161. Springer Nature Singapore, Singapore (2023). https://doi.org/10.1007/978-981-19-7685-8_10
20. Valera, Á., Valero, F., Vallés, M., Besa, A., Mata, V., Llopis-Albert, C.: Navigation of autonomous light vehicles using an optimal trajectory planning algorithm. Sustainability **13**(3), 1233 (2021)