ME 425 HW 2

Arda Gencer

November 2025

1 Derivation Of System Description

The system that will be used is given at Figure 1. All computations will be done with respect to the inertial frame unless otherwise specified:

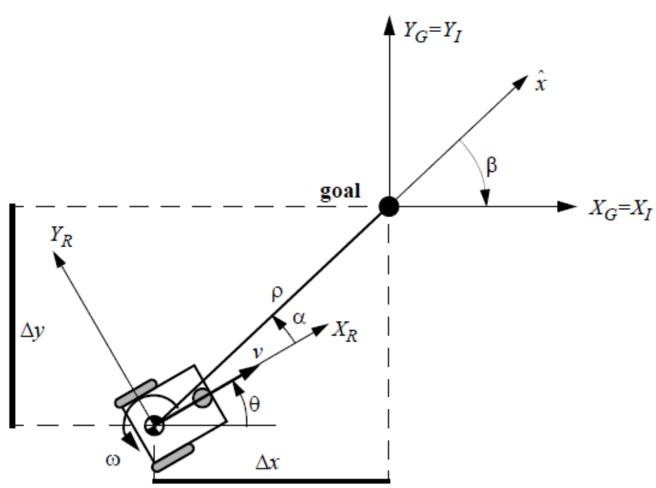

Figure 1: Robot kinematics and its frames of interest.

From equation 3.48 in the textbook, we have the matrix equation:

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

So, $\dot{x}, \dot{y}, \dot{\omega}$ are defined as:

$$\dot{x} = v\cos\theta$$

$$\dot{y} = v\sin\theta$$

$$\dot{\theta} = \omega$$

From equations 3.50 - 3.52, we know that:

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

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

$$\beta = -\theta - \alpha$$

To find the rates of changes of parameters in polar coordinates (i.e. $\dot{\rho}, \dot{\alpha}, \dot{\beta}$) we need to take the time derivatives of the parameters as they are defined in equation 3.50 - 3.52 in the textbook:

1.1 Derivation of $\dot{\rho}$

$$\dot{\rho} = \frac{d}{dt}\sqrt{(\Delta x)^2 + (\Delta y)^2} = \frac{1}{2\sqrt{(\Delta x)^2 + (\Delta y)^2}}(2\Delta x\frac{d(\Delta x)}{dt} + 2\Delta y\frac{d(\Delta y)}{dt})$$

There are a few simplifications and substitutions we can do here. First, we can simplify the 2s in the numerator and denominator.

We can also substitute:

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

Since $\Delta x = X_G - x, \Delta y = Y_G - y$, and X_G, Y_G are constants, we can compute $\dot{\Delta}x$ and $\dot{\Delta}y$ as:

$$\frac{d(\Delta x)}{dt} = -\dot{x},\ \frac{d(\Delta y)}{dt} = -\dot{y}$$

We can also represent Δx and Δy as:

$$\Delta x = \rho\cos(\theta + \alpha) \text{ and } \Delta y = \rho\sin(\theta + \alpha)$$

If we plug these values back into the equation:

$$\dot{\rho} = \frac{1}{\rho}(\rho\cos(\theta + \alpha) \cdot -\dot{x} + \rho\sin(\theta + \alpha) \cdot -\dot{y})$$

If we simplify further and also substitute

$$\dot{x} = v\cos\theta$$

$$\dot{y} = v\sin\theta$$

We will end up with:

$$\dot{\rho} = -v(\cos(\theta + \alpha)\cos\theta + \sin(\theta + \alpha)\sin\theta)$$

If we use the trigonometric identities:

$$\cos(\theta + \alpha) = \cos\theta\cos\alpha - \sin\theta\sin\alpha$$

$$\sin(\theta + \alpha) = \sin\theta\cos\alpha + \cos\theta\sin\alpha$$

The equation becomes:

$$\begin{aligned}\dot{\rho} &= -v((\cos\theta\cos\alpha - \sin\theta\sin\alpha)\cos\theta + (\sin\theta\cos\alpha + \cos\theta\sin\alpha)\sin\theta) \\ &= -v(\cos^2\theta\cos\alpha - \sin\theta\cos\theta\sin\alpha + \sin^2\theta\cos\alpha + \cos\theta\sin\theta\sin\alpha) \\ &= -v(\cos^2\theta\cos\alpha + \sin^2\theta\cos\alpha) = -v\cos\alpha\end{aligned}$$

So,

$$\dot{\rho} = -v\cos\alpha$$

1.2 Derivation of $\dot{\alpha}$

$$\dot{\alpha} = -\dot{\theta} + \frac{d(\text{atan2}(\Delta y, \Delta x)}{dt}$$

The derivative of atan2 can be computed as:

$$\frac{d(\text{atan2}(\Delta y, \Delta x)}{dt} = \frac{\partial(\text{atan2}(\Delta y, \Delta x))}{\partial(\Delta x)}\frac{d(\Delta x)}{dt} + \frac{\partial(\text{atan2}(\Delta y, \Delta x))}{\partial(\Delta y)}\frac{d(\Delta y)}{dt}$$

Partial derivatives will be equal to:

$$\frac{\partial(\text{atan2}(\Delta y, \Delta x))}{\partial(\Delta x)} = -\frac{\Delta y}{\Delta x^2 + \Delta y^2}$$

$$\frac{\partial(\text{atan2}(\Delta y, \Delta x))}{\partial(\Delta y)} = \frac{\Delta x}{\Delta x^2 + \Delta y^2}$$

So,

$$\begin{aligned}\frac{d(\text{atan2}(\Delta y, \Delta x))}{dt} &= \frac{\Delta y\dot{x}}{\Delta x^2 + \Delta y^2} - \frac{\Delta x\dot{y}}{\Delta x^2 + \Delta y^2} = \frac{\dot{x}\Delta y - \Delta x\dot{y}}{\Delta x^2 + \Delta y^2} \\ &= v\rho\frac{\cos(\theta)\sin(\theta + \alpha) - \sin(\theta)\cos(\theta + \alpha)}{\rho^2} \\ &= v\frac{\cos\theta(\sin\theta\cos\alpha + \cos\theta\sin\alpha) - \sin\theta(\cos\theta\cos\alpha - \sin\theta\sin\alpha)}{\rho} \\ &= v\frac{(\cos\theta\sin\theta\cos\alpha + \cos^2\theta\sin\alpha - \sin\theta\cos\theta\cos\alpha + \sin^2\theta\sin\alpha)}{\rho} \\ &= v\frac{\sin\alpha}{\rho}\end{aligned}$$

If we substitute this back into the original equation:

$$\dot{\alpha} = -\dot{\theta} + v\frac{\sin\alpha}{\rho}$$

1.3 Derivation of $\dot{\beta}$

$$\dot{\beta} = -\dot{\theta} - \dot{\alpha}$$

Here, we can use the derivation we got for $\dot{\alpha}$

$$\dot{\beta} = -\dot{\theta} - (-\dot{\theta} + v\frac{\sin\alpha}{\rho}) = -\frac{\sin\alpha}{\rho}$$

Finally, by combining these 3 equations and writing them in matrix form, we reach the equation 3.53 in the textbook:

$$\begin{bmatrix} \dot{\rho} \\ \dot{\alpha} \\ \dot{\beta} \end{bmatrix} = \begin{bmatrix} -\cos\alpha & 0 \\ \frac{\sin\alpha}{\rho} & -1 \\ -\frac{\sin\alpha}{\rho} & 0 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{1}$$

2 Simulation Of The System In Simulink And MATLAB

2.1 System Modeling And Controller Design In Simulink

The system was simulated as a subsystem that takes v and ω as inputs and gives out ρ, α and β as outputs inside Simulink as shown in the figure:

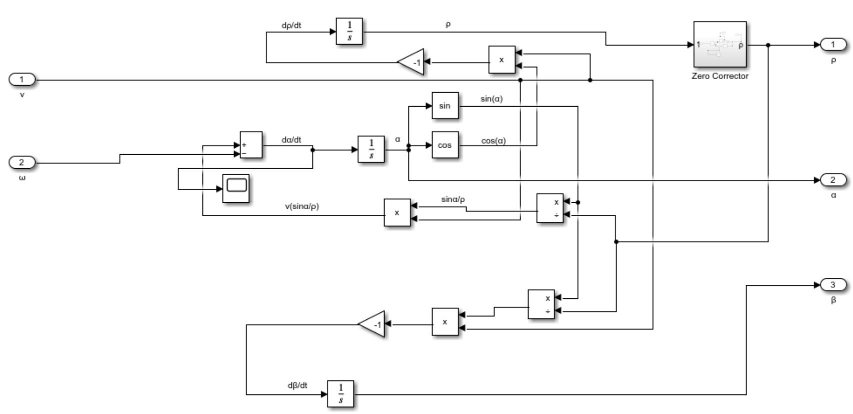

Figure 2: The system (plant) inside Simulink

A zero corrector was also implemented to ensure a zero term won't happen in denominators in any of the integrators inputs. It will instead provide a very small non zero value so that the inputs won't be infinity. The implementation is as follows:

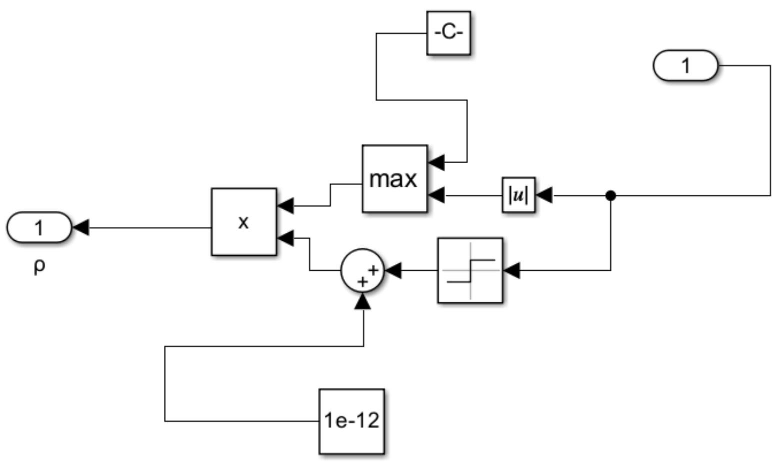

Figure 3: The Zero Corrector

We then need to use the control law:

$$v = k_\rho \rho$$

$$\omega = k_\alpha \alpha + k_\beta \beta$$

to implement a controller block that will take ρ, α and β as inputs from the system and will output v and ω to plug back into the system. The controller is implemented as a subsystem as follows:

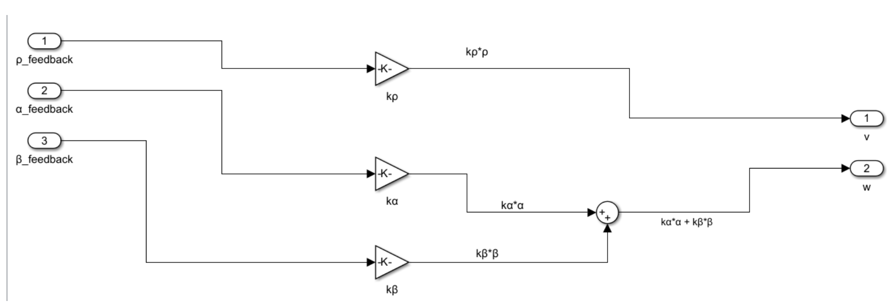

Figure 4: Controller Subsystem

The whole system is combined as follows:

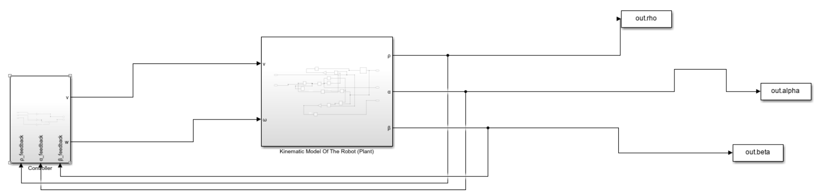

Figure 5: The Closed Loop System

Listing 1: The MATLAB Code Used To Run Simulink Model And Plot Results

```
clc;clear;close all;
%Variable Initialization
rho_initial=5;
alpha_initial=pi;
beta_initial=pi/4;
k_rho=1;
k_alpha=2;
k_beta=-1;
simTime = 20;

%Running the simulation
out=sim("ME425HW2SIMULINK.slx");

%Getting the variables from the simulation
t=out.tout;
rho=out.rho;
alpha=out.alpha;
beta=out.beta;


%Plotting
figure;
subplot(3,1,1);
plot(t,rho);
xlabel("Time (s)");
ylabel("ρ (m)");
grid on;
subplot(3,1,2);
plot(t,alpha);
xlabel("Time (s)");
ylabel("α (rad)");
grid on;
subplot(3,1,3);
plot(t,beta);
xlabel("Time (s)");
ylabel("β (rad)");
grid on;
```

2.2 Testing The System With Different Initial Conditions In MATLAB

After this point, the system was run on MATLAB with the pre-determined initial conditions and controller parameters. For all simulations, the parameters that are used are: $k_\rho = 0.45, k_\alpha = 3, k_\beta = -1$. Initially, the parameters were set as $k_\rho = 0.15, k_\alpha = 2, k_\beta = -1$. However, this resulted in robot not being able to reach the goal in simulation time. So, k_ρ was increased to compensate for this. The simulation was run for 20 seconds at each test. The initial conditions for each test and the parameters and trajectory is as follows:

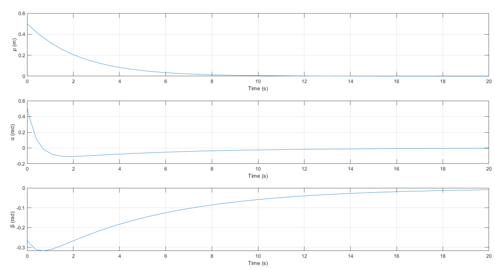

Figure 6: The system parameters with initial conditions: $\rho_0 = 0.5\ m, \alpha_0 = \pi/6\ rad, \beta_0 = \pi/12\ rad$

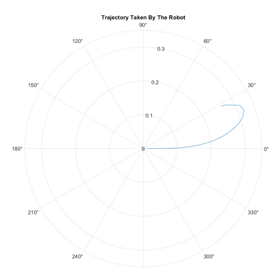

Figure 7: The trajectory taken with initial conditions: $\rho_0 = 0.5\ m, \alpha_0 = \pi/6\ rad, \beta_0 = \pi/12\ rad$

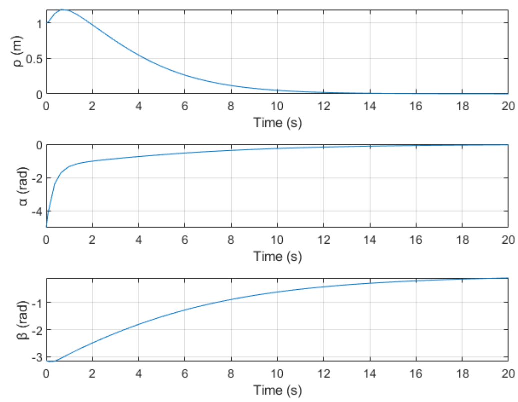

Figure 8: The system parameters with initial conditions: $\rho_0 = 1\ m, \alpha_0 = -1.59\pi\ rad, \beta_0 = -\pi\ rad$

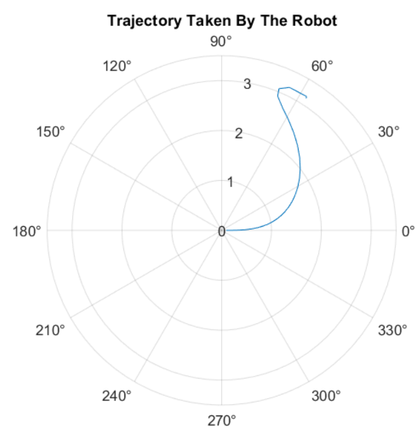

Figure 9: The trajectory taken with initial conditions: $\rho_0 = 1\ m, \alpha_0 = -1.59\pi\ rad, \beta_0 = -\pi\ rad$

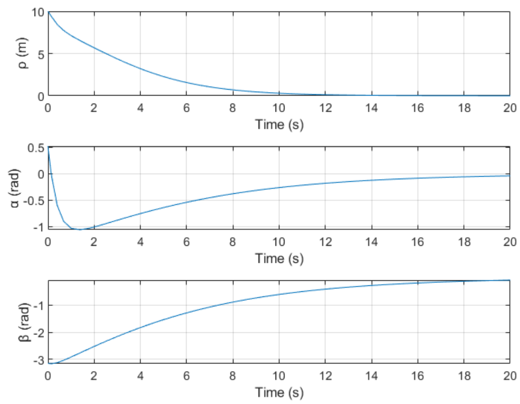

Figure 10: The system parameters with initial conditions: $\rho_0 = 10\ m, \alpha_0 = \pi/6\ rad, \beta_0 = -\pi\ rad$

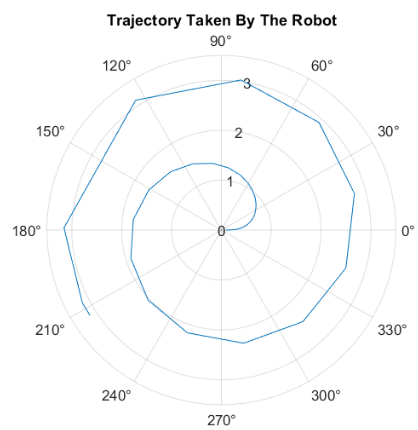

Figure 11: The trajectory taken with initial conditions: $\rho_0 = 10\ m, \alpha_0 = \pi/6\ rad, \beta_0 = -\pi\ rad$

2.3 Discussion Of The Results

For each of the tests, the robot's pose parameters converged to zero in a relatively quick time (8-10 seconds). However, for some initial conditions, the robot took a relatively long path than it could to reach the destination as quickly as possible. In general, it can be said that the system is stable since the parameters manage to converge to zero even if very large values of ρ are used. It also converges to zero for any α value (both values that make the robot face the goal and values that make it face away from the goal).