Lab 01: Trajectory Tracking

Roman Herrera
Harvey Mudd College Engineering Department
Harvey Mudd College
Claremont, CA
rherrera@g.hmc.edu

Lauren West
Harvey Mudd College Engineering Department
Harvey Mudd College
Claremont, CA
lwest@hmc.edu

Abstract—**In this lab, we created a trajectory tracker by building on a point tracking controller to make a robot closely follow a desired Dublin's path. The point tracker implements a simple proportional control law that calculates right and left wheel outputs to drive the robot to a specified location. The trajectory tracker then uses the point tracker to follow points along the desired path. We used a virtual environment to simulate a robot following our trajectory tracker to generate a path. We contrasted the path traversed in the simulation with the desired path and plotting the results. The trajectory tracker simulated a path for the robot that closely resembled the desired Dublin's path, having reasonable errors for x, y, and theta positions. The errors for x, y, and theta are 0.153m, 0.093m, and 0.044m, respectively.**

Keywords—point tracking, point controller, trajectory tracker, motion planning

I. INTRODUCTION

We sought in Lab 1 to create a controller that has the robot follow the desired trajectory. The robot's trajectory is obtained by using a point tracking controller that gets the robot from point a to point b. By creating this, we can now verify that our robot can closely follow trajectories given to it, such that later on, the robot can follow subsequent trajectories to reach a desired goal. Most of our work was derived from the lecture notes of E206, which provided structure and equations for the point-tracking control law that were implemented in our controller [1]. The lectures notes referenced also described to us how to implement a trajectory tracker, given a successful point-tracking controller [2].

II. METHOD

We assume that the forward kinematics of a robot can be described as the following linear differential equation with many dimensions:

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} \tag{1}$$

where A is a matrix such that $A \in R^{nxn}$ and x is a vector such that $\mathbf{x} \in R^{nxl}$[1]. The system is stable if A's eigen-values are less than zero.

According to [1], our robot's forward kinematics can be represented by:

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix} = \begin{bmatrix}\cos\theta & 0\\ \sin\theta & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ w\end{bmatrix} \tag{2}$$

Then, we used the coordinate transformation to get values for our distance to the goal (ρ), our angle to the goal (α), and the angle from the x direction to the goal angle (β).

$$\rho = \sqrt{\Delta x^2 + \Delta y^2} \tag{3}$$

$$\alpha = -\theta + \operatorname{atan2}(\Delta y,\ \Delta x) \tag{4}$$

$$\beta = -\theta - \alpha \tag{5}$$

Allowing us to represent (2) as:

$$\begin{bmatrix}\dot{\rho}\\ \dot{\alpha}\\ \dot{\beta}\end{bmatrix} = A\begin{bmatrix}\rho\\ \alpha\\ \beta\end{bmatrix} \tag{6}$$

From this, we generated gain constants to force the error to zero, trying the control laws:

$$v = k_\rho \rho \qquad w = k_\alpha + k_\beta \beta \tag{7}$$

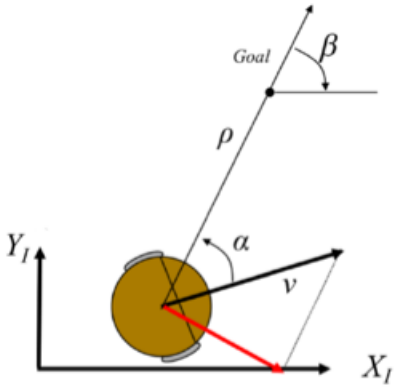
To determine the stability of (7), we took the determinant of A and solved for its eigenvalues. We found that the system will be stable if $k_\rho > 0,\ k_\beta < 0,$

and $k_\alpha - k_\rho > 0$ [1]. For backwards motion, $\alpha \in [\frac{-\pi}{2}, \frac{\pi}{2}]$, we use:

$$\rho = \sqrt{\Delta x^2 + \Delta y^2} \quad (8)$$

$$\alpha = -\theta + \text{atan2}(-\Delta y, -\Delta x) \quad (9)$$

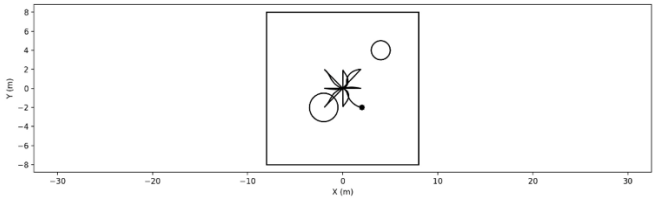
$$\beta = -\theta - \alpha \quad (10)$$



This illustration from E206 Lecture 2A visually depicts the variables ρ, v, α, β used in our controller [1].

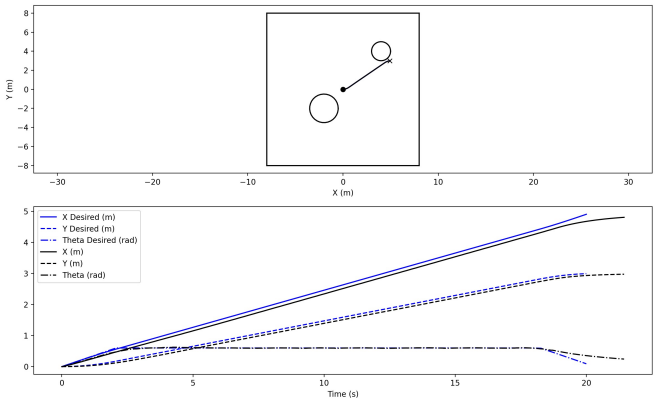
III. Experiments and Results

In the simulations, we observed the robot traveling from the origin to the desired point, asymptote towards its goal location. We were able to reset the environment to go back to the origin after getting within 0.1m of the goal location. We represent each point in a path as [x, y, θ]. Below is a plot of these back-to-back simulations from [0,0,0] to these points: [2, 0, 0], [2, 2, 0], [0, 2, pi/2], [-2, 2, 0], [-2, 0, 0], [-2, -2, 0], [0, -2, -pi/2], and [2, -2, 0].



XY Plot of Point Tracker generated using [4]

From the plot, we can see that the robot goes to the desired points, but perhaps not with the most optimal route. The slight deviation was deemed negligible, so we implemented the point tracker to create a trajectory-tracker. The difference between a trajectory and a path is that each point also has a timestamp. As such, each point in a trajectory is represented as [time, x, y, θ]. These are the results from our trajectory tracker from [0, 0, 0, 0] to [20, 5, 0, 0].



XY Plot and XYθ vs t Plot for the Trajectory Tracker. Generated using [4].

The robot seems to closely follow the desired trajectory. With an x-error of 0.153 m, a y-error of 0.0925 m and a theta-error of 0.049 rad, our robot does not stray too far off of the desired trajectory. We also noticed that the robot asymptotically approaches the goal, as is usual of proportional control laws.

IV. Conclusions

Based on our results, we can conclude that the trajectory tracker we implemented is satisfactory at correctly guiding the robot along the desired trajectory. We had relatively small average errors and based on our graphs, we can see that the robot closely follows the desired trajectory. In the future, we plan on having the robot determine its own trajectories for it to follow in response to changes or obstacles in its environment.

Acknowledgment

Ginger Schmidt and C. Clark, thank you for your guidance on the point tracking controller and for helping us revise our work.

References

[1] C. Clark, E206 Class Lecture, "Lecture 02A- Point Tracking." Harvey Mudd College, Claremont, CA. Feb., 2021.

[2] C. Clark, E206 Class Lecture, "Lecture 02B- Trajectory Tracking." Harvey Mudd College, Claremont, CA. Feb., 2021.

[3] C. Clark, E206 Class Site, "Lecture 02A- Motion Models." Harvey Mudd College, Claremont, CA. Feb., 2021.

[4] C. Clark, Starter Code, "Lab 00: Onboarding". Harvey Mudd College, Claremont, CA. Feb., 2021.