Design of Path Following Controller for Mobile Robot

Raj Patel

November 23, 2020

Contents

1 Introduction

The line following controller for a mobile robot iteratively finds a line that both is perpendicular to the desired linear path and intersects the position of the robot. Given by geometric constraints, the shortest distance between a line and a point is found on another line that satisfies both said properties. Extrapolating this geometric principle to a nonlinear path, there exists multiple such lines that both are perpendicular to the nonlinear path and intersect the position of the robot. To solve the path following problem for nonlinear paths, the line following controller must be adjusted to (1) take into account the changing direction of the path and (2) minimize the Euclidean distance function in place of solving a linear dot product.

In this paper, a path following controller for a mobile robot is derived and demonstrated. In Section 2, a detailed derivation of the path following controller is shown. Theoretically, the path following controller would work for any path defined by a continuous function. In Section 3, the results of the path following controller for a given path and several initial conditions are discussed.

2 Technical Approach

2.1 Unicycle Model

In this demonstration, a planar mobile robot with unicycle model dynamics will be used. Measurements will be taken with respect to the inertial frame. The unicycle model is defined as

$$\dot{x} = v\cos(\theta) \ , \tag{2.1}$$

$$\dot{y} = v\sin(\theta) \ , \tag{2.2}$$

$$\dot{\theta} = \omega \ , \tag{2.3}$$

where (x, y) is the position of the robot, θ is the orientation of the robot, v is the linear velocity of the robot, and ω is the angular velocity of the robot.

2.2 Open-Loop Dynamics

To follow a given path, the distance between the robot and the path must be minimized to the desired distance, and the angular difference between the orientation of the robot and the direction of the path must be minimized to 0. For this controller design, the robot will assume a constant speed. The open-loop dynamics are defined as

$$v = v_f \tag{2.4}$$

$$\dot{\rho} = -v\cos(\frac{\pi}{2} - \phi) = -v\sin(\phi)\ , \tag{2.5}$$

$$\phi = \theta_o - \theta\ , \tag{2.6}$$

$$\dot{\phi} = \dot{\theta}_o - \dot{\theta} = \dot{\theta}_o - \omega = -vu\ , \tag{2.7}$$

$$\omega = \dot{\theta}_o + vu\ , \tag{2.8}$$

where v is the linear velocity of the robot, ρ is the shortest distance between the robot and the path, θ is the orientation of the robot, θ_o is the direction of the path, ω is the angular velocity of the robot, ϕ is the angular difference between θ and θ_o, and u is the control input.

2.3 General Path Constraints

A derivation of the closed-loop forms of parameters is necessary for real-time operation of the controller:

$$y = f(x)\ , \tag{2.9}$$

$$f'(x) = \frac{d}{dx}f(x)\ , \tag{2.10}$$

$$f''(x) = \frac{d^2}{dx^2}f(x)\ , \tag{2.11}$$

$$\overrightarrow{p}(x) = \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x \\ f(x) \end{bmatrix}\ , \tag{2.12}$$

$$q_R(t) = \begin{bmatrix} x_R \\ y_R \end{bmatrix}\ , \tag{2.13}$$

$$d(x) = \|q_R(t) - \overrightarrow{p}(x)\| = \sqrt{(x_R - x)^2 + (y_R - f(x))^2} \tag{2.14}$$

$$x_{d_{\text{extrema}}}(t) = \{x|\frac{d}{dx}d(x)^2 = 0, x \in \mathbb{R}\}\ , \tag{2.15}$$

$$\rho(t) = \min(\{d(x_{d_{\text{extrema}}}(t))\})\ , \tag{2.16}$$

$$x_{d_{\min}}(t) = x|\rho = d(x)\ , \tag{2.17}$$

$$\overrightarrow{v}(x) = \frac{d}{dx}\overrightarrow{p}(x) = \begin{bmatrix} 1 \\ f'(x) \end{bmatrix} , \tag{2.18}$$

$$\hat{T}(x) = \frac{\overrightarrow{v}(x)}{\|\overrightarrow{v}(x)\|} = \begin{bmatrix} \frac{1}{\sqrt{1+f'(x)^2}} \\ \frac{f'(x)}{\sqrt{1+f'(x)^2}} \end{bmatrix} , \tag{2.19}$$

$$\theta_o(t) = \arctan(\frac{T_y(x_{d_{\min}}(t))}{T_x(x_{d_{\min}}(t))}) = \arctan(\dot{f}(x_{d_{\min}}(t))) , \tag{2.20}$$

$$\dot{\theta}_o(t) = \frac{d}{dt}\theta_o(t) = \frac{\ddot{f}(x_{d_{\min}}(t))}{1 + \dot{f}(x_{d_{\min}}(t))^2} , \tag{2.21}$$

where $y, f(x), \overrightarrow{p}(x)$ are the (same) desired path, t is the current iteration, $q_R(t)$ is the position of the robot at iteration t, $d(x)$ is the Euclidean distance between the robot and the path, $x_{d_{\text{extrema}}}(t)$ is the set of all local extrema of distances between the robot and the path, $\rho(t)$ is the shortest distance given by the minima in set $x_{d_{\text{extrema}}}(t)$, $x_{d_{\min}}(t)$ is the x-coordinate of the point on the path that results in ρ, $\hat{T}(x)$ is the unit tangential vector of the path, and $\theta_o(t)$ is the direction of the curve at the point of shortest distance to the robot at iteration t.

2.4 Closed-Loop Dynamics

Proportional feedback control is used to iteratively adjust ρ and ϕ. The controller is designed to incorporate knowledge of the desired path, the direction of the path, and the velocity and acceleration of the path. The closed-loop dynamics are given by

$$u = -k_\rho(\rho - \rho_o) - k_\phi(\theta - \theta_o) , \tag{2.22}$$

$$v = v_o , \tag{2.23}$$

$$\dot{\rho} = -v_o \sin(\phi) , \tag{2.24}$$

$$\dot{\phi} = -v_o(-k_\rho(\rho - \rho_o) - k_\phi(\theta - \theta_o)) , \tag{2.25}$$

$$\omega = \frac{\ddot{f}(x_{d_{\min}})}{1 + \dot{f}(x_{d_{\min}})^2} + v_o(-k_\rho(\rho - \rho_o) - k_\phi(\theta - \theta_o)) , \tag{2.26}$$

where u is the control input, ρ is the shortest distance between the robot and the path, v is the linear velocity of the robot, v_o is the desired constant speed of the robot, ρ_o is the desired distance between the robot and the path, k_ρ is the proportional gain for ρ, θ is the orientation of the robot, θ_o is the direction of the path, ϕ is the angular difference between θ and θ_o, k_ϕ is the proportional gain for ϕ, ω is the angular velocity of the robot, $x_{d_{\min}}$ is the x-coordinate of the point on the path that results in ρ, and $f(x)$ is the desired path.

2.5 Stability

Stability can be analyzed through the linearization of $\dot{\rho}$ and $\dot{\phi}$:

$$\lim_{x \to 0} \cos(x) \approx 1 \ , \tag{2.27}$$

$$\lim_{x \to 0} \sin(x) \approx x \ , \tag{2.28}$$

$$\dot{\rho} \approx -v_o \phi \ , \tag{2.29}$$

$$\dot{\phi} = -v_o(-k_\rho(\rho - \rho_o) - k_\phi(-\phi)) \ , \tag{2.30}$$

$$\begin{bmatrix} \dot{\rho} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & -v_o \\ v_o k_\rho & -v_o k_\phi \end{bmatrix} \begin{bmatrix} \rho - \rho_o \\ \phi \end{bmatrix} \ , \tag{2.31}$$

$$A = \begin{bmatrix} 0 & -v_o \\ v_o k_\rho & -v_o k_\phi \end{bmatrix} \ , \tag{2.32}$$

$$cp_A = \det(\lambda I_2 - A) = \lambda^2 + k_\phi v_o \lambda + k_\rho v_o^2 \ , \tag{2.33}$$

$$\lambda_A = \{\text{roots } cp_A\} = v_o \frac{-k_\phi \pm \sqrt{k_\phi^2 - 4k_\rho}}{2} \ , \tag{2.34}$$

For the system poles to be on left-half of complex plane:

$$k_\phi > 0 \ , \tag{2.35}$$

$$k_\rho > 0 \ , \tag{2.36}$$

Further, for the system poles to be real:

$$k_\phi^2 > 4k_\rho \ , \tag{2.37}$$

where ρ is the shortest distance between the robot and the path, v_o is the desired constant speed of the robot, ϕ is the angular difference between θ and θ_o, k_ρ is the proportional gain for ρ, ρ_o is the desired distance between the robot and the path, k_ϕ is the proportional gain for ϕ, A is a state space matrix of the linearized system, cp_A is the characteristic polynomial of the linearized system, and λ_A is the set of eigenvalues of the linearized system.

3 Results

The parameters had been tuned to optimize results for a specific path:

$$v_o = 1.2 \ , \tag{3.1}$$

$$\text{acc} = 1 \ , \tag{3.2}$$

$$\text{turn\_rt\_lim} = 2.3 \ , \tag{3.3}$$

$$\text{frame\_shift} = 0.2 \ , \tag{3.4}$$

$$k_\rho = 0.7 \ , \tag{3.5}$$

$$k_\phi = \sqrt{4k_\rho} + 0.2 \ , \tag{3.6}$$

$$\rho_o = 0.2 \ , \tag{3.7}$$

$$f(x) = 0.01x^5 - 0.27x^3 + 0.14x^2 + 1.2x - 3 \ , \tag{3.8}$$

where v_o is the desired constant speed of the robot, acc is the acceleration of the robot, turn_rt_lim is the limit of the wheel turning rate of the robot, frame_shift is the shift difference of the robot's coordinate frame based on the shifted reference point model, k_ρ is the proportional gain for ρ, k_ϕ is the proportional gain for ϕ, ρ_o is the desired distance between the robot and the path, and $f(x)$ is the desired path.

These parameters were used consistently for all of the simulations discussed later. ρ_o is consistently nonzero to simulate convergence to a desired distance from an obstacle or wall. The robot was able to converge to the path using the path following controller for many initial conditions. The only parameter changed between each simulation run was the initial condition of the robot.

The initial condition of the robot is given in the form

$$x(0) = \begin{bmatrix} x(0) \\ y(0) \\ \theta(0) \end{bmatrix} \tag{3.9}$$

3.1 Controller Success

In this subsection, the initial conditions discussed are

$$x_1 = \begin{bmatrix} -4 \\ 2 \\ \frac{\pi}{4} \end{bmatrix} \tag{3.10}$$

$$x_2 = \begin{bmatrix} 0 \\ 2 \\ \frac{-\pi}{4} \end{bmatrix} \tag{3.11}$$

Figure 3.1 and Figure 3.2 both show that the robot converges to the desired path from its initial condition. Near sharper turns, using the path following controller, the robot is capable of adjusting and re-converging to the desired path.

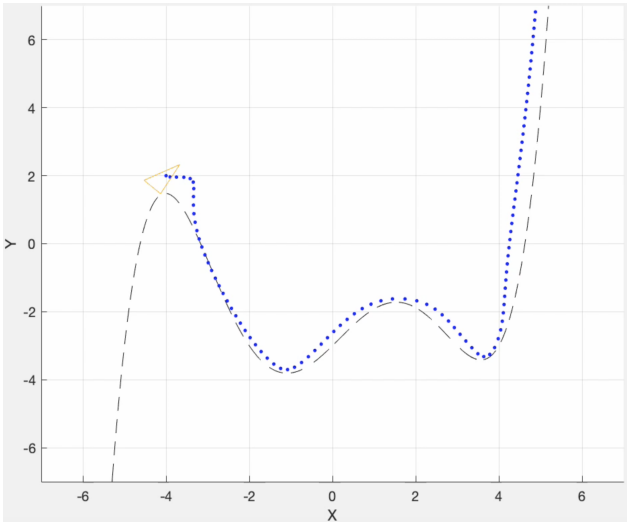

Figure 3.1: Simulation Plot of Desired Path and Robot Path with Initial Condition x_1.

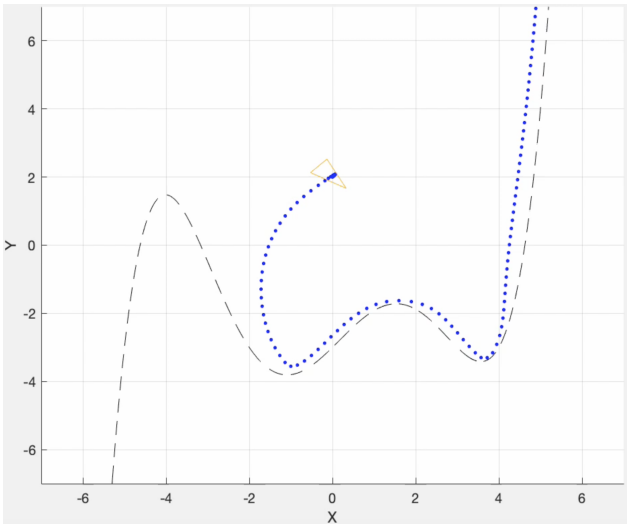

Figure 3.2: Simulation Plot of Desired Path and Robot Path with Initial Condition x_2.

3.2 Controller Semi-Failure

In this subsection, the initial conditions discussed are

$$x_3 = \begin{bmatrix} -6 \\ 4 \\ \frac{\pi}{3} \end{bmatrix} \tag{3.12}$$

$$x_4 = \begin{bmatrix} -4 \\ 2 \\ 0 \end{bmatrix} \tag{3.13}$$

Around regions of higher curvature of the desired path, the robot is more susceptible to getting to close to the desired path, or even touching or shortly crossing over the desired path. Since there is no knowledge of future positions on the desired path in the path following controller, the robot has no method of preemptively adjusting its motion to account for regions of high curvature, other than the knowledge of acceleration of the desired path in $\dot{\theta}_o$. This phenomenon can be seen in near the first local extrema from the left in both Figure 3.3 and Figure 3.4.

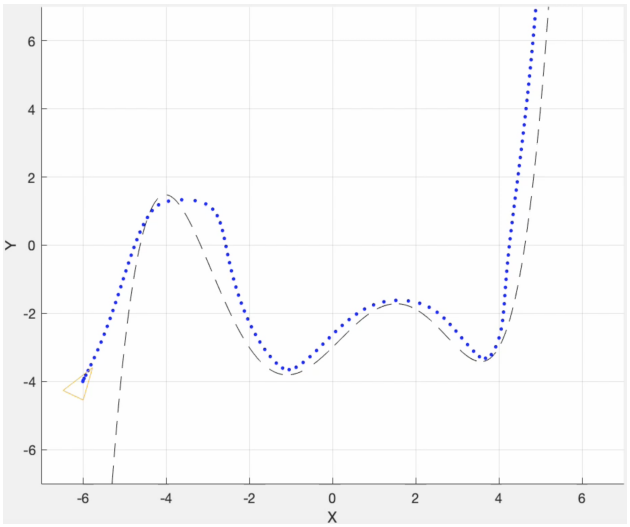

Figure 3.3: Simulation Plot of Desired Path and Robot Path with Initial Condition x_3.

The robot in Figure 3.4 starts near the first local extrema from the left, and crosses over the desired path. Figure 3.4 is a simulation with the almost initial condition as 3.1; the only difference is $\theta(0)$. The difference in the robot behavior between initial conditions x_1 and x_4 shows the importance of having a reasonable, with regards to physical constraints, initial condition. Eventually, the robot converges back to the desired path.

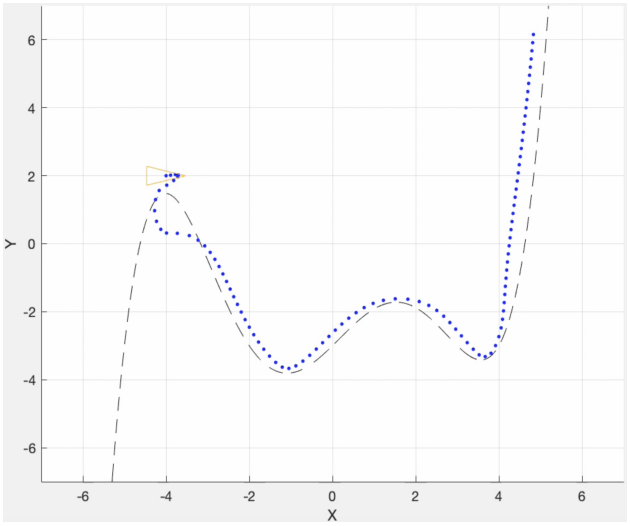

Figure 3.4: Simulation Plot of Desired Path and Robot Path with Initial Condition x_4.

3.3 Controller Failure

In this subsection, the initial conditions discussed are

$$x_5 = \begin{bmatrix} 0 \\ 4 \\ \pi \end{bmatrix} \tag{3.14}$$

$$x_6 = \begin{bmatrix} -4 \\ -5 \\ \frac{\pi}{3} \end{bmatrix} \tag{3.15}$$

Figure 3.5 is an example of when the path following controller fails the robot. Due to the nature of distance minimization, there are initial conditions where the robot cannot converge to the desired path because the robot enters a loop where the minimum distance oscillates, leading to movement over the same set of positions.

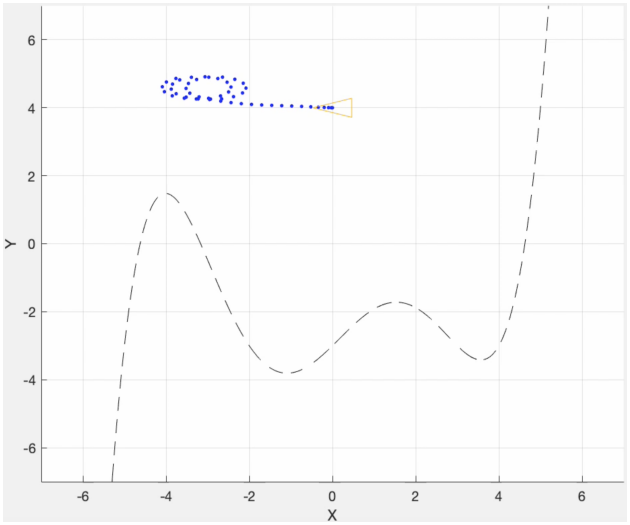

Figure 3.5: Simulation Plot of Desired Path and Robot Path with Initial Condition x_5.

Figure 3.6 is an example that shows the limitations of the path following controller; the path following controller defines the desired path to be on the right side of the robot. The robot in Figure 3.6 attempts to bring the desired path to the right side of the robot, and then converges to the desired path.

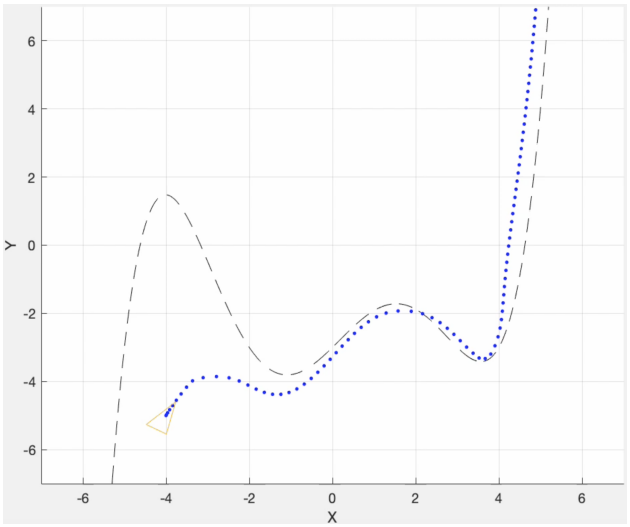

Figure 3.6: Simulation Plot of Desired Path and Robot Path with Initial Condition x_6.