Lab Report: Simulation of Differential Drive Mobile Robot Trajectory Planning

Matthew Raghunandan

Objective

The aim of this lab was to simulate a differential drive mobile robot using matlab. The robot had a goal position and our goal was to generate a smooth trajectory to the goal. The robot was modeled using a set of ordinary differential equations (ODEs) and the control law was based on three error components.

Methodology

Differential Equations

We model the robot's dynamics using the following set of ordinary differential equations (ODEs):

$$\dot{x} = v\cos(\theta)$$
$$\dot{y} = v\sin(\theta)$$
$$\dot{\theta} = \omega$$

Where v and ω are the linear and angular velocities, respectively, and θ is the robot's heading.

Control

The control law for the robot is based on three error components:

1. ρ - Distance to the goal
2. α - Angle to the goal from the current heading
3. β - Angle to the goal

The control law is formulated as:

$$v = k_p * \rho$$
$$\omega = k_\alpha * \alpha + k_\beta * \beta$$

Where k_p, k_α, and k_β are the proportional gains for the three error components.

Parameters Used

- $r = 50$: Initial radius of the circle
- $\text{angle} = -\frac{3\pi}{4}$: Initial angle in radians
- $k_p = 3$: Proportional Gain for positional error
- $k\_\alpha = 8$: Proportional Gain for angular error α
- $k\_\beta = -1.5$: Proportional Gain for angular error β
- $t\_\text{span} = [0, 10]$: Time span of the simulation

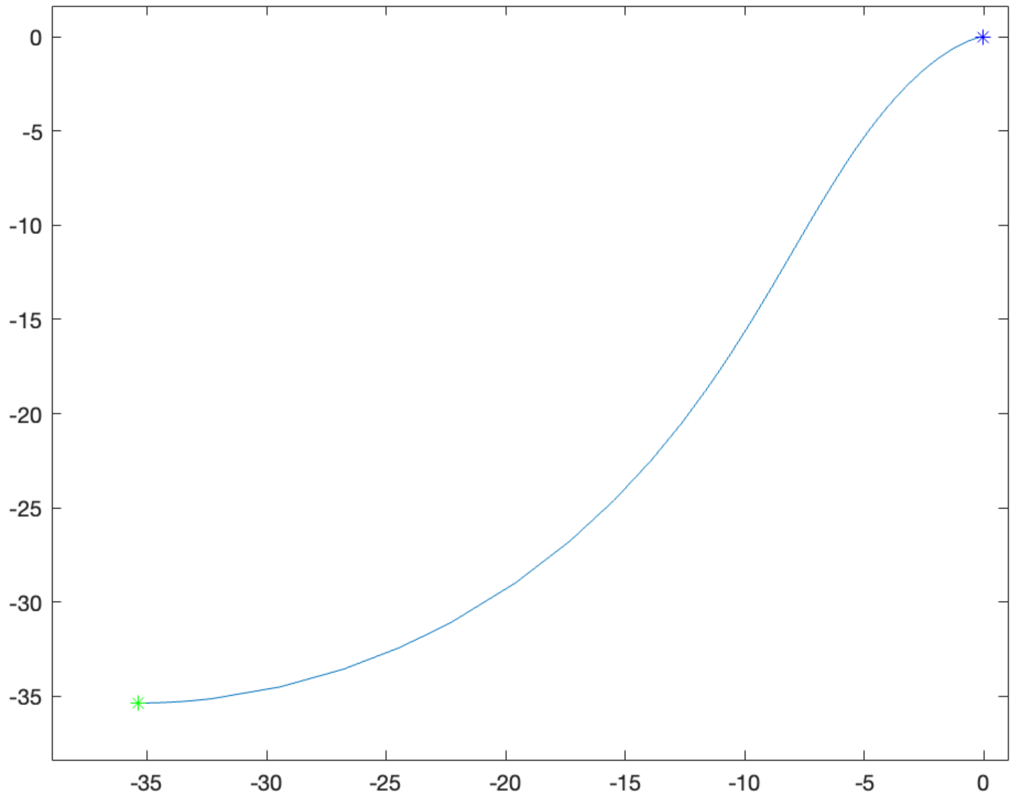
Implementation

The Matlab code starts by setting the initial position of the robot based on r and angle. The code uses Matlab's ode45 function to solve the system of ODEs over the time span specified.

The code then plots the trajectory.

Simulation Results

Trajectory Plot



Observations

- The robot starts from the initial point and moves toward the center.
- The trajectory is smooth, which indicates that the control strategy is effective.

Conclusion

The experiment successfully simulates the path of a differential drive robot moving from an initial position on a circle to a goal position at the center of the circle. The control law effectively guides the robot along a smooth path to the goal.

Full Code

```
% For a differential drive mobile robot as shown in the picture,
program in Matlab to simulate the paths shown in Figure 3.20 of the
Textbook, where the robot is initially on a circle in the xy plane.
All movements should have smooth trajectories toward the goal in the
center.

% Initial Position
r = 50
angle = -3*pi/4
x_0 = r*cos(angle)
y_0 = r*sin(angle)
```

```
theta_0 = 0
X0 = [x_0; y_0; theta_0]

% Solve ODE
tspan = [0, 10]
[t, X] = ode45(@f, tspan, X0)

% Plot
plot(X(:,1), X(:,2))

hold on
% Put a thing at the initial point
plot(x_0, y_0, 'g*')
hold on
% Put a thing at the last point
plot(X(end,1), X(end,2), 'b*')
hold off

% ODE Function for xdot
function xdot = f(t, X) % t is time, X is state vector
x = X(1)
y = X(2)
theta = X(3)

% Parameters
k_p = 3 % Proportional Gain for positional error
k_alpha = 8 % Proportional Gain for angular error
k_beta = -1.5 % Proportional Gain for angular error

% Goal
x_f = 0
y_f = 0
theta_f = 0

rho = sqrt((x_f - x)^2 + (y_f - y)^2) % Distance to goal

alpha = atan2(y_f - y, x_f - x) - theta % Angle to goal from current
heading
beta = -alpha - theta % Angle to goal

% Control Law
v = k_p*rho
omega = k_alpha*alpha + k_beta*beta

% Define xdot
xdot = [v*cos(theta); v*sin(theta); omega]
end
```