FRA331: Basic Control Theory

Homework 3: Dynamic Simulation (MATLAB & SImulink)

There is no written part for this homework Assignment

MATLAB & Simulink Programming

1.) STATIC STATE FEEDBACK CONTROLLER

A simple dynamic model for differential drive robot can be written as the followings.

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

or

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} u_1\cos(x_3) \\ u_1\sin(x_3) \\ u_2 \end{bmatrix}$$

Our control goal is to drive this differential drive robot to a specified location in a map. The specified location can be represented as a pair of coordinate $\vec{p}_g$ in Cartesian space.

$$\vec{p}_g = \begin{bmatrix} x_g \\ y_g \end{bmatrix}$$

There are 2 inputs, linear velocity v and angular velocity ω . We can accomplish this task by using "go-to-goal algorithm".

The command angular velocity ω is set to be the followings.

$$\omega = K_p atan2(\sin(e), \cos(e))$$

where

$$e = atan2(y_g - y, x_g - x) - \theta$$

x, y, θ are the current positions and orientation of the robot.

K_p is a control gain.

The command linear velocity is set to be constant except when the robot is relatively close to the goal. We can write the control law for linear velocity as the followings.

$$v = v_0 * \left(\sqrt{(x_g - x)^2 + (y_g - y)^2} \geq \varepsilon\right)$$

Notice: the term in the parenthesis is a logical statement. In MATLAB, if a logical statement is true, it will provide 1. Otherwise, the logical statement will return 0.

Let

$$\begin{bmatrix} x(0) \\ y(0) \\ \theta(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \qquad \begin{bmatrix} x_g \\ y_g \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}, \qquad v_0 = 1, \qquad \varepsilon = 0.01\,, \qquad K_p = 10$$

a.) Create a function called "differentialDrive" which describes the dynamics of the system.
- The input arguments of the function are time 't', state variables 'x', and control input 'u'
- The output argument of the function is the rate of change of state variables 'dx'

b.) Create a function called "goToGoal" which described the stated control law.
- The input arguments of the function are time 't', state variables 'x', and desired location 'p_g'
- The output arguments of the function is control input 'u'

c.) Create a script called "simDifferentialDrive" which use ode45 to simulate the closed-loop system. The template for this script is provided. $t_{max} = 10$

YOU MUST SAVE ALL FUNCTIONS AS m-files

2.) DYNAMIC STATE FEEDBACK CONTROLLER

Given a nonlinear first-order dynamic system

$$\dot{x}_1 = kx_1^2 + u_1$$

This particular system has an unknown constant parameter k. We only know the nominal value of k_0.

Our goal is to move x_1 along a desired position trajectory $x_d(t)$. In other words, the error between the state x_1 and the desired trajectory should approach zero.

$$x_d(t) - x_1 = e \to 0$$

The problem is we do not know the parameter k.

To remedy this situation, an adaptive control law is used. The control law is given as the following.

$$u(t, x_1, \hat{k}) = \dot{x}_d(t) + K_p(x_d(t) - x_1) - \hat{k}x_1^2$$

where

$\hat{k}$ is an estimated parameter k
$\dot{x}_d$ is a desired velocity trajectory (time-derivative of desired position)
K_p is a control gain

The adaptive law (integration law) can be defined as the following.

$$\hat{k} = \int_{\tau=0}^{t} \left(-K_a x_1^2 (x_d(t) - x_1)\right) d\tau$$

or

$$\frac{d}{dt}(\hat{k}) = f_z(t, \vec{x}, \vec{z}) = -K_a x_1^2 (x_d(t) - x_1)$$

$$\vec{z} = [\hat{k}]$$

Let

$$x_1(0) = 0, \qquad x_d(t) = \cos(\pi t), \qquad K_p = 1, \qquad K_a = 50, \qquad \hat{k}(0) = 5,$$

a.) Create a function called “nonlinearDyn” which describes the dynamics of the system.
- The input arguments of the function are time ‘t’, state variables ‘x’, control input ‘u’, and parameter ‘k’
- The output argument of the function is the rate of change of state variables ‘dx’

b.) Create a function called “trajTracking” which described the stated control law.
- The input arguments of the function are time ‘t’, state variables ‘x’, integration variable ‘z’, desired position trajectory ‘x_d’, an desired velocity trajectory ‘dx_d’.
- The output arguments of the function is control input ‘u’

c.) Create a function called “adaptiveLaw” which described the adaptive law.
- The input arguments of the function are time ‘t’, state variables ‘x’, integration variable ‘z’, and desired position trajectory ‘x_d’

- The output arguments of the function is the rate of change of the integration variable ‘dz’

d.) Create a script called “simNonlinear” which use ode45 to simulate the closed-loop system. The template for this script is provided. $t_{max} = 10$

YOU MUST SAVE ALL FUNCTIONS AS m-files

HINT:

$$\dot{\vec{s}} = \begin{bmatrix} \dot{s}_1 \\ \dot{s}_2 \end{bmatrix} = \begin{bmatrix} \dot{\vec{x}} \\ \dot{\vec{z}} \end{bmatrix} = \begin{bmatrix} f(t, \vec{x}, \vec{u}) \\ f_z(t, \vec{x}, \vec{z}) \end{bmatrix} = \begin{bmatrix} \dot{x}_1 \\ \dfrac{d}{dt}\hat{k} \end{bmatrix} = \begin{bmatrix} f(t, x_1, u_1) \\ f_z\left(t, x_1, \hat{k}\right) \end{bmatrix} = \begin{bmatrix} f\left(t, s_1, u_1(t, s_1, s_2)\right) \\ f_z(t, s_1, s_2) \end{bmatrix}$$

$$\vec{u}(t, \vec{x}.\vec{z}) = u_1\left(t, x_1, \hat{k}\right)$$

$$\vec{s}(0) = \begin{bmatrix} x_1(0) \\ \hat{k}(0) \end{bmatrix}$$

When you run the script. The error in position should reduce down to zero.

3.) SIMULATION in Simulink

Given the dynamics of a DC motor as follows

$$J_m \frac{d^2\theta_i}{dt^2} = -B_m \frac{d\theta_i}{dt} + K_m i - \tau_i$$

$$L \frac{di}{dt} = -K_b \frac{d\theta_i}{dt} - Ri + v_{in}$$

$$\tau_o = r\tau_i$$

$$\theta_o = \frac{\theta_i}{r}$$

The state-space representation is the following.

$$\vec{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \theta_i \\ \dfrac{d\theta_i}{dt} \\ i \end{bmatrix}$$

$$\dot{\vec{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} \dfrac{d\theta_i}{dt} \\ \dfrac{d^2\theta_i}{dt^2} \\ \dfrac{di}{dt} \end{bmatrix} = \begin{bmatrix} x_2 \\ \dfrac{1}{J_m}\left(-B_m x_2 + K_m x_3 - \dfrac{u_2}{r}\right) \\ \dfrac{1}{L}(-K_b x_2 - R x_3 + u_1) \end{bmatrix}$$

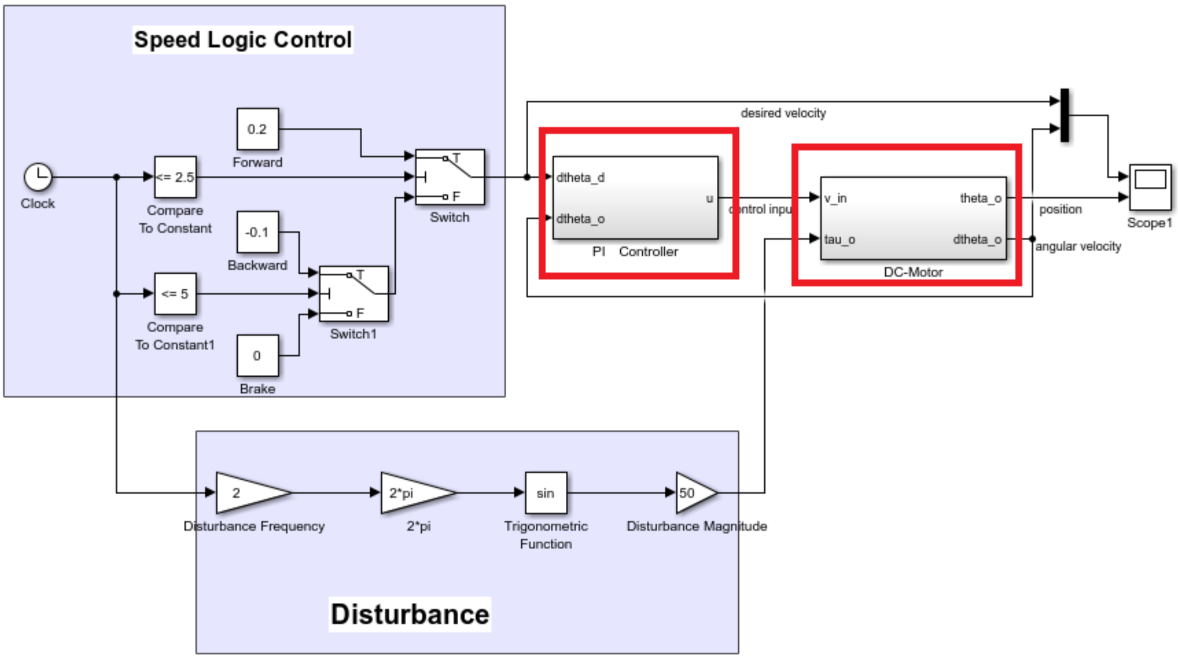
Our goal is to regulate the speed of the motor's output shaft to the given desired speed. In other words, the error between the desired speed and the actual speed of the motor's output shaft much approach zero.

$$\dot{\theta}_d - \dot{\theta}_o = e \to 0$$

To do this, we can use PI Controller.

$$u = K_p\left(\dot{\theta}_d - \dot{\theta}_o\right) + K_i \int_{\tau=0}^{t} \left(\dot{\theta}_d - \dot{\theta}_o\right) d\tau$$

Using the provided template, **complete** the subsystems "PI Controller" and "DC-Motor".



Let

$$J_m = 2, \quad B_m = 0.1, \quad r = 20, \quad K_m = 10\,, \quad K_b = 1, \quad L = 0.1, \quad R = 10,$$

$$K_p = 150, \quad K_i = 70, \quad \begin{bmatrix} \theta_i(0) \\ \dot{\theta}_i(0) \\ i(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \quad t_{max} = 10$$

The result should look like this from the scope.