

Trident: Design and Simulation of a Differential-Drive Pick and Place Robot

Joshua John Leo Valan Johnson

Roll No. 110123044

National Institute of Technology Tiruchirappalli, India

Obstacle Avoidance youtu.be/h_9SjreUzvU
Mapping youtu.be/z217gSfaEqg
Hardware youtu.be/rt1YuzRq05M

Abstract—Trident is a small differential-drive robot that I designed and built for a competition task: follow a line, then carry coloured coins to the bins of their colours. This paper reports its mechanical design and a simulation in which the whole task runs autonomously. The gripper does not lift: two parabolic jaws on a pair of equal involute gears, driven by one servo, cage a coin on the floor and the robot slides it, and a sliding condition for the jaw profile is derived as a design rule. The mass properties of the CAD model show that the centre of mass lies inside the support triangle but that the wheels carry only 18 % of the weight, which limits the acceleration to 1.8 m/s^2 and leaves a tipping margin of a factor of two. The depth camera cannot see the last 0.15 m of floor, so an eight-channel infrared array guides the end of every approach, on floor markings dimensioned from its pitch. The robot is modelled in Gazebo from the CAD export, with RTAB-Map and Nav2 for mapping and navigation and a line follower whose damping is shown to be set by the distance of the array ahead of the axle. In twelve simulated missions with four obstacle layouts the robot delivered all 36 coins, in 300 s per mission on average, scored by a referee on ground truth. The line was followed within 6 mm , the pose in the map was accurate to 10 mm RMS, and coins were placed within 21 mm of the centres of their squares. All measurements were taken in simulation.

Index Terms—mobile robot design, differential drive, caging gripper, line following, visual SLAM, navigation, Gazebo

I. INTRODUCTION

Trident is a small differential-drive robot that I designed and built for a task of the WorldSkills competition: follow a black line, then carry plastic coins, cylinders about 40 mm across and 15 mm high, each to the bin of its colour. A first version was a test rig for sensors and for ROS 2. The second, the subject of this paper, was designed in SolidWorks, printed, and completed with its electronics in fifteen days (Fig. 1). It won the district round. In the state round its infrared array stopped working and the task was not completed.

A robot of this size has little room for mechanism, and one built in two weeks has little time for trial runs. The work reported here therefore had two aims: a chassis, drive and gripper that move a coin of known size with one servo and no sensing in the jaws, and a simulation, built from the same CAD model, in which the whole mission can be run, scored against ground truth and repeated.

The contributions are:

- the mechanical design of the robot and of a gripper that cages a coin between two geared jaws and slides it along the floor (Section III);
- an analysis of the design from the mass properties of the CAD model, which gives the wheel loads and the traction and tipping limits (Section III-B);
- a Gazebo model of the robot with an emulated infrared array, and an arena whose floor markings are dimensioned from the pitch of that array (Sections IV and V);
- an autonomy stack in which a line follower takes over from the navigation where the camera can no longer see the floor (Section VI);
- measurements of the simulated mission over repeated runs (Section VII).

I am the sole author of the design, the hardware and the software.

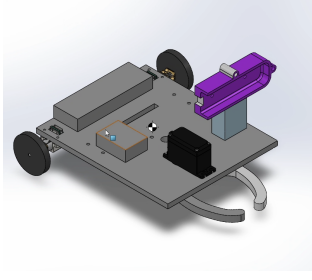
II. RELATED WORK

The kinematics and odometry of the differential drive are textbook material [1], and its frames in ROS are fixed by REP 105 [2]. A line is commonly followed with a proportional and derivative law [3] on the offset measured by a row of reflectance sensors; the distance of the row ahead of the axle acts as the look-ahead of pure pursuit [4]. A gripper that surrounds an object without clamping it is a cage [5], and the sliding of a pushed object along its pusher was analysed by Mason [6]. The simulation uses Gazebo [7] with ROS 2 [8]. For autonomy, RTAB-Map [9] provides visual SLAM on the streams of a RealSense depth camera [10], `robot_localization` [11] the odometry filter, and Nav2 [12] layered costmaps [13], a Dijkstra [14] planner and the Regulated Pure Pursuit controller [15], sequenced by a behaviour tree [16]. None of these components is new. The contribution is the design of a robot around a task and the way its limits decide how the components are combined.

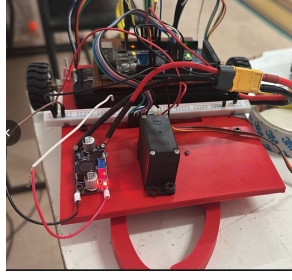
III. MECHANICAL DESIGN

A. Layout

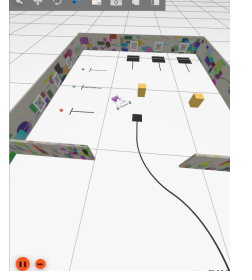
The chassis is one printed plate of $180 \times 150 \times 6 \text{ mm}$. Two N20 gearmotors under its rear edge drive wheels of 22 mm radius, 195.5 mm apart, and a ball caster 85 mm ahead of the axle is the third support. N20 motors were chosen because



(a) CAD model



(b) Printed robot during wiring



(c) Simulated robot in the arena

Fig. 1. Trident: two driven wheels at the rear, a ball caster under the plate and two geared jaws at the front. The holder at the side of the plate carries the depth camera.

motors of higher torque do not fit a robot of this size. The servo of the gripper stands on the plate near its front edge, and the jaws turn in a horizontal plane below the plate. An array of eight infrared reflectance channels at a pitch of 15 mm hangs under the plate, 48 mm ahead of the axle and 7.4 mm above the floor. An Intel RealSense D435 depth camera sits in a holder 85 mm above the floor and 54 mm to the left of the centre line. All parts are printed in PLA; the chassis went through more than four printed iterations to settle the tolerances for the bolts and nuts in use.

B. Mass properties

The design required the centre of mass to lie inside the support triangle of the wheels and the caster. The CAD export gives the mass and centre of mass of every part, so this can be checked and the wheel loads calculated. The model has a mass of 0.64 kg, and its centre of mass lies $x_c = 69.3$ mm ahead of the axle and 7.8 mm to the left (Fig. 2). With the caster at $x_k = 85.0$ mm and $W = 6.28$ N, the static loads on the caster and on the two wheels together are

$$N_k = W \frac{x_c}{x_k} = 5.12 \text{ N}, \quad N_w = W \left(1 - \frac{x_c}{x_k}\right) = 1.16 \text{ N}. \quad (1)$$

The centre of mass is inside the triangle, as required, and lies closer to the caster than to the axle: the wheels carry 18 % of the weight. The camera and its holder, 131 g at the front of the plate, are the main cause. Three consequences follow.

Traction. With a friction coefficient μ the wheels can accelerate or brake the robot at no more than

$$a_{\max} = \mu g \left(1 - \frac{x_c}{x_k}\right) = 1.8 \mu \text{ m/s}^2. \quad (2)$$

In the simulation, a robot without a braking limit skidded and turned on every stop command; the drive is now limited to 1.0 m/s^2 .

Drive sizing. On the motor data assumed in the model (300 rpm, 0.05 N m at the wheel, both placeholders) the motors can push with 4.5 N at stall, four times what the wheels can transmit. The drive is limited by traction and not by torque.

Tipping. The triangle is narrow at the caster. The centre of mass is 6.7 mm from its left edge and 42 mm above the floor, so a braking deceleration above 2.1 m/s^2 tips the robot over

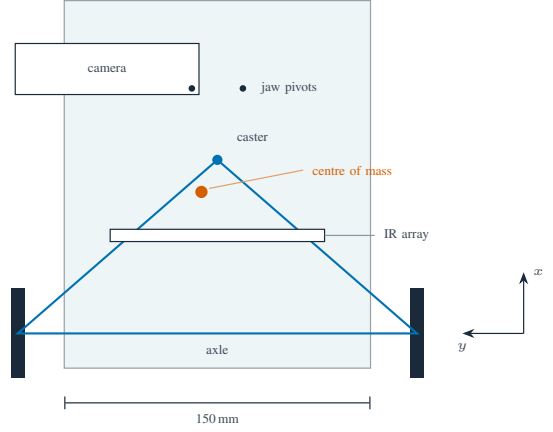


Fig. 2. Support triangle of the wheels and the caster from above, to scale. The centre of mass of the model is 16 mm behind the caster and 6.7 mm from the left edge.

that edge. The margin over the allowed 1.0 m/s^2 is a factor of two.

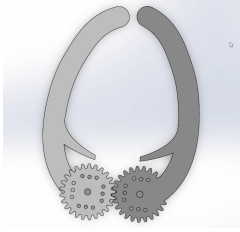
Moving the caster forward and the camera to the centre line would correct all three. The masses are CAD values and the printed robot has not been weighed on its supports, so the real centre of mass may lie further back.

C. Gripper

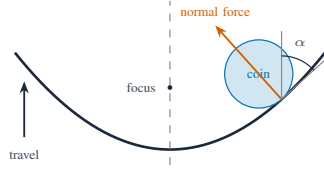
The robot is light, and a coin held out in front of the caster could tip it forward. The gripper therefore does not lift. Two jaws close around the coin on the floor and the robot slides it to its destination; it is enough that the coin cannot leave the space between the jaws (Fig. 3). This removes the lift axis and the need to know the mass of the coin.

The jaws have a parabolic profile, so that a coin which meets a jaw anywhere along its length is led inward. For a profile $y^2 = 4fx$ that advances along its axis onto a coin at a lateral offset y , the surface at the contact is inclined at $\tan \alpha = 2f/y$ to the direction of travel. The coin slides along the jaw as long as the direction of travel lies outside the friction cone about the surface normal, $90^\circ - \alpha > \arctan \mu$, that is,

$$|y| > 2\mu f. \quad (3)$$



(a) Jaws and gear pair



(b) Coin against a parabolic jaw

Fig. 3. The gripper. A coin is pushed along the normal of the profile, which has an inward component everywhere except at the vertex.

Centring therefore works best at the tips of the jaws, where a misaligned approach makes first contact, and improves with a short focal length and a slippery jaw. The focal length of the printed jaws was not measured, and (3) is a design rule and not a verified figure.

Each jaw carries a ridge near its pivot. When the jaws open, the ridges push the coin out, so that it is not drawn back with the robot. One MG995 servo turns one jaw directly, and a pair of involute gears [17] with equal numbers of teeth, on pivots 25 mm apart, turns the other by the same angle in the opposite sense. The stall torque of 0.83 N m gives 9 N at the tip of a jaw 90 mm long, against the 0.08 N that slides a coin of 23 g.

D. What the camera cannot see

The depth image has a vertical field of view of 58° . From a height of 85 mm its lowest ray reaches the floor at $85 \text{ mm} / \tan 29^\circ \approx 0.15 \text{ m}$ ahead of the lens, and depth is not reported below 0.2 m. A caged coin lies about 4 cm ahead of the lens. The camera therefore never sees the coin that the robot is about to pick, nor the last 15 cm of floor. This is the most important constraint that the mechanical design places on the software.

IV. SIMULATION MODEL

The robot is simulated in Gazebo Fortress with ROS 2 Humble, in an unprivileged Docker container. The robot description is maintained by hand on top of the URDF export of the CAD assembly. The export needed corrections before it was usable: a floor frame after REP 105, wheel axes turned so that positive rotation drives forward, a right wheel that the export had placed 0.5 mm off the axle, gripper joint limits that were all zero, the body of the camera, which was missing from the assembly, and a mass for the infrared array. The chassis mesh of 277 864 triangles is replaced for contact by boxes, cylinders and a sphere; the jaws keep their meshes, since their shape is what the gripper depends on. Gazebo Fortress ignores the mimic relation of a URDF, so the gear pair is reproduced by a node that sends the commanded angle to one jaw and its negative to the other.

The depth camera is modelled with the field of view of the D435, $87^\circ \times 58^\circ$, at 848×480 pixels. Gazebo has no reflectance sensor, so each infrared channel is a camera of 8×8 pixels that looks down on a spot about 8 mm across. A node averages each image into a voltage between 0 and 5 V

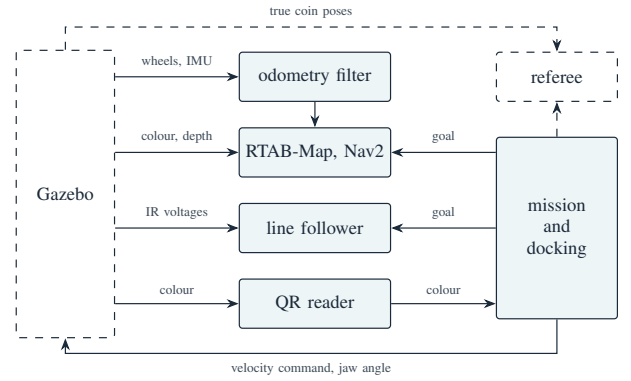


Fig. 4. Software architecture. Dashed boxes exist only in simulation.

and adds Gaussian noise of 0.05 V. A channel over the edge of a line returns an intermediate voltage, as the real sensor does. The arena is lit evenly and without shadows, so that the readings do not depend on heading. Motors and servo are ideal within their limits, and the floor is flat.

V. ARENA

The arena models the structure of the task and does not reproduce the competition field (Fig. 5). It measures $3.6 \times 2.4 \text{ m}$. A line leads from the start through an S-bend and a doorway to a start square in a room with three pick bays on one wall and three drop zones on another. A QR code behind each station names a colour: of the coin at a bay, of the coins accepted at a drop zone. The assignment is a launch argument, as is one of three layouts of obstacle boxes.

The floor markings follow from the pitch $p = 15 \text{ mm}$ and the span of 105 mm of the infrared array. Lines are 12 mm wide, less than one pitch, so that a line darkens at most two neighbouring channels, each partially, and its position can be interpolated. Markers (the start square and the crossbars of the bays) are 120 mm wide, more than the span, so that all eight channels go dark together, which nothing else on the floor can cause. A pick bay is a T: a stem of 0.30 m leads to a crossbar, and the coin lies 0.13 m beyond it, the distance from the array to the space between the open jaws. A drop zone is a stem into a square of 250 mm. The walls carry generated textures, because plain walls give a visual SLAM system nothing to track.

VI. AUTONOMY

Odometry, mapping, localisation and navigation are existing packages configured for this robot; the line follower, the docking sequences, the mission state machine and the referee were written for it (Fig. 4).

A. Localisation and navigation

An extended Kalman filter fuses the forward speed of the wheels with the yaw rate of the gyroscope; the yaw rate of the wheels is left out because they slip when the robot turns on the spot. RTAB-Map builds the map once, on a survey of the empty arena with a full turn at each of seven stops, and

localises in it during a mission. Nav2 plans with NavFn on a costmap of 2 cm cells, replans once a second, and tracks the path at 0.15 m/s. Obstacles come from the depth image between 3 cm and 0.40 m above the floor. Because the camera looks forward only, the controller may not reverse and the back-up recovery is removed from the behaviour tree.

The arena shows each QR code twice, at a bay and at a drop zone. Seen from close by, a match of the features of the code alone made RTAB-Map place the robot in front of the other wall. A fix is therefore accepted only when its matched features are spread across the image; during development true views scored 0.004 to 0.014 on this measure and look-alikes below 0.0025, and the threshold is 0.005.

B. Line follower

Each voltage is converted to a darkness d_i between 0 and 1, with readings below 0.15 set to zero. The line offset is the weighted mean $e = p \sum_i d_i(i - 3.5) / \sum_i d_i$, and the robot steers with

$$\omega = \text{sat}_{\omega_{\max}}(k_p e + k_d \dot{e}), \quad (4)$$

with $k_p = 45 \text{ rad}/(\text{s m})$, $k_d = 2 \text{ rad}/\text{m}$ and $\omega_{\max} = 1.5 \text{ rad}/\text{s}$ at 20 Hz. Let the array be a distance L ahead of the axle, ψ the heading relative to the line and $y = -e$ the lateral position of the array. Then $\dot{y} = v\psi + L\omega$ and $\dot{\psi} = \omega$, and with the proportional part of (4),

$$\ddot{y} + Lk_p \dot{y} + vk_p y = 0, \quad (5)$$

with damping ratio $\zeta = \frac{1}{2}L\sqrt{k_p/v}$. The damping comes entirely from L : an array on the axle would oscillate under proportional control at any gain. With $L = 48 \text{ mm}$, ζ is 0.51 at the entry speed of 0.10 m/s and 0.72 at the docking speed of 0.05 m/s. A marker is reported when six or more channels are dark in two successive samples.

C. Docking and mission

Nav2 brings the robot to a station pose on the extension of a stem, about 0.15 m short of its foot. The line follower then drives until the array finds the stem, steers onto it and stops at the marker. To pick, the robot opens the jaws, follows the stem to the crossbar, reads the QR code, closes the jaws and backs out by the distance just driven. To drop, it follows the stem until the array enters the square, opens the jaws and backs out. The stem absorbs the error of the navigation, and backing out along a path just driven needs no sensing behind the robot. A QR code is trusted only at its station and after three identical decodes, since the decoder does not report where in the image a code lies.

The mission follows the entry line, reads the three drop zones, then picks and delivers the three coins. A station whose marker is not found is tried once more; a coin whose code cannot be read is left in its bay. A referee reads the true positions of the coins from the simulator and passes a run when every coin lies inside the square of its colour. Nothing on the robot may read the ground truth.

TABLE I
TWELVE SIMULATED MISSIONS AND 36 PICKS AND DROPS

Quantity	Mean or RMS	Range or largest
Coins delivered	36 of 36	
Mission time	300 s	293 to 307 s
Path length	18.1 m	17.5 to 18.7 m
Offset from entry line (RMS)	1.6 mm	6.0 mm
Map pose error (RMS)	9.9 mm	39 mm
Map heading error (RMS)	0.7°	
Odometry alone, error at end	16 mm	30 mm
Pick: lateral offset on crossbar (RMS)	2.7 mm	7.1 mm
Pick: heading error (RMS)	2.9°	7.1°
Drop: coin from square centre	9.0 mm	20.7 mm

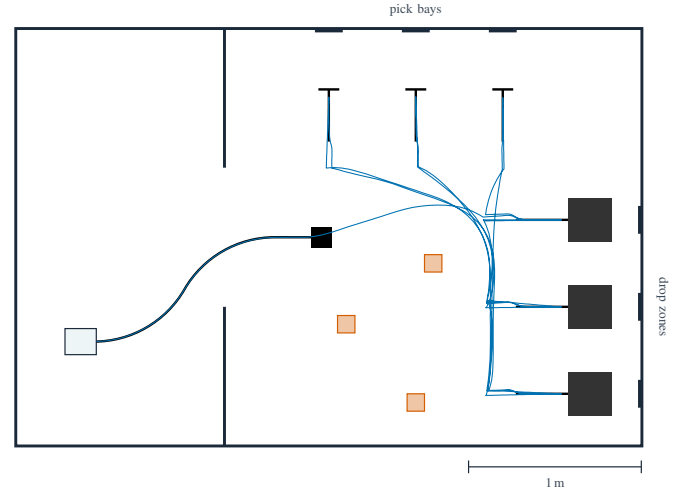


Fig. 5. Plan of the arena, to scale, with the true path of the robot in one mission with three obstacle boxes. The entry line leads from the start, at the left, to the start square in the room.

VII. RESULTS

The mission was run twelve times without a display, three runs for each obstacle layout (none and three presets of two or three boxes), with eleven different combinations of pick and drop colours, the same map and the same start pose. A recording node that only listens logged the true pose, the pose in the map and the odometry at 10 Hz.

All twelve runs passed, with 36 coins of 36 inside the squares of their colours (Table I). No station needed a second attempt and the navigation entered no recovery. A mission takes 300 s on average: 19 s on the entry line, 83 s for the survey of the drop zones and 66 s per coin. The obstacles cost at most 5 s. Fig. 5 shows the true path of one run.

Line following. The array stayed within 6.0 mm of the line, a tenth of the channel pitch in RMS terms, so the interpolation between partially covered channels works as intended. Run alone on the entry line (Fig. 6), the follower is insensitive to noise up to 0.2 V, four times the nominal level, and degrades above it, when noise crosses the floor threshold. The offset grows quickly with speed, from 1.1 mm RMS at 0.05 m/s to 17.6 mm at 0.30 m/s, in the direction that the fall of ζ from 0.72 to 0.29 predicts.

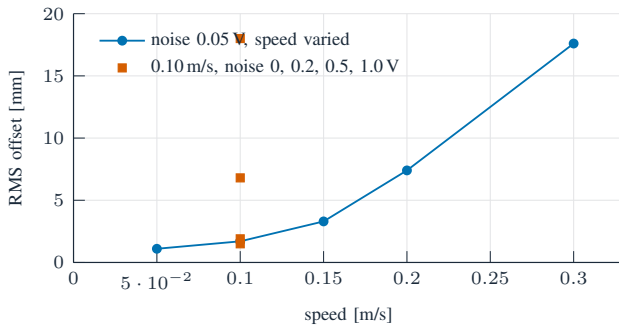


Fig. 6. Line follower alone on the entry line, one run per point: RMS offset of the array from the line.

Localisation. The pose in the map was within 9.9 mm RMS of the truth. The odometry filter alone, however, was never more than 30 mm off in any run: the simulated encoders are ideal and the floor is uniform. The runs show that the localisation is stable and does not jump; they do not show what it would contribute on a real floor.

Docking. When the jaws closed, the array was 2.7 mm RMS to the side of the stem and always on the crossbar. Heading is the least well controlled quantity, with 7.1° at worst, and all 36 coins were nevertheless caged, which is the tolerance that the shape of the jaws was meant to provide. Released coins lay 9.0 mm from the centre of a square that extends 125 mm from it.

VIII. LIMITATIONS

All measurements were taken in simulation, and none on the printed robot. The masses are CAD values and the motor data placeholders; the robot was not weighed. The simulated motors are ideal within their limits and the encoders have no quantisation, so the odometry is better than it would be on a floor. The arena was designed with the software: its markings, wall textures and shadow-free lighting are the conditions under which the approach works. The twelve runs share one map and one start pose, and each line-following condition was run once. The condition that rejects look-alike views is one number chosen for this arena. The planner treats the robot as a disc that does not enclose the jaws. The mission has no feedback from the jaws, and, like the robot that failed in the state round, it depends on a single line sensor with no fallback.

IX. CONCLUSION

The design decisions that matter most in Trident are the ones that removed something: a gripper without a lift axis or a sensor, and no second camera for the floor that the first cannot see. Two lessons recur. A model answers questions that were settled by eye during the build: the centre of mass lies inside the support triangle, as intended, but the CAD model puts 82 % of the weight on the caster, which the simulation showed as a skid before the calculation explained it. And the position of a sensor is a control parameter: the distance of the infrared array ahead of the axle sets the damping of the line follower.

The simulated mission is complete and repeatable. The steps that remain are on the hardware: to weigh the robot and move the caster and the camera, to replace the breadboard wiring, and to run the mission on the printed robot.

REFERENCES

- [1] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*, 2nd ed. Cambridge, MA: MIT Press, 2011.
- [2] W. Meeussen, “REP 105: Coordinate frames for mobile platforms,” ROS Enhancement Proposal, 2010, <https://www.ros.org/reps/rep-0105.html>.
- [3] K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ: Princeton University Press, 2008.
- [4] R. C. Coulter, “Implementation of the pure pursuit path tracking algorithm,” Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-92-01, 1992.
- [5] A. Rodriguez, M. T. Mason, and S. Ferry, “From caging to grasping,” *The International Journal of Robotics Research*, vol. 31, no. 7, pp. 886–900, 2012.
- [6] M. T. Mason, “Mechanics and planning of manipulator pushing operations,” *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 53–71, 1986.
- [7] N. Koenig and A. Howard, “Design and use paradigms for Gazebo, an open-source multi-robot simulator,” in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, 2004, pp. 2149–2154.
- [8] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, “Robot operating system 2: Design, architecture, and uses in the wild,” *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022.
- [9] M. Labbé and F. Michaud, “RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation,” *Journal of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.
- [10] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, “Intel RealSense stereoscopic depth cameras,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 1267–1276.
- [11] T. Moore and D. Stouch, “A generalized extended Kalman filter implementation for the Robot Operating System,” in *Intelligent Autonomous Systems 13*, ser. Advances in Intelligent Systems and Computing, vol. 302. Springer, 2016, pp. 335–348.
- [12] S. Macenski, F. Martín, R. White, and J. G. Clavero, “The Marathon 2: A navigation system,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 2718–2725.
- [13] D. V. Lu, D. Hershberger, and W. D. Smart, “Layered costmaps for context-sensitive navigation,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014, pp. 709–715.
- [14] E. W. Dijkstra, “A note on two problems in connexion with graphs,” *Numerische Mathematik*, vol. 1, pp. 269–271, 1959.
- [15] S. Macenski, S. Singh, F. Martín, and J. Ginés, “Regulated pure pursuit for robot path tracking,” *Autonomous Robots*, vol. 47, pp. 685–694, 2023.
- [16] M. Colledanchise and P. Ögren, *Behavior Trees in Robotics and AI: An Introduction*. Boca Raton, FL: CRC Press, 2018.
- [17] R. G. Budynas and J. K. Nisbett, *Shigley’s Mechanical Engineering Design*, 10th ed. New York: McGraw-Hill Education, 2015.