10th CMAME 2023

The 10th International Conference on MECHANICAL, AUTOMOTIVE AND MATERIALS ENGINEERING

PMAE The 5th International conference on Progress in Mechanical and Aerospace Engineering

Da Nang, Vietnam | December 20-22, 2023

Co-Sponsored by

Supported by

CMAME 2023
The 10th International Conference on Mechanical, Automotive and Materials Engineering
PMAE
The 5th International conference on Progress in Mechanical and Aerospace Engineering

DODGING DYNAMICAL OBSTACLES USING TURTLEBOT4 CAMERA FEED

Wei-Teng Chu
National Tsing Hua University, Taiwan

MURO Lab, UC San Diego
Advisor: Prof. Jorge Cortés

Da Nang, Vietnam | December 20-22, 2023

ACKNOWLEDGMENT

- Words cannot express my gratitude to Professor Jorge Cortés and Ph.D. students Neilabh Banzal, Parth Paritosh, and Scott Addams in the MURO Lab at UC San Diego. They provided me with some tips and guidance to help me complete the research. In addition, thanks should also go to the J. Yang & Family Foundation for sponsoring the scholarship and the University System of Taiwan for offering me the opportunity to conduct research at UC San Diego.

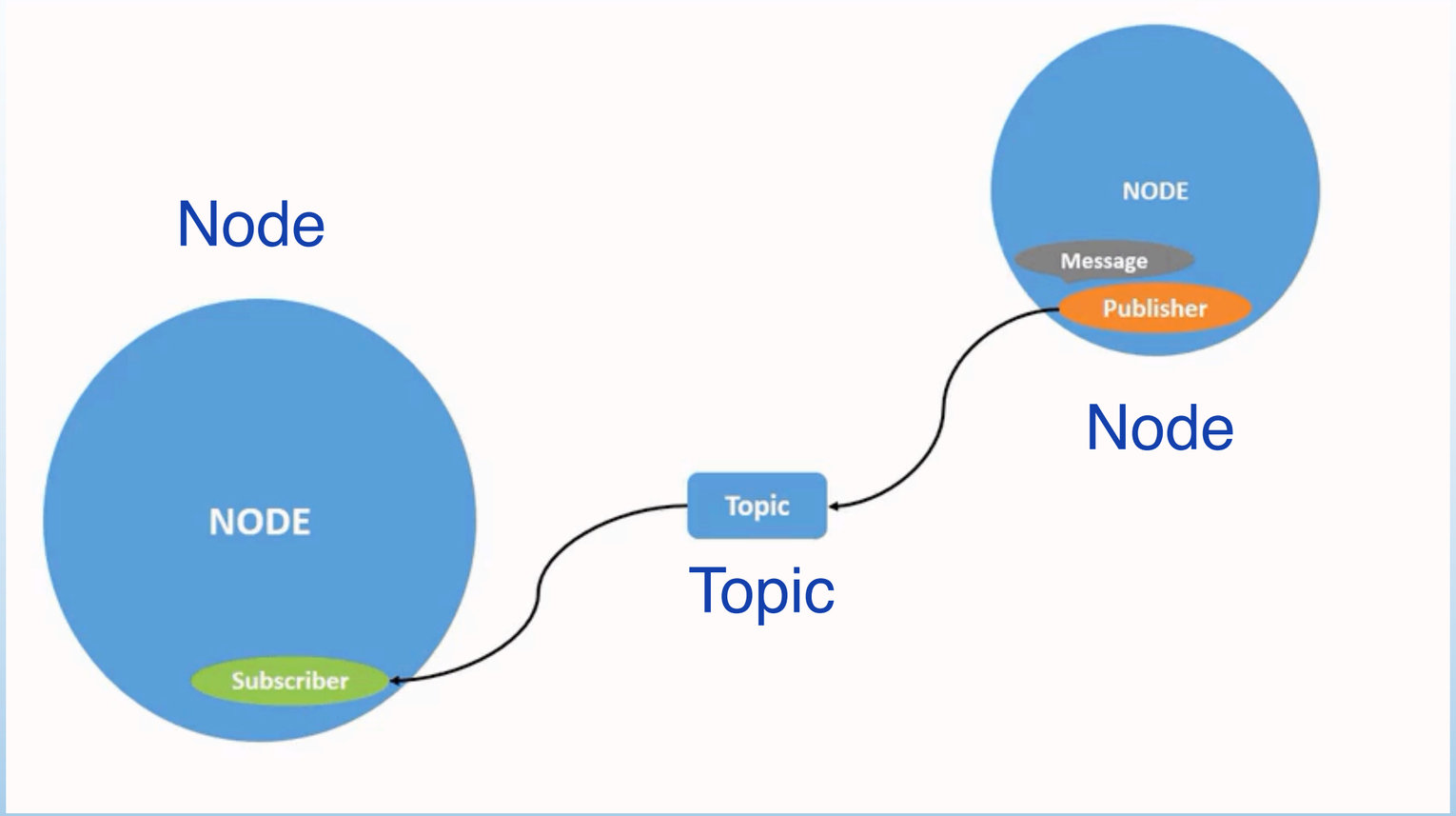
RESEARCH AIM

- Drive the robot to the designated destination.
- Dodge moving obstacle which will go across its path to the goal.
- The perception of the robot is based on CAMERA instead of LiDAR → Cheaper

ROS2
ROBOT OPERATING SYSTEM 2

Open-Source Framework	Develop and control the robot systems.
Flexible	Communication, real-time control, collaboration between robotics systems.
Powerful	Create, simulate, and deploy robotics applications across a wide range of platforms.

Topics

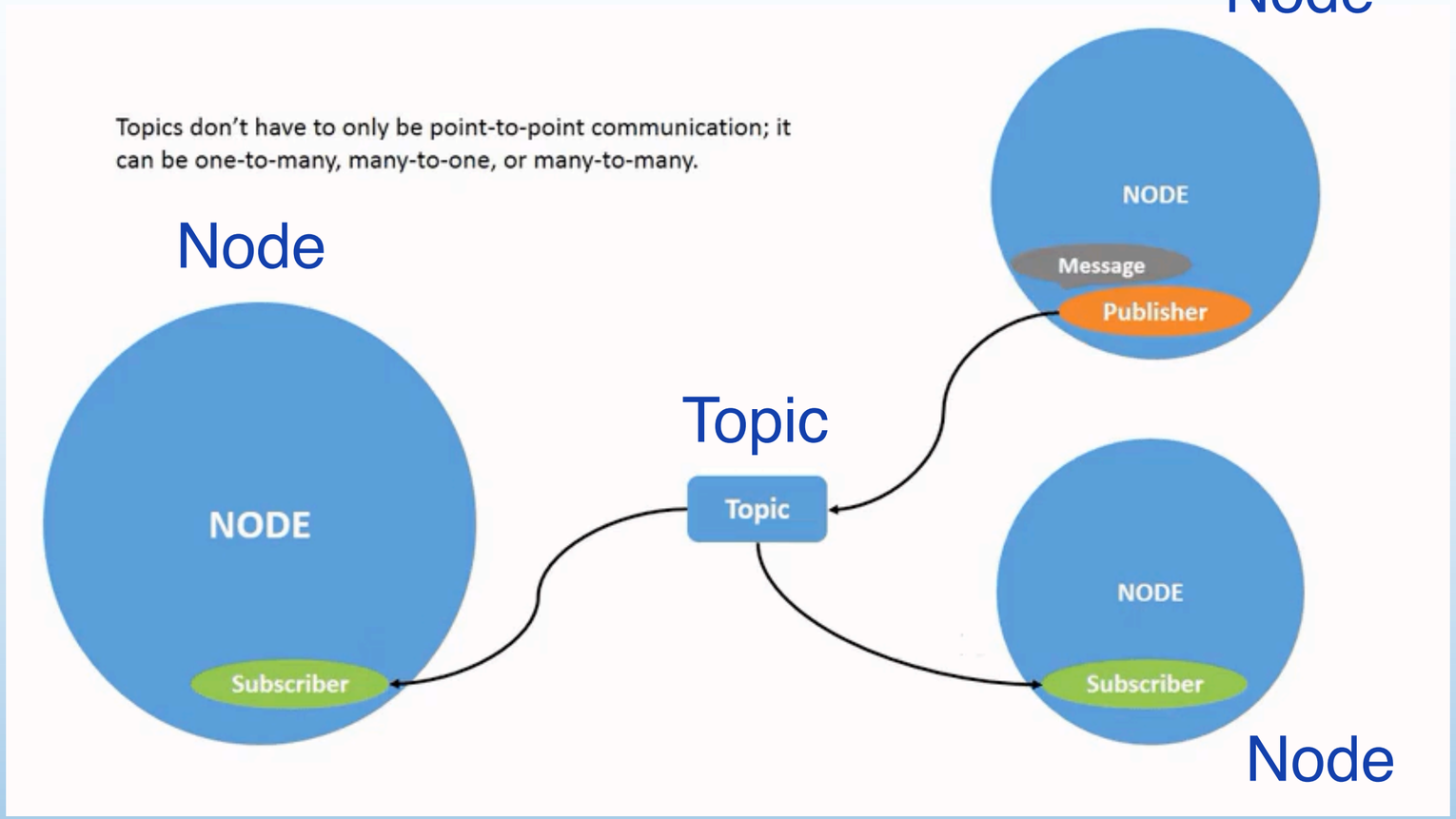

Node
Publisher
Subscriber
NODE
Message
Publisher
Node
NODE
Subscriber
Topic
Topic

Topics

Node

Publisher

Subscriber

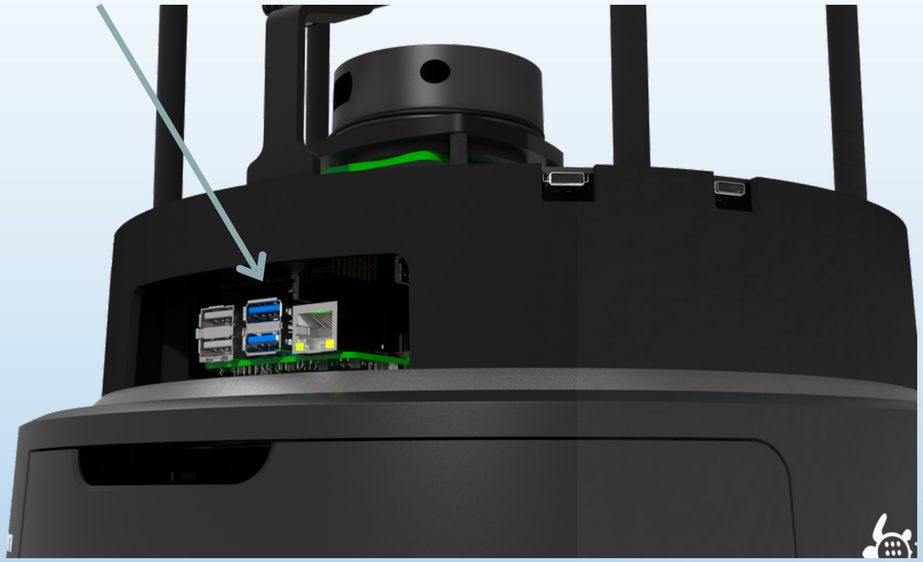


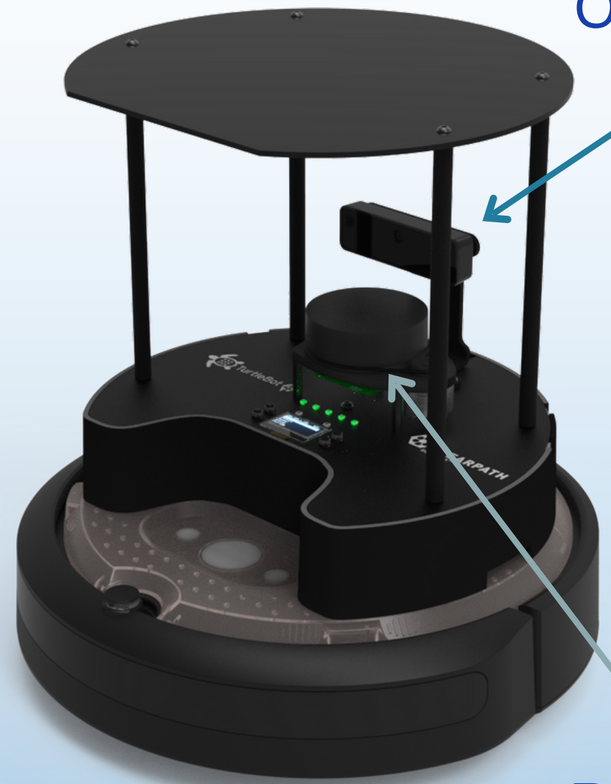
Subscriber

Node

TURTLEBOT4

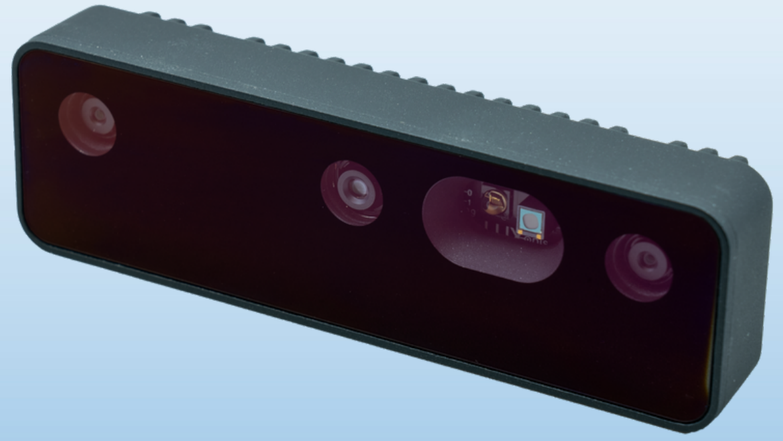
Raspberry Pi 4B

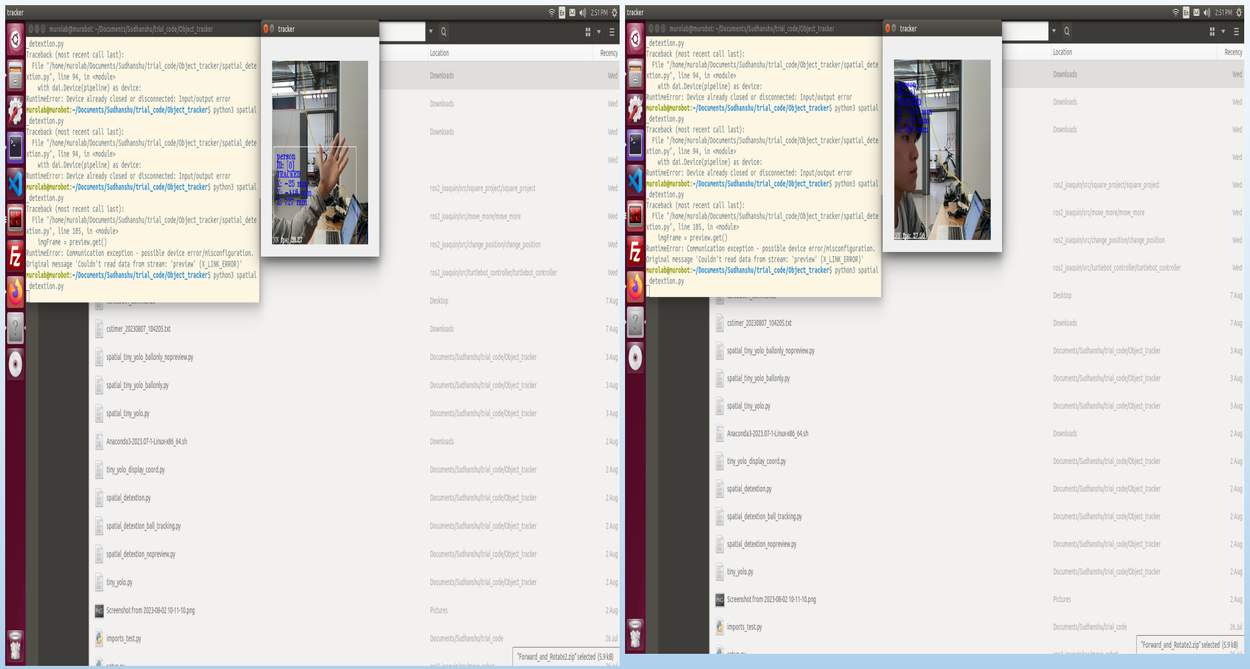




OAK-D Pro

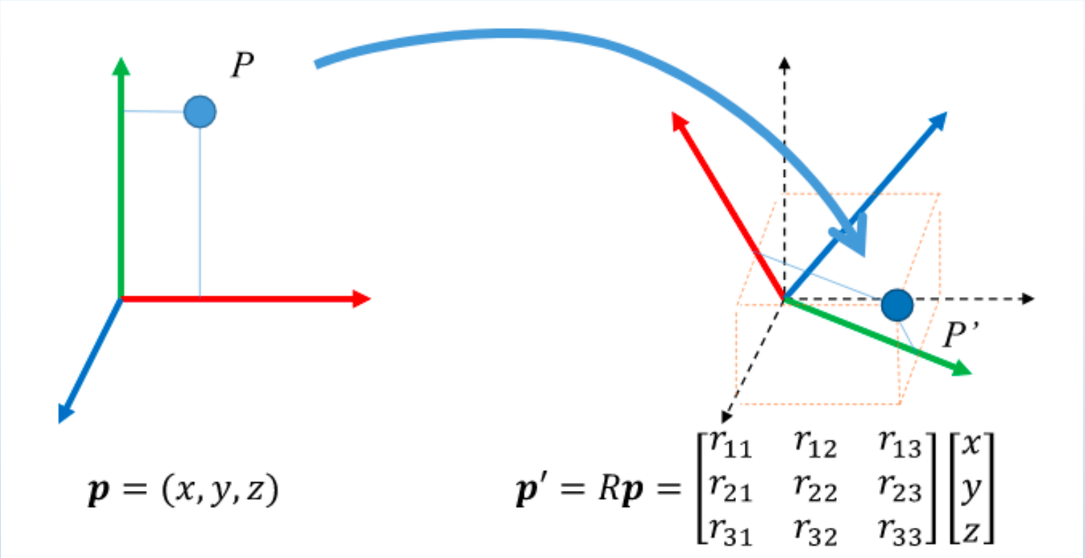
- Using the pretrained deep learning model provided by Luxonis.





Transformation?

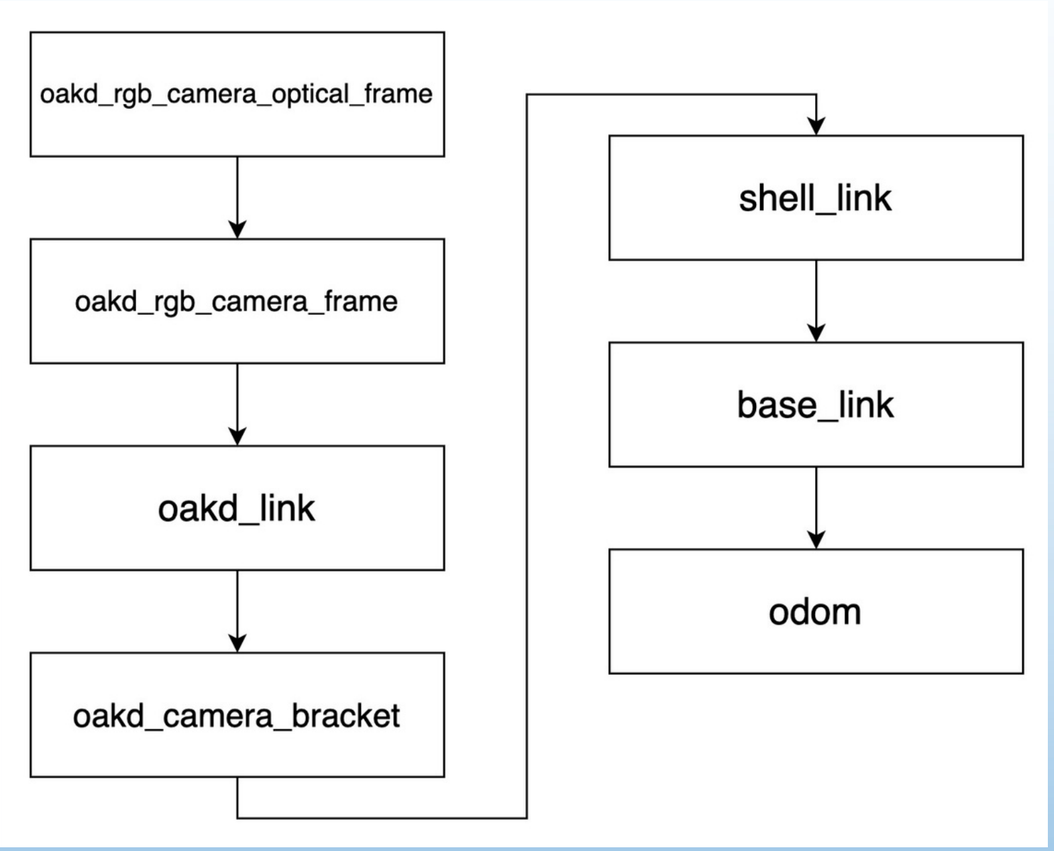
- Although the robot can see the obstacles through the camera, it actually doesn't know where the obstacles actually are.
- Transform the coordinates from the camera frame to the odometry frame.



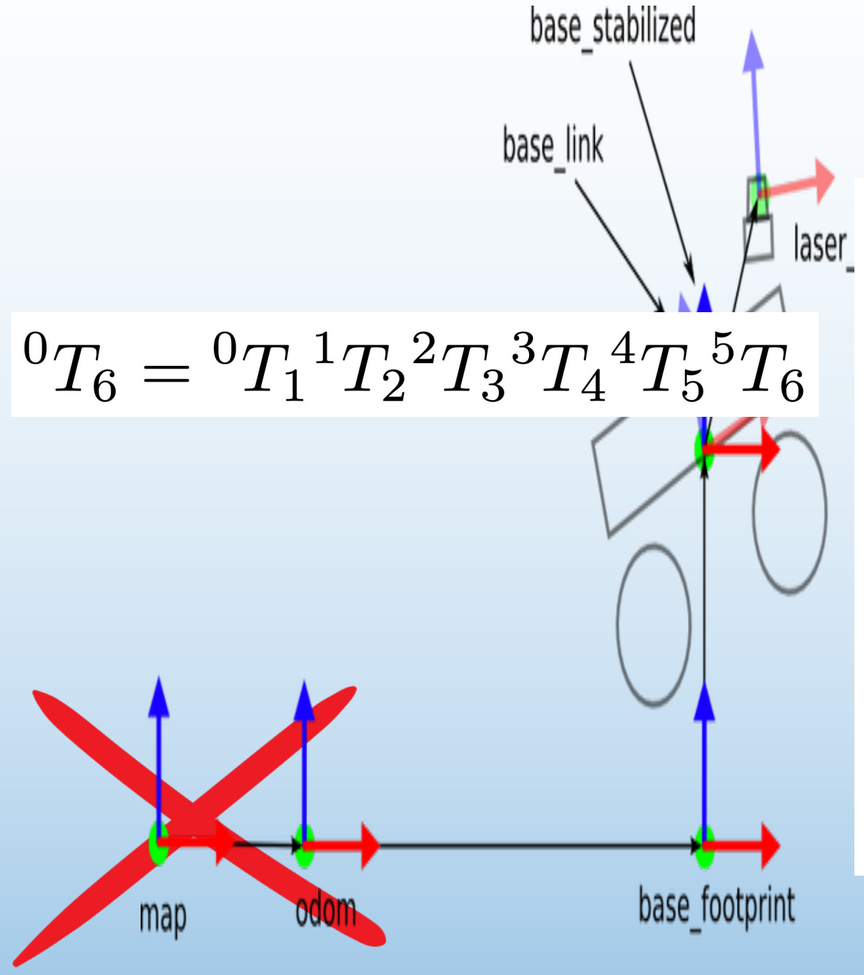
Forward Kinematics

$$^{0}T_{6} = {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{4}\,{}^{4}T_{5}\,{}^{5}T_{6}$$

- There are already many frames set inside the Turtlebot4.
- The relationship between them are also recorded in the system → Retrieve them from specific topic.
- Transform the coordinates step by step from the beginning to the goal frame.

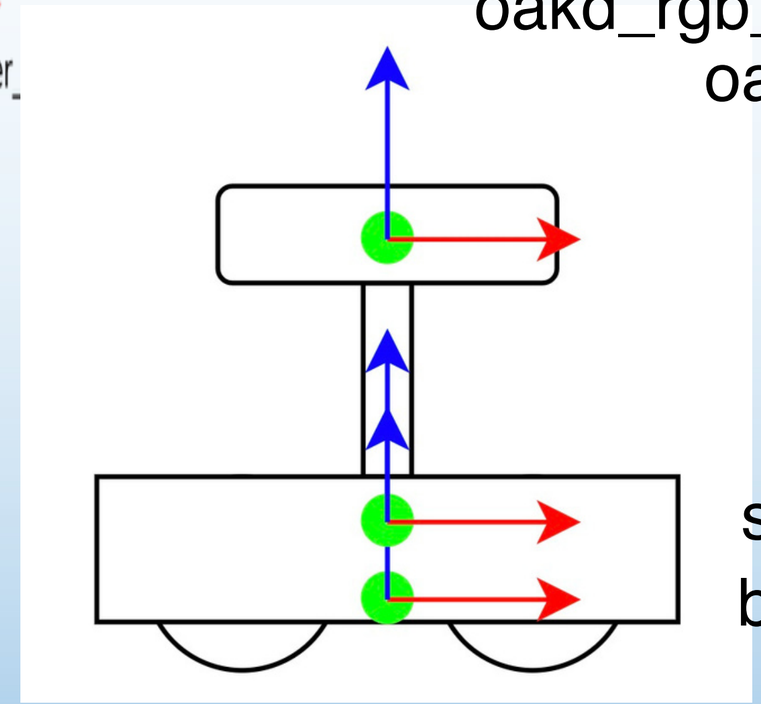


Forward Kinematics



$$^{0}T_{6} = {}^{0}T_{1}\,{}^{1}T_{2}\,{}^{2}T_{3}\,{}^{3}T_{4}\,{}^{4}T_{5}\,{}^{5}T_{6}$$

oakd_rgb_camera_optical_frame
oakd_rgb_camera_frame
oakd_link



shell_link
base_link

Forward Kinematics

$$R(Q) = \begin{bmatrix} 2(q_0^2 + q_1^2) - 1 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & 2(q_0^2 + q_2^2) - 1 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_1) & 2(q_0^2 + q_3^2) - 1 \end{bmatrix}$$

——Rotational Matrix

$$^0T_6 = {}^0T_1\,{}^1T_2\,{}^2T_3\,{}^3T_4\,{}^4T_5\,{}^5T_6$$

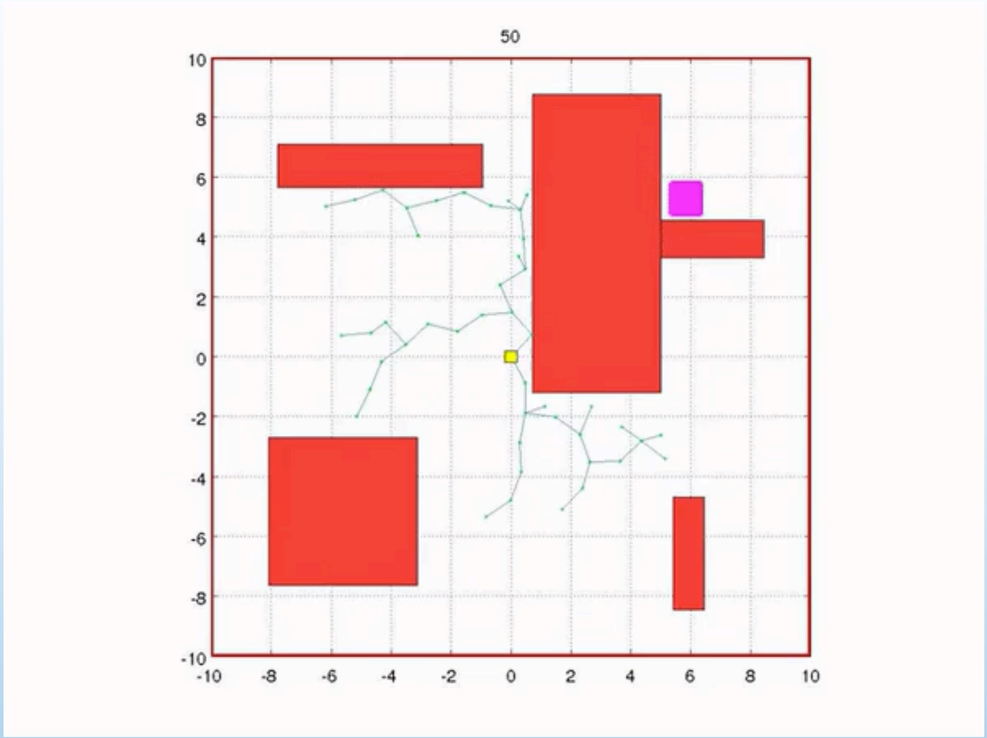
$$^aT_b = \begin{bmatrix} R_{3\times3}(Q) & \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}_{3\times1} \\ O_{1,3} & 1 \end{bmatrix}_{4\times4}$$

- q_1, q_2, q_3, q_4 are Quaternions, which describe orientation or rotations in 3D space.
- It is recorded in the Turtlebot system.

RRT*

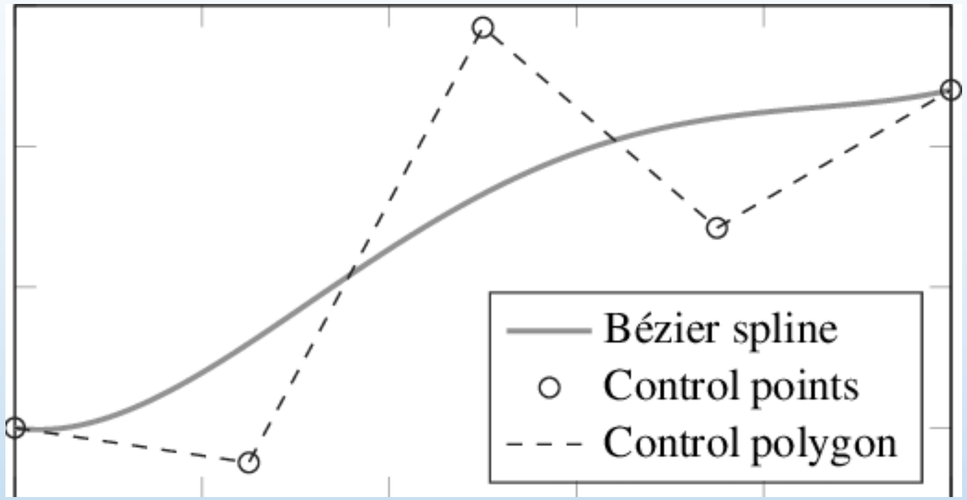
Rapidly-exploring Random Trees

- RRT* algorithm is used for path planning for the robot in the research.
- Convenient to find collision-free paths in complex environments.
- Build a tree structure by iteratively extending towards randomly selected points.



Bézier Curve

- After the path is planned by the RRT* algorithm, the path is usually not smooth enough for the robot to follow properly.
- Create Bézier Curve between each point of the RRT* path by interpolation.
- The path can therefore be smoothed out.



Kinematic and Dynamic Control of a Wheeled Mobile Robot

David DeVon and Timothy Bretl
Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
{devon,tbretl}@uiuc.edu

Controller

- The control can be analogous to that of the unicycle.
- The Turtlebot will measure the difference between the current state and the desired state to calculate the inputs to control the robot.
- $k_1 = 1100 \cdot \left(10^{-4}\right),\ k_2 = 5k_1$ when the robot is not dodging.
- $k_1 = 500 \cdot \left(10^{-4}\right),\ k_2 = 20k_1$ when the robot is dodging.

$$x = x_{current} - x_{destination}$$
$$y = y_{current} - y_{destination}$$
$$\theta = \theta_{current} - \theta_{destination}$$

$$z_1 = \theta$$
$$z_2 = x\cos\theta + y\sin\theta$$
$$z_3 = x\sin\theta - y\cos\theta$$

$$x_1 = z_1$$
$$x_2 = z_2$$
$$x_3 = -2z_3 + z_1 z_2$$

$$u_1 = -k_1 x_1 + \frac{k_2 x_3}{x_1^2 + x_2^2} x_2$$
$$u_2 = -k_1 x_2 - \frac{k_2 x_3}{x_1^2 + x_2^2} x_1$$

$$\omega = u_1$$
$$v = u_2 + z_3 u_1$$

Kinematic and Dynamic Control of a Wheeled Mobile Robot

David DeVon and Timothy Bretl
Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
{devon,tbretl}@uiuc.edu

Controller

- The control can be analogous to that of the unicycle.
- The Turtlebot will measure the difference between the current state and the desired state to calculate the inputs to control the robot.
- $k_1 = 1100 \cdot \left(10^{-4}\right),\ k_2 = 5k_1$ when the robot is not dodging.
- $k_1 = 500 \cdot \left(10^{-4}\right),\ k_2 = 20k_1$ when the robot is dodging.

$$x = x_{current} - x_{destination}$$
$$y = y_{current} - y_{destination}$$
$$\theta = \theta_{current} - \theta_{destination}$$

$$z_1 = \theta$$
$$z_2 = x\cos\theta + y\sin\theta$$
$$z_3 = x\sin\theta - y\cos\theta$$

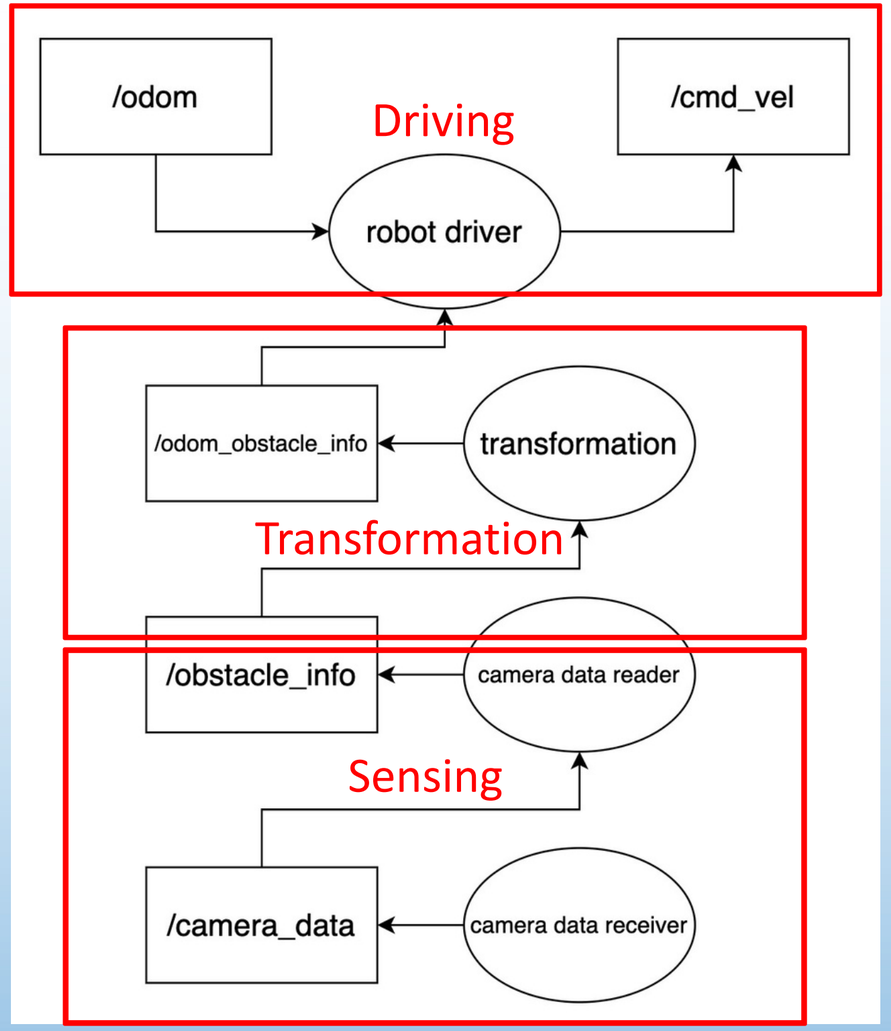
$$x_1 = z_1$$
$$x_2 = z_2$$
$$x_3 = -2z_3 + z_1 z_2$$

$$u_1 = -k_1 x_1 + \frac{k_2 x_3}{x_1^2 + x_2^2} x_2$$
$$u_2 = -k_1 x_2 - \frac{k_2 x_3}{x_1^2 + x_2^2} x_1$$

$$\omega = u_1$$
$$v = u_2 + z_3 u_1$$

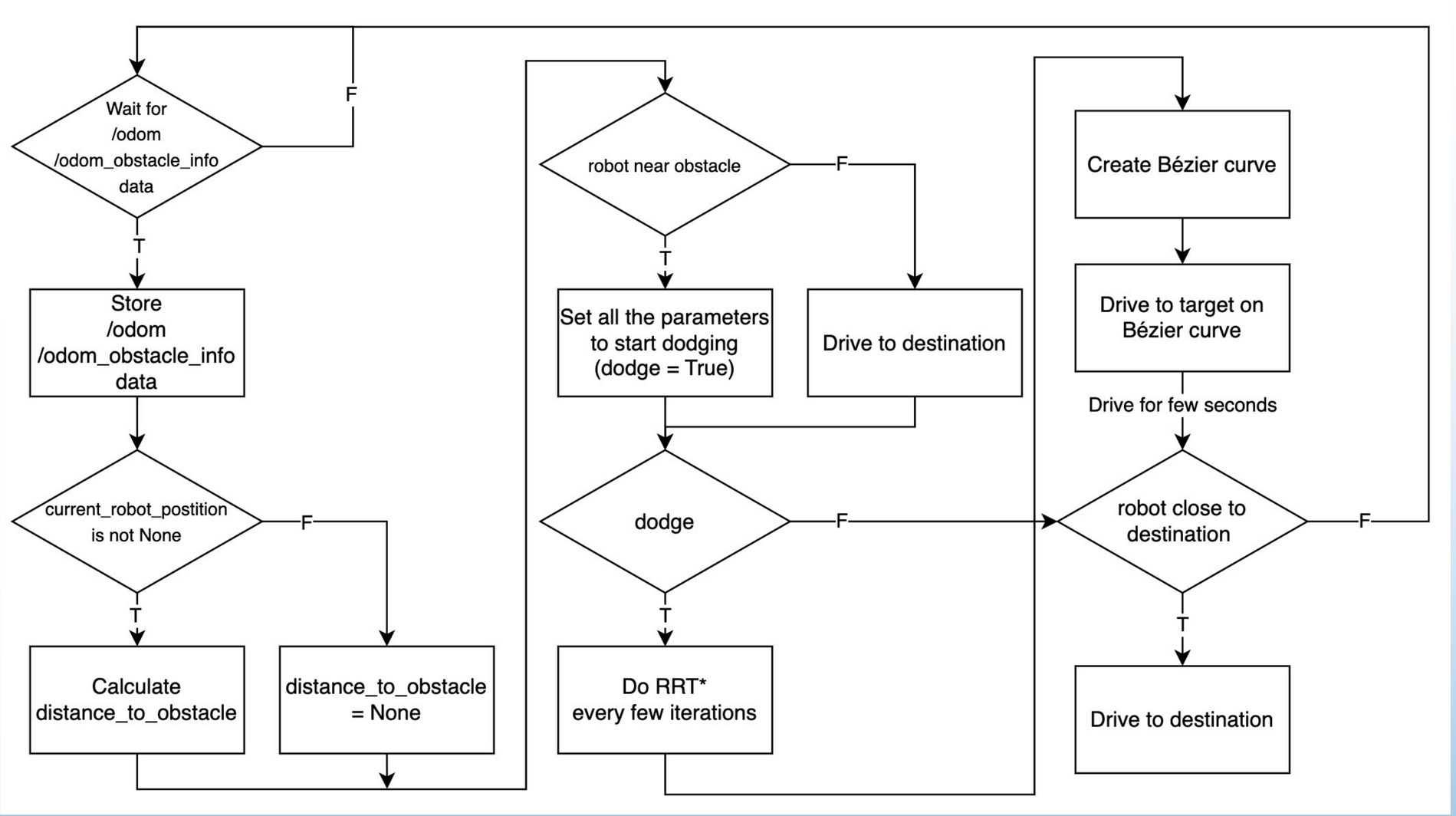
Implementation

- All the nodes and topics used in this research are shown in the figure on the right.
- The system is mainly composed of three parts → Sensing, Transformation, Driving.
- “/cmd_vel” is the node that drive the Turtlebot.



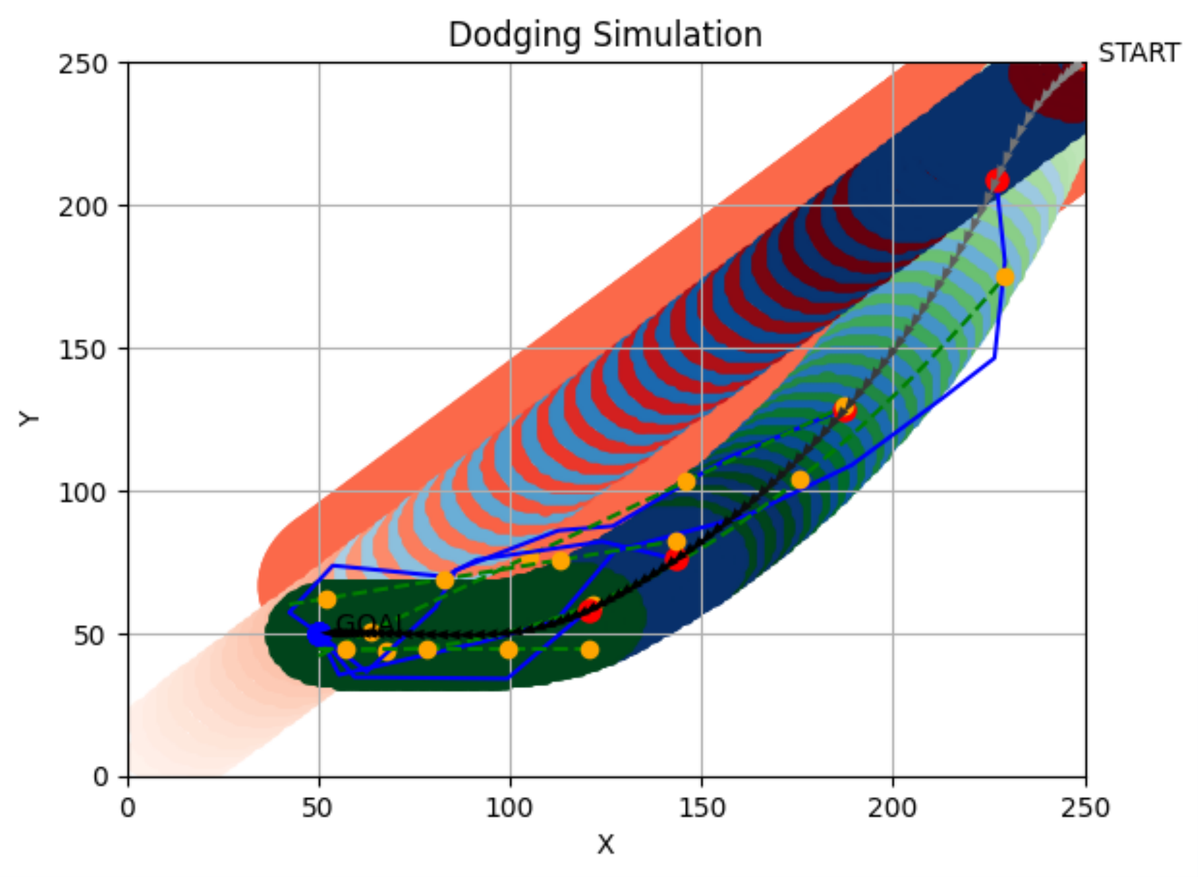
Robot Driver

Implementation

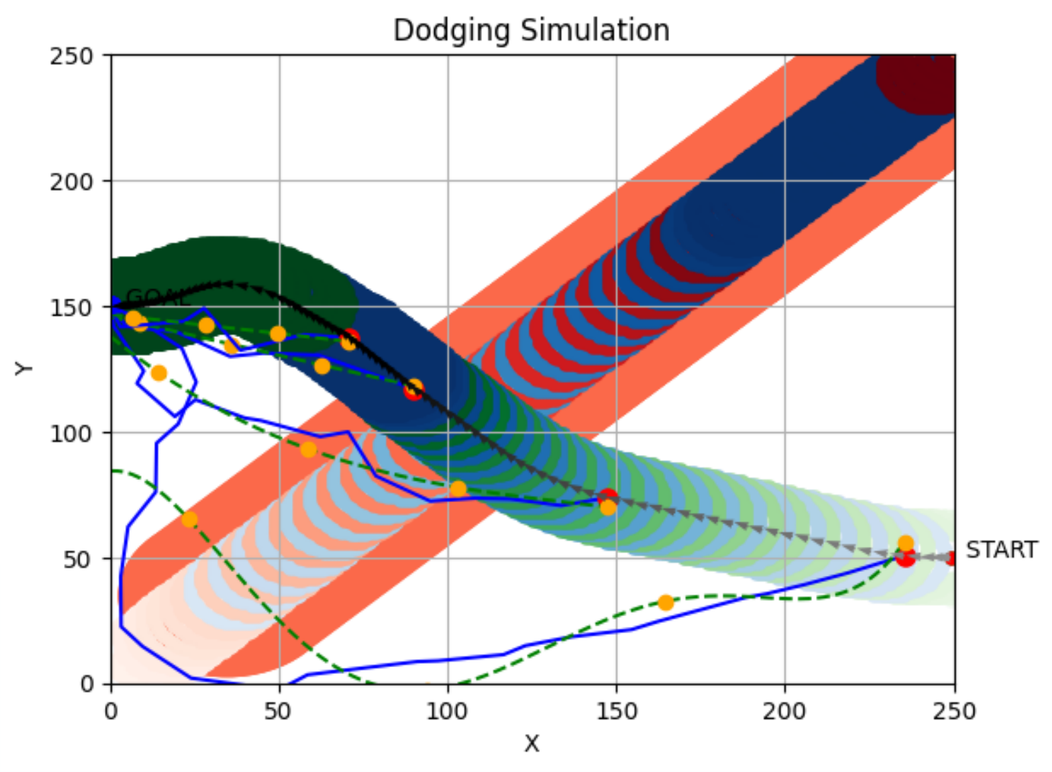


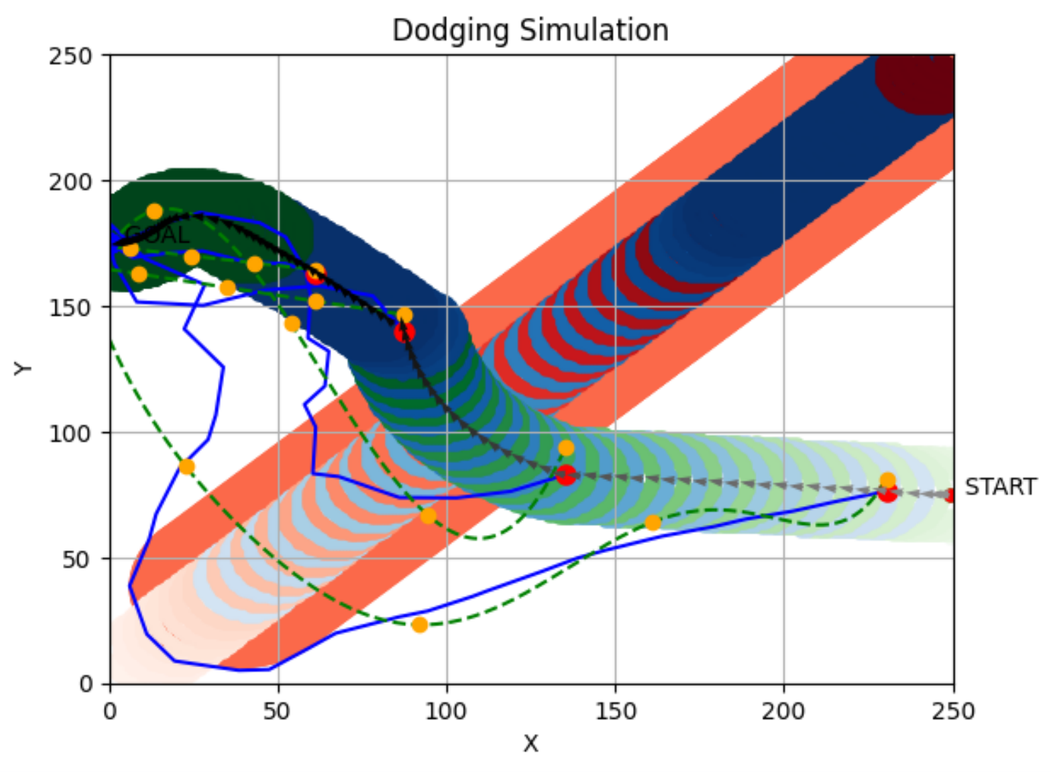
Simulation

- Obstacle starts from (0, 0) to (250, 250)
- Turtlebot starts from (250, 250) to (50, 50)
- Red dot → Start RRT* (blue line)
- Green dash line → Bézier Curve
- Yellow dot → Separate the curve
- Repeat the steps above until the robot reaches the destination

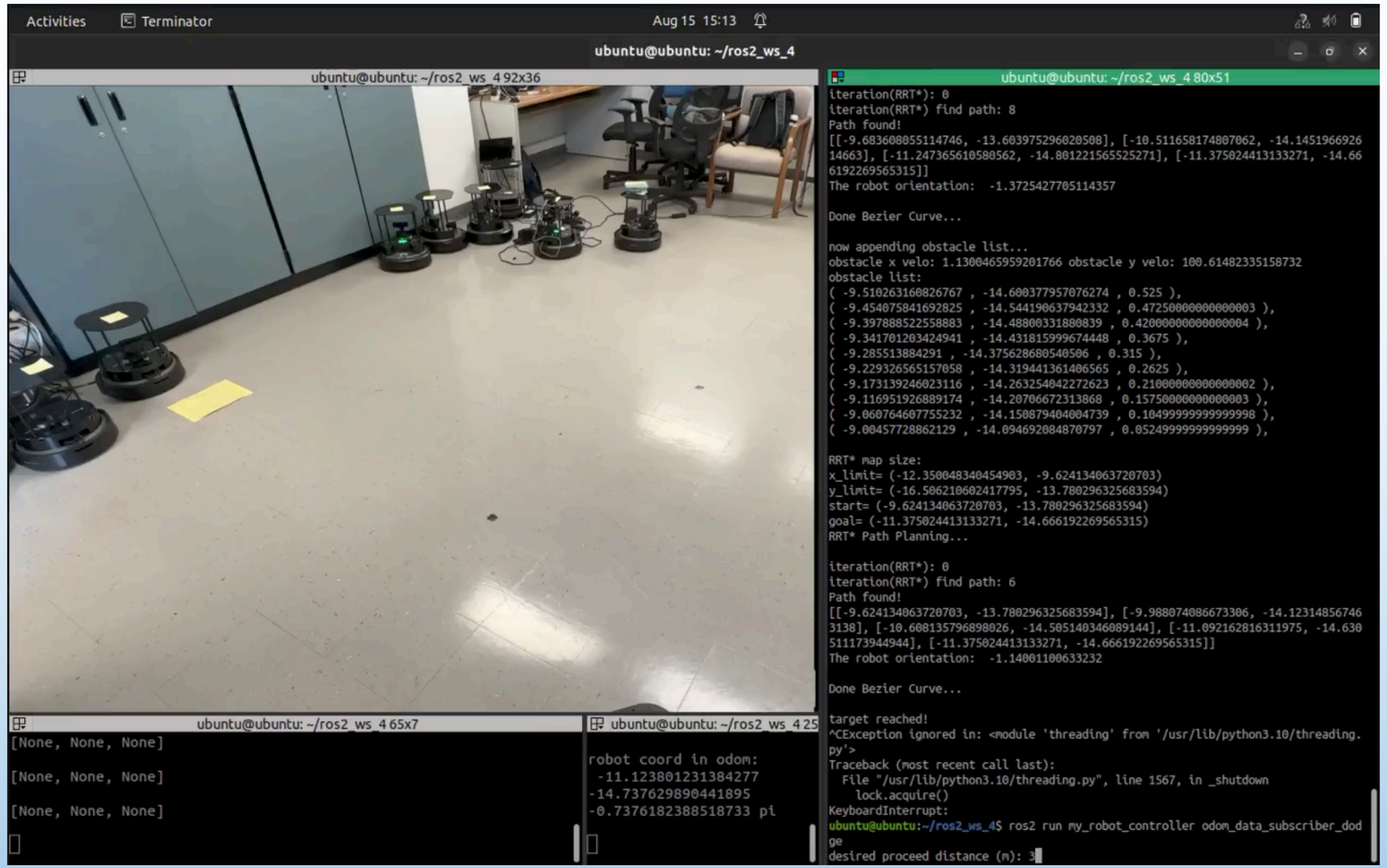


Simulation


Dodging Simulation
GOAL
START
X
Y
0
50
100
150
200
250


Dodging Simulation
GOAL
START
X
Y
0
50
100
150
200
250

Result


Activities
Terminator
Aug 15 15:13
ubuntu@ubuntu: ~/ros2_ws_4
ubuntu@ubuntu: ~/ros2_ws_4 92x36
ubuntu@ubuntu: ~/ros2_ws_4 80x51
iteration(RRT*): 0
iteration(RRT*) find path: 8
Path found!
[[-9.683608055114746, -13.603975296020508], [-10.511658174807062, -14.145196692614663], [-11.247365610580562, -14.801221565525271], [-11.375024413133271, -14.666192269565315]]
The robot orientation: -1.3725427705114357
Done Bezier Curve...
now appending obstacle list...
obstacle x velo: 1.1300465959201766 obstacle y velo: 100.61482335158732
obstacle list:
(-9.510263160826767 , -14.600377957076274 , 0.525),
(-9.454075841692825 , -14.544190637942332 , 0.4725000000000003),
(-9.397888522558883 , -14.48800331880839 , 0.4200000000000004),
(-9.341701203424941 , -14.431815999674448 , 0.3675),
(-9.285513884291 , -14.375628680540506 , 0.315),
(-9.229326565157058 , -14.319441361406565 , 0.2625),
(-9.173139246023116 , -14.263254042272623 , 0.2100000000000002),
(-9.116951926889174 , -14.20706672313868 , 0.1575000000000003),
(-9.060764607755232 , -14.150879404004739 , 0.10499999999999998),
(-9.00457728862129 , -14.094692084870797 , 0.05249999999999999),
RRT* map size:
x_limit= (-12.350048340454903, -9.624134063720703)
y_limit= (-16.506210602417795, -13.780296325683594)
start= (-9.624134063720703, -13.780296325683594)
goal= (-11.375024413133271, -14.666192269565315)
RRT* Path Planning...
iteration(RRT*): 0
iteration(RRT*) find path: 6
Path found!
[[-9.624134063720703, -13.780296325683594], [-9.988074086673306, -14.123148567463138], [-10.608135796898026, -14.505140346089144], [-11.092162816311975, -14.630511173944944], [-11.375024413133271, -14.666192269565315]]
The robot orientation: -1.14001100633232
Done Bezier Curve...
target reached!
^CException ignored in: <module 'threading' from '/usr/lib/python3.10/threading.py'>
Traceback (most recent call last):
File "/usr/lib/python3.10/threading.py", line 1567, in _shutdown
lock.acquire()
KeyboardInterrupt:
ubuntu@ubuntu:~/ros2_ws_4$ ros2 run my_robot_controller odom_data_subscriber_dodge
desired proceed distance (m): 3
ubuntu@ubuntu: ~/ros2_ws_4 65x7
[None, None, None]
[None, None, None]
[None, None, None]
ubuntu@ubuntu: ~/ros2_ws_4 25
robot coord in odom:
-11.123801231384277
-14.737629890441895
-0.7376182388518733 pi

References

- [1] D. Falanga, K. Kleber, and D. Scaramuzza, "Dynamic obstacle avoidance for quadrotors with event cameras," *Science Robotics*, vol. 5, no. 40, p. eaaz9712, 2020.
- [2] T.-T.-N. Nguyen, T.-D. Phan, M.-T. Duong, C.-T. Nguyen, H.-P. Ly, and M.-H. Le, "Sensor fusion of camera and 2d lidar for self-driving automobile in obstacle avoidance scenarios," in *2022 International Workshop on Intelligent Systems (IWIS)*, pp. 1–7, 2022.
- [3] P. Michel, J. Chestnutt, J. Kuffner, and T. Kanade, "Vision-guided humanoid footstep planning for dynamic environments," in *5th IEEE-RAS International Conference on Humanoid Robots, 2005.*, pp. 13–18, Dec 2005.
- [4] N. Ye, R. Wang, and N. Li, "A novel active object detection network based on historical scenes and movements," *International Journal of Computer Theory and Engineering*, vol. 13, pp. 79–83, 01 2021.
- [5] P. H. Kashika and R. B. Venkatapur, "Deep learning technique for object detection from panoramic video frames," International Journal of Computer Theory and Engineering, vol. 14, no. 1, pp. 20–26, 2022.
- [6] X. Xie, H. Li, and F. Hu, "The flocs target detection algorithm based on the three frame difference and enhanced method of the otsu," International Journal of Computer Theory and Engineering, vol. 7, no. 3, p. 197, 2015.
- [7] B. Siciliano and O. Khatib, Springer Handbook of Robotics. Springer Publishing Company, Incorporated, 2nd ed., 2016.
- [8] D. DeVon and T. Bretl, "Kinematic and dynamic control of a wheeled mobile robot," in 2007 IEEE/RSJ International Conference on Intel ligent Robots and Systems, pp. 4065–4070, 2007.
- https://docs.ros.org/en/rolling/Tutorials/Beginner-CLI-Tools/Understanding-ROS2-Topics/Understanding-ROS2-Topics.html
- https://turtlebot.github.io/turtlebot4-user-manual/
- https://docs.luxonis.com/projects/hardware/en/latest/pages/DM9098pro/
- https://automaticaddison.com/coordinate-frames-and-transforms-for-ros-based-mobile-robots/
- http://motion.cs.illinois.edu/RoboticSystems/CoordinateTransformations.html
- https://automaticaddison.com/how-to-convert-a-quaternion-to-a-rotation-matrix/
- https://husarion.com/tutorials/ros2-tutorials/7-transformation/
- https://en.wikipedia.org/wiki/B%C3%A9zier_curve
- https://www.researchgate.net/figure/Bezier-curve-with-five-control-points-and-control-polygon_fig3_327704259
- http://what-when-how.com/advanced-methods-in-computer-graphics/curves-and-surfaces-advanced-methods-in-computer-graphics-part-5/

Thanks for Listening!

10th CMAME 2023

The 10th International Conference on MECHANICAL, AUTOMOTIVE AND MATERIALS ENGINEERING

PMAE The 5th International conference on Progress in Mechanical and Aerospace Engineering

Da Nang, Vietnam | December 20-22, 2023

Co-Sponsored by

Supported by