The Application of TurtleBot3 Burger for Obstacle Detection and ArUco Marker-Based Navigation

Azamat Turganbayev

Abstract

Autonomous Mobile robots (AMR) is one of the major fields in robotics. The application of AMR can be found in various fields, starting from healthcare and ending with energy resources sector. The following project for CSCI 5551 course will focus on the development of obstacle detection and the application of ArUco markers for navigation of TurtleBot3 Burger robot both in simulation and real-life environment. The implementation of Raspberry-Pi camera in combination with LiDAR will allow TurtleBot to follow the marker for navigation. The ROS2 Humble will be used for the project due to the technical inability of the laptop to use ROS Noetic. The current project goal is not final and can be modified.

1 Problem Description

1.1 Introduction

Autonomous mobile robots (AMR) are devices that can perform their tasks in various kinds of environments: whether it is indoor or outdoor. AMR is able to move fully independently or with minor human intervention by predefined path. Today the field of AMR is a major robotics research field with tremendous expanding rate. AMRs can be used in various fields due to their abilities. For example: energy infrastructure sector [1], patrolling, personal/medical care [2], reconnaissance and surveillance [3] transportation [4, 5], urban search and rescue applications [6] and other industrial and non industrial applications. To achieve the autonomous navigation the first AMRs were equipped with various planar sensors like laser range finder [7]. Mobile robots can differ based on their task and environment where they are going to be used: land, air or water. The design of any mobile robot consists of the following main parts: locomotion, perception and navigation [7–9]. Locomotion represents the methods of the AMRs' unbounded movement that can be derived and solved by understanding mechanism and kinematics, dynamics, and control theory. The scope of this project will focus on wheel based land AMRs due to their simplicity and cost-effectiveness compared to other types, like leg based AMRs [7–9]. TurtleBot3 Burger is an example of low-cost wheeled AMR that will be described in detail in the next subsection. Perception is responsible for environment identification via application of various sensors. Sensors allow robots to achieve simultaneous localization and mapping (SLAM), a concept that will be described in detail in the next sections [10]. There are two main classifications for sensors: proprioceptive/exteroceptive and active/passive sensors [7–9, 11]. Proprioceptive sensors are responsible for the robot's internal values, like motor speed, wheel load, joint angles, etc. Exteroceptive sensors gather the information about the environment. Cameras, LiDAR, ultrasonic sensors are some of the examples of exteroceptive sensors. Active sensors emit energy/signals and check the effect of it on the environment, while passive sensors get the information from the environment. Therefore, LiDAR is considered as an active exteroceptive sensor, and camera is a passive exteroceptive sensor. The combination of different types of sensors will allow AMRs to get the information about the environment and perform the tasks properly. Lastly, robot is considered as AMR if only it is able to do the following things: gather information about environment from sensors and transfer it into meaningful data (perception), identify its position in space (localization), understand and make a decision on how to reach the desired goal (cognition), and regulate actuators outputs to follow the desired trajectory (motion control) [7–10]. All of these 4 aspects can be combined into one single and most important skill for AMR called navigation. Mobile robot's navigation can be represented in three tasks:

1. Build an environmental model in the form of a map
2. Generate a smooth trajectory from a starting to the desired position in space
3. Initiate the movement along the created trajectory implementing obstacle avoidance algorithm

Apart from the kinematics aspects for navigation, computer vision can also be considered as an area to aid navigation. After getting the understanding of what AMR is, the next subsection will describe the project idea in detail, where TurtleBot3 mobile robot will be used to achieve the desired goals.

1.2 Project Description

TurtleBot3 Burger robot is one of the variations of an affordable multifunctional device that can find its application in various fields including education and research. First ever created TurtleBot was assembled 15 years ago at Willow Garage research laboratory by Melonee Wise and Tully Foote [12]. Unlike other variations of TurtleBot, TurtleBot3 is considerably smaller both by size and price, customizable and is functional and qualitative enough to perform the same duties as the other iterations of the robot. The main functions of TurtleBot3 include SLAM and navigation. Running SLAM algorithms and navigation is possible for such a small platform thanks to integrating compact and cost-effective components: a 360° LiDAR sensor for SLAM & navigation, a single board computer (SBC) Raspberry Pi for robust embedded system, an open source control board OpenCR 1.0 for hardware & software communication, and 2 DYNAMIXEL servo motors for motion. TurtleBot robots are integrated with robot operating system (ROS). ROS is an open-source framework that consists of various tools and software libraries for building robot applications [13]. TurtleBot3 robot is an ideal platform to start learning ROS.

For this project, the idea is to utilize all of the components of TurtleBot3 and also implement Raspberry-Pi camera for ArUco marker tracking navigation. ArUco markers are square markers widely used in computer vision for different tasks including pose estimation and navigation [14]. The combination of active exteroceptive LiDAR and passive exteroceptive camera sensor will allow the robot to track the marker and identify its depth, enabling autonomous movement. In addition, the obstacle detection algorithm will be used applying either LiDAR, camera or both and compare the efficiency. The next section will describe the application of different versions of TurtleBots for tasks such as crowd navigation, cooperative exploration.

1.3 Related Works

The exploration of an unknown environment is one of the most common tasks for AMRs. To perform this task efficiently, the authors in [15] demonstrated the exploration method of corridors by applying reinforcement learning and using only RGB-D Kinect sensor. All experiments were conducted in Gazebo simulation tool. Results show that TurtleBot successfully acquired obstacle avoidance ability and it could move in simulated environments that differed from the pre-training ones. In another work, authors implemented the crowd navigation policy for TurtleBot 2i firstly in simulation, and then successfully transferred them to a real-life environment [16]. To address the problem of robot decision making in crowd navigation, authors demonstrated the decentralized structural-Recurrent Neural Network (DS-RNN). With the help of DS-RNN it was possible to consider both spatial and temporal relationships in terms of crowd navigation. However, the following approach considered only the human-robot interaction, or in other words it did not cover the interaction between humans in the crowd. Therefore, authors in [17] developed a novel recurrent graph neural network that also considers human-human interaction via predicting human trajectories for several timesteps. The work was also done in simulation first and the learned policy was transferred to TurtleBot afterwards. In the next example, the authors demonstrated the successful application of AMR equipped with affordable Microsoft Kinect XBOX 360 RGB-D sensor for SLAM and autonomous navigation using Gazebo and demonstrating the results using rviz tools [18]. In the recent work done by [19], authors demonstrated the real-time navigation of TurtleBot in human-centered environments via the application of a novel vision-language model (VLM). The goal was to develop AMR's cognition in a way that would socially compliant with human expectations. Experiments that were conducted in 4 different scenarios showed that VLM demonstrated considerable improvement in average success rate compared to Dynamic-Window Approach (DWA) and Behavior Cloning (BC) methods. Another interesting application is the collaboration between the TurtleBot and an Unmanned Aerial Vehicle (UAV) for cooperative exploration [20]. The following cooperation can be applied for search and rescue missions in hard-to-reach environments. The setup was tested using TurtleBot 2 and a Parrot Bebop 2 UAV and experiments showed that such cooperation is feasible.

After summarizing the related works, this project will focus on the integration of a camera in cooperation with a LiDAR for obstacle avoidance and ArUco marker navigation. The combination of the camera and LiDAR will allow to track the marker and get the depth information. The integration of second camera might also help to get the depth information, however, since LiDAR was already integrated in the robot, additional camera would lust overload the whole system.

2 Tasks/Timeline

As it was mentioned above the following project will consider TurtleBot3 Burger for applying navigation algorithms. The idea for the project is to develop obstacle avoidance algorithm for TurtleBot and implement the Raspberry-Pi camera in combination with LiDAR for navigation via ArUco marker tracking. Camera will detect the marker while LiDAR will give the depth data so TurtleBot could navigate via it. Another approach might be the application of two cameras for stereo matching to estimate the depth, however, since TurtleBot is already equipped with LiDAR, such addition will be redundant. The project can be considered as completed when the following tasks will be done:

1. Install required ROS packages for TurtleBot3 Burger and run it in Gazebo with teleoperation **(Done)**
2. Assemble real TurtleBot3 Burger and run it with teleoperation **(Done)**
3. Set up Raspberry-Pi camera in TurtleBot and test its performance **(Done)**
4. Develop obstacle detection algorithm using data from LiDAR **(Done)**
5. Transfer the obstacle avoidance algorithm to TurtleBot3 Burger, test it in real environment and adjust if necessary (Done)
6. Develop ArUco marker tracking navigation and test in real-life environment **(Done)**

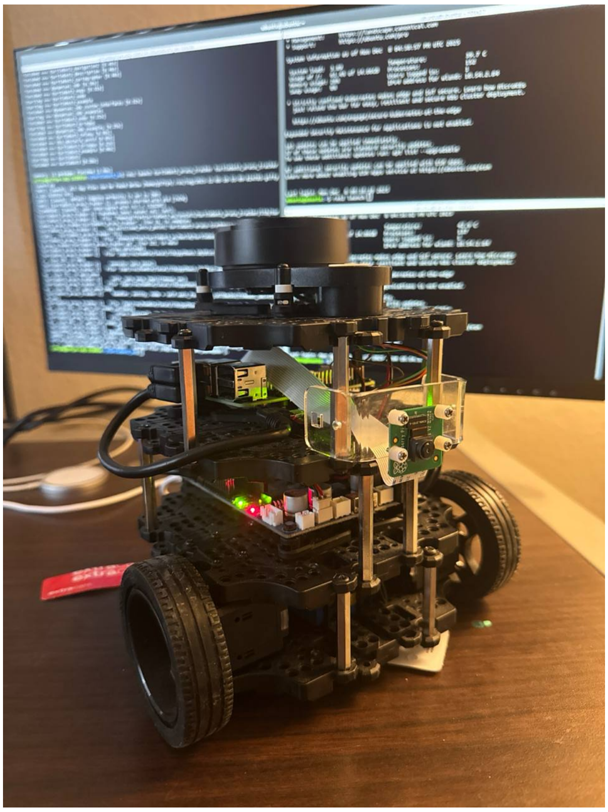


Figure 1: Assembled Turtlebot3 Burger Robot with integrated Raspberry-Pi Camera

3 Risk Management

The following project was initially considered to be done by a single person in the scope of the CSCI 5551 course, but then additional two members joined the group to fulfill the requirements of the EE 5271 course's final project and helped to finalize the project. Due to the lack of opportunity to work in the laboratory for a long period of time and for safety reasons, first of all all work was done and tested in simulation. Then, the obtained results were transferred to the real robot and tested in real-life scenarios. In addition, despite the fact that ROS Noetic has larger community support compared to the ROS2 Humble, this project will be done using ROS 2 for two reasons:

1. ROS Noetic has already reached its end-of-life date and will not be supported anymore.
2. The laptop on which all of the work will be conducted does not support the Ubuntu 20.04 version due to the Kernel version incompatibility. The application of virtual machine is not considered since opportunities are limited in such case

In addition, a calibration of LiDAR for real-life environment should be considered. During the simulation in Gazebo the LiDAR uses a perfect version of a sensor that does not gather noise. However, in the real environment, LiDAR can gather noises and therefore the obstacle avoidance might work inproperly. Therefore, it should be calibrated to filter out the noise.

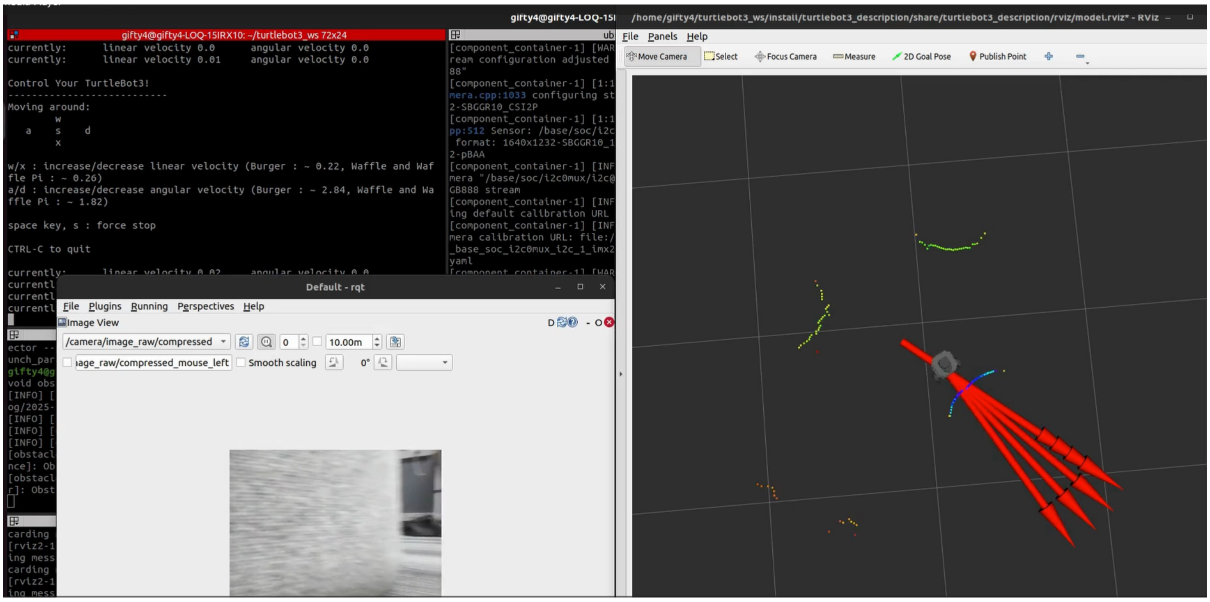
It has to be noted that initially TurtleBot3 Burger is not integrated with raspberry-pi camera. It means that the camera frame should be integrated in the robot to get the correct pose estimation of the markers.

To implement the ArUco marker navigation, for this project the environment with sufficient amount of light and a static position of the marker was considered due to the limited amount of time on the project. As a result, camera was calibrated in the light environment and all demo versions were recorded in such conditions. The marker navigation with a dynamic change in pose and in harsh environments can be implemented in the future.

Lastly, the project idea is not finalized and can be updated. The course project can be developed in a conference proceeding in the future, where the addition of crowd navigation mentioned in [15, 16, 19] or the combination of aerial and land AMR [20] can be implemented.

4 Results and Insights

The following project successfully integrated the LiDAR-based obstacle avoidance and the combination of LiDAR and camera for ArUco-based visual navigation on the TurtleBot3 robot. For obstacle-avoidance algorithms two ROS nodes were developed: **obstacle_detector** and **obstacle_avoidance**. **Obstacle_detector** node was responsible for monitoring the front sector of LiDAR and publishing obstacle topics in Boolean, while **obstacle_avoidance** subscribed to these topics and overrode the velocity parameters of the robot so it would turn right and make a slight back maneuver whenever an obstacle was detected. In the Gazebo environment, the following pipeline worked without any issues. The obstacles were correctly identified within the threshold distance (30 cm) and robot was able to safely turn away from the obstacle and also allowing teleoperation commands to pass through when the path was clear. For real-life environment, the parameters of the real LiDAR were tuned to filter out the noise and as a result the same behavior was transferred to the real robot. The visualisation in Rviz can be seen in the fig. 2.



```
currently:      linear velocity 0.0       angular velocity 0.0
currently:      linear velocity 0.01      angular velocity 0.0

Control Your TurtleBot3!
---------------------------
Moving around:
        w
   a    s    d
        x

w/x : increase/decrease linear velocity (Burger : ~ 0.22, Waffle and Waf
fle Pi : ~ 0.26)
a/d : increase/decrease angular velocity (Burger : ~ 2.84, Waffle and Wa
ffle Pi : ~ 1.82)

space key, s : force stop

CTRL-C to quit
```

```
[component_container-1] [WAR
ream configuration adjusted
88"
[component_container-1] [1:1
mera.cpp:1033 configuring st
2-SBGGR10_CSI2P
[component_container-1] [1:1
pp:512 Sensor: /base/soc/i2c
 format: 1640x1232-SBGGR10_1
2-pBAA
[component_container-1] [INF
mera "/base/soc/i2c0mux/i2c@
GB888 stream
[component_container-1] [INF
ing default calibration URL
[component_container-1] [INF
mera calibration URL: file:/
_base_soc_i2c0mux_i2c_1_imx2
yaml
```

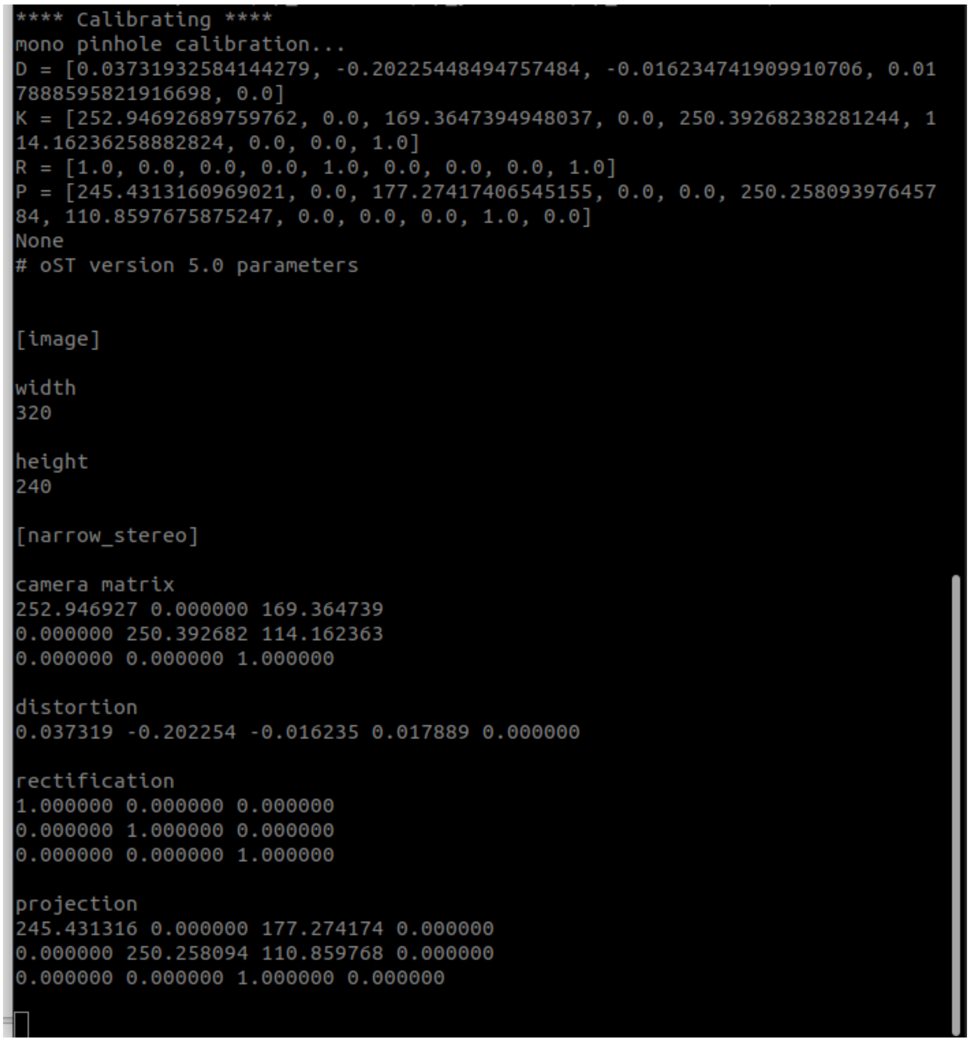
Figure 2: Obstacle Avoidance Visualisation in RViz

To implement the ArUco-based navigation, firstly it was required to do a camera calibration to get the accurate intrinsic parameters for raspberry-pi camera. This step was essential since ArUco pose esimation is highly sensitive to lens distortion, focal length accuracy and environment itself. The calibration was done using the ROS2 camera calibration package. The calibration procedure can be described in this way:

1. Prepare the 8x6 checkerboard pattern with a square size of 3 cm
2. Run the camera node in the ”RGB888” format with the 320x240 resolution for stable stream
3. Run calibration node with the parameters described in step 1
4. Proceed with the camera calibration by modifiying the position of the checkerboard
5. Obtain the intrinsic parameters and modify the calibration file so when camera node is turned on, the following parameters will be applied to it

The obtained and used parameters can be seen in fig. 3.

After camera calibration step was done, the next step was to implement the ArUco Tracker package. Initially the open source package was used for AruCo tracking [12]. However, the following package was only implemented for TurtleBot3 robot of the model Waffle which was integrated with the camera from the beginning. It means that initially the Turtlebot3 Bugrger did not have information about the pose of camera



```
**** Calibrating ****
mono pinhole calibration...
D = [0.03731932584144279, -0.20225448494757484, -0.016234741909910706, 0.01
7888595821916698, 0.0]
K = [252.94692689759762, 0.0, 169.3647394948037, 0.0, 250.39268238281244, 1
14.16236258882824, 0.0, 0.0, 1.0]
R = [1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0]
P = [245.4313160969021, 0.0, 177.27417406545155, 0.0, 0.0, 250.258093976457
84, 110.8597675875247, 0.0, 0.0, 0.0, 1.0, 0.0]
None
# oST version 5.0 parameters


[image]

width
320

height
240

[narrow_stereo]

camera matrix
252.946927 0.000000 169.364739
0.000000 250.392682 114.162363
0.000000 0.000000 1.000000

distortion
0.037319 -0.202254 -0.016235 0.017889 0.000000

rectification
1.000000 0.000000 0.000000
0.000000 1.000000 0.000000
0.000000 0.000000 1.000000

projection
245.431316 0.000000 177.274174 0.000000
0.000000 250.258094 110.859768 0.000000
0.000000 0.000000 1.000000 0.000000
```

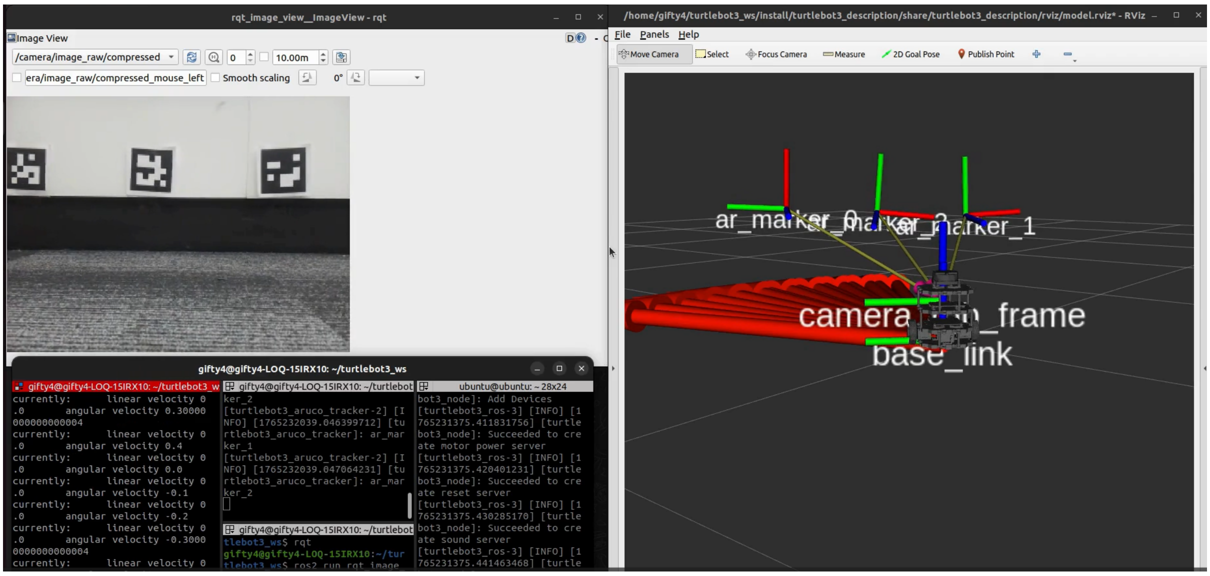
Figure 3: Camera Intrinsic Parameters Obtained After Calibration

on it. Therefore, a static frame was added to the package which approximately represented the real camera position on the robot (1 cm forward and 10 cm above the base link). This step allowed to correctly compute the marker's position relative to the robot's body frame. The following procedures were done for ArUCo marker tracking:

1. Bring up basic packages of TurtleBot3
2. Run camera node on TurtleBot3 with the calibrated parameters
3. Run ArUco tracker package with the specific marker size (10 cm in demo)
4. Run Rviz for visualization

Fig. 4 shows the Rviz vizualisation of tracking three ArUco markers with different IDs. The robot can not only track and differ the markers based on their IDs but also with the help of LiDAR can estimate the pose.

After correctly tracking ArUco markers, the **aruco_follower** node was developed to follow the ArUco markers. It allows the robot to orient itself toward the ArUco marker with a determined ID (ID:=0 for demo) and maintain a desired distance (distance:=10 cm for demo) via using both visual information from the AruCo tracker package and



```
currently:      linear velocity 0
.0       angular velocity 0.30000
000000000004
currently:      linear velocity 0
.0       angular velocity 0.4
currently:      linear velocity 0
.0       angular velocity 0.0
currently:      linear velocity 0
.0       angular velocity -0.1
currently:      linear velocity 0
.0       angular velocity -0.2
currently:      linear velocity 0
.0       angular velocity -0.3000
000000000004
currently:      linear velocity 0
```

```
ker_2
[turtlebot3_aruco_tracker-2] [I
NFO] [1765232039.046399712] [tu
rtlebot3_aruco_tracker]: ar_mar
ker_1
[turtlebot3_aruco_tracker-2] [I
NFO] [1765232039.047064231] [tu
rtlebot3_aruco_tracker]: ar_mar
ker_2
```

```
tlebot3_ws$ rqt
gifty4@gifty4-LOQ-15IRX10:~/tur
tlebot3_ws$ ros2 run rqt_image_
```

```
bot3_node]: Add Devices
[turtlebot3_ros-3] [INFO] [1
765231375.411831756] [turtle
bot3_node]: Succeeded to cre
ate motor power server
[turtlebot3_ros-3] [INFO] [1
765231375.420401231] [turtle
bot3_node]: Succeeded to cre
ate reset server
[turtlebot3_ros-3] [INFO] [1
765231375.430285170] [turtle
bot3_node]: Succeeded to cre
ate sound server
[turtlebot3_ros-3] [INFO] [1
765231375.441463468] [turtle
```

Figure 4: ArUco Marker Tracking in TurtleBot3 Burger with Integrated Camera Frame

range data from the LiDAR. To ensure the reliability, whenver the marker is not in the scope of camera's vision the node stops the robot. As a result the camera is used to track the marker and supplies the directional information and LiDAR measures the actual physical distance to the marker. As a result the node generates the commands to move toward the marker and drive to the marker until its in the desired distance ot the marker is out of the scope.

5 Lessons Learnt & Problems Encountered

Throughout the work on the obstacle avoidance and ArUco marker navigation aglortithms, several insights were learnt. Firstly, during the test of obstacle avoidance in real environment, the LiDAR was initially treated the same way as it was during the test in Gazebo. In Gazebo, LiDAR is simulated in ideal case scenario, where it did not gather noise. After implementing it in the real robot the noise filtering was implemented so obstacle avoidance would work appropriately. Also, assigning the Transform Frames properly was essential for ArUco navigation. Since the open source package was designed only for the TurtleBot3 waffle, the camera pose was assigned manually with respect to the base link. Finally, debugging in ROS environment is one of the major skills. The workspace, various packages, launch files require constant awareness and consistent rebuilding and sourcing.

6 Conclusion & Future Work

This project successfully demonstrated the implementation of the obstacle avoidance algorithm in the TurtleBot3 Burger in real environment and the integration of Raspberry Pi Camera for ArUco marker-based navigation. By applying camera calibration, correctly defining the camera frame and combining the directional information from camera and depth measurements from the LiDAR the TurtleBot achieved marker ArUco marker following behaviour on the real hardware. Despite the fact that real-world testing brought sensor noise, timing and environmental variability issues compared to the simulation in Gazebo environment, noise filtering and system restructuring allowed to transfer algorithms in the real environment successfully.

References

[1] D. Akimbay, Y. Bushanov, M. Nursultan, D. Zholtayev, D. Dauletiya, P. Rakhmetova, and A. Yeshmukhametov, "A novel hybrid robot for in-pipe maintenance and inspection," *IEEE Access*, vol. 13, pp. 168 498–168 511, 2025. doi: 10.1109/ACCESS.2025.3609193

[2] J. Mišeikis, P. Caroni, P. Duchamp, A. Gasser, R. Marko, N. Mišeikienė, F. Zwilling, C. de Castelbajac, L. Eicher, M. Früh, and H. Früh, "Lio-a personal robot assistant for human-robot interaction and care applications," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5339–5346, 2020. doi: 10.1109/LRA.2020.3007462

[3] D. Hougen, S. Benjaafar, J. Bonney, J. Budenske, M. Dvorak, M. Gini, H. French, D. Krantz, P. Li, F. Malver, B. Nelson, N. Papanikolopoulos, P. Rybski, S. Stoeter, R. Voyles, and K. Yesin, "A miniature robotic system for reconnaissance and surveillance," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, 2000, pp. 501–507 vol.1. doi: 10.1109/ROBOT.2000.844104

[4] N. Lashkari, M. Biglarbegian, and S. X. Yang, "Development of a novel robust control method for formation of heterogeneous multiple mobile robots with autonomous docking capability," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1759–1776, 2020. doi: 10.1109/TASE.2020.2977465

[5] T. Machado, T. Malheiro, S. Monteiro, W. Erlhagen, and E. Bicho, "Multi-constrained joint transportation tasks by teams of autonomous mobile robots using a dynamical systems approach," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 3111–3117. doi: 10.1109/ICRA.2016.7487477

[6] F. Niroui, K. Zhang, Z. Kashino, and G. Nejat, "Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 610–617, 2019. doi: 10.1109/LRA.2019.2891991

[7] M. B. Alatise and G. P. Hancke, "A review on challenges of autonomous mobile robot and sensor fusion methods," *IEEE Access*, vol. 8, pp. 39 830–39 846, 2020. doi: 10.1109/ACCESS.2020.2975643

[8] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, 2nd ed. The MIT Press, 2011. ISBN 0262015358

[9] F. Rubio, F. Valero, and C. Llopis-Albert, "A review of mobile robots: Concepts, methods, theoretical framework, and applications," *International Journal of*

Advanced Robotic Systems, vol. 16, no. 2, p. 1729881419839596, 2019. https://doi.org/10.1177/1729881419839596. doi: 10.1177/1729881419839596

[10] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, “Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016. doi: 10.1109/TRO.2016.2624754

[11] P. Corke, *Robotics, Vision and Control: Fundamental Algorithms In MATLAB, Second Edition*, 2nd ed. Springer Publishing Company, Incorporated, 2017. ISBN 3319544128

[12] “What is turtlebot?” https://www.turtlebot.com/about/.

[13] “What is ros?” https://www.ros.org/.

[14] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014. https://www.sciencedirect.com/science/article/pii/S0031320314000235. doi: https://doi.org/10.1016/j.patcog.2014.01.005

[15] L. Tai and M. Liu, “A robot exploration strategy based on q-learning network,” in *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, 2016, pp. 57–62. doi: 10.1109/RCAR.2016.7784001

[16] S. Liu, P. Chang, W. Liang, N. Chakraborty, and K. Driggs-Campbell, “Decentralized structural-rnn for robot crowd navigation with deep reinforcement learning,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 3517–3524. doi: 10.1109/ICRA48506.2021.9561595

[17] S. Liu, P. Chang, Z. Huang, N. Chakraborty, K. Hong, W. Liang, D. L. McPherson, J. Geng, and K. Driggs-Campbell, “Intention aware robot crowd navigation with attention-based interaction graph,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 12 015–12 021. doi: 10.1109/ICRA48891.2023.10160660

[18] R. Mishra and A. Javed, “Ros based service robot platform,” in *2018 4th International Conference on Control, Automation and Robotics (ICCAR)*, 2018, pp. 55–59. doi: 10.1109/ICCAR.2018.8384644

[19] D. Song, J. Liang, A. Payandeh, A. H. Raj, X. Xiao, and D. Manocha, “Vlm-social-nav: Socially aware robot navigation through scoring using vision-language models,” *IEEE Robotics and Automation Letters*, vol. 10, no. 1, pp. 508–515, 2025. doi: 10.1109/LRA.2024.3511409

[20] S. Hood, K. Benson, P. Hamod, D. Madison, J. M. O’Kane, and I. Rekleitis, “Bird’s eye view: Cooperative exploration by ugv and uav,” in *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2017, pp. 247–255. doi: 10.1109/ICUAS.2017.7991513