Lab 2: Robot Kinematics - Report

Matias Cinera
Computer Science Department, *University of South Florida*
Tampa, Florida, 33612, USA
cinera@usf.edu

I. Introduction

This document is the report of my 2nd lab from my Spring 2023 class CDA-6626 Autonomous Robots, as a graduate student. The individuals who are overviewing this project are our professor Dr. Alfredo Weitzenfeld (Professor in the department of Computer Science) and Mr. Chance Hamilton (Teaching Assistant & Graduate Student)

II. Waypoint Navigation

The first task required the robot to follow a set of waypoints, given the robot's initial position. This will be achieved by pre-computing the robot's movement based on the kinematic equations. For this lab, only 3 motions are needed to follow the path: Straight Motion, 90° Rotation in place, and Quarter-Circle Rotations..

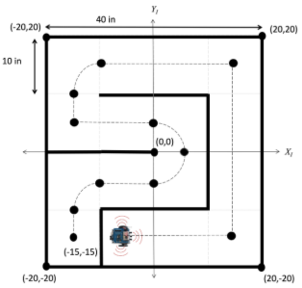

Figure 1: Path followed by the robot

A. Straight-Motion

Straight Motions happen when the robot's path is a straight line. To achieve this both the right & left motor rotational speed is the same. For this lab, I used the max-speed for both motors, and the distance can be derived with the waypoints. The only variable left to calculate is time.

$$t = d/v$$

Figure 2: Derived time when v and d are known

B. 90°in-place Rotation

To rotate the robot 90° $\mathbf{v_l}$ must be equal to -($\mathbf{v_r}$). Since there is no time limit for the task, I picked 0.4 ***in/s*** for the speed. The distance covered by this motion is equal to **R*(π/2)**. Since, the robot is rotating in place, **R** equals $\mathbf{d_{mid}}$. The only variable left to calculate is time.

$$S = \begin{cases} \theta d_{mid}, & v_l = -v_r \\ \theta R, & v_l \neq -v_r \end{cases}$$

$$t = S/v$$

Figure 3: Rotation in place equations

C. Quarter Circle Motion

Conceptually, this motion is the same as a 90° in-place rotation. The only difference is that $\mathbf{v_l}$ does not equal $\mathbf{-v_r}$. Thus, the equation for **S** changes. For this motion, I set the velocity for the dominant wheel (based on the direction of the run), and then calculated **ω, v**. Since all the waypoints were given, R equals 5 and a quarter circle motion means that **S = R*(π/2).**

$$\omega = \frac{v_l}{R + d_{mid}}$$

$$v_r = \omega \cdot (R - d_{mid})$$

$$S = \theta R$$

Figure 4: Quarter-Circle kinematic equations when $\mathbf{v_l > v_r}$

III. Task 2 – IMU & Robot Pose

A. IMU

The IMU (Inertial Measurement Unit) sensors combine multiple sensors (Accelerometer, a Gyro, and a Compass) to determine the robot's motions. For this lab, I used the IMU to determine the robot's orientation to update its current angle.

B. Updating Robots Pose (X, Y)

Updating the pose of the robot in straight and 90°in-place motions is trivial. The distance covered by straight motions is d = v*t. For in-place turns there is no change in x & y, thus only the IMU readings are needed to update the pose. However, to update the robot's pose in quarter-circle motions I had to use differential drive equations. Usually, to do this, first, you can find the angle and then derive S ***(S=Rθ).*** However, since this lab required updating the position every 32ms, the R-value calculated was inaccurate. Thus, instead of using R, I used S to determine the pose. To do this, I derived vl & vr from the encoders, then calculated **V**, and **S**. Finally, I used **S** the current robot's pose to determine the $\mathbf{x_c}$ and $\mathbf{y_c}$ values.

$$\begin{bmatrix} x_c \\ y_c \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 \\ S \\ \theta \end{bmatrix} + \begin{bmatrix} S \cdot \cos(\theta) \\ S \cdot \sin(\theta) \\ \theta \end{bmatrix} = \begin{bmatrix} x_c + S \cdot \cos(\theta) \\ y_c + S \cdot \sin(\theta) \\ \alpha + \theta \end{bmatrix}$$

Figure 5: Differential Drive Equations used to update the robot pose

IV. Conclusions

There are a few problems I encountered when dealing with the wheel encoders. Since the shortest time step in the simulation is 32ms, will not be very precise. Because of this the robot pose did not align perfectly with the waypoints. This means that by the end of each motion, there was a ±0.5 disparity in the x & y values of the robot's pose. Because of this discrepancy, the angle retrieved the IMU also differed by a small margin.

References

[1] Webots Reference Manual. Webots. (n.d.). Retrieved January 23, 2023, from https://www.cyberbotics.com/doc/guide/imu-sensors?version=develop