Control
Lab One

Prepared By

Ahmed Aly Gamal El-Din El-Ghannam

Electronics and Communications - Level 4
ID: 19015292

October 2023

Contents

Listings

1 Introduction

This report showcases the results of simulating the block diagram shown in figure 1 using both Simulink and MatLAB. All the simulation files and MatLAB codes used to produce this result can be found in the lab's Github Repository.

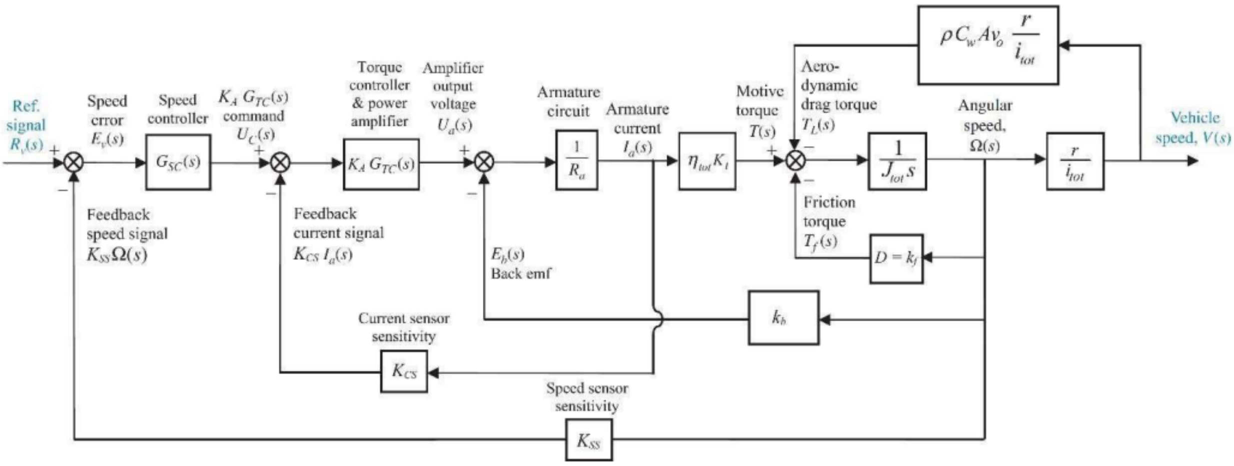

Figure 1: Block Diagram of the System as Described in the Lab Manual

2 System Representation on Simulink

2.1 Simulation Parameters and Block Diagram

Figure 2 depicts the system's block diagram on Simulink. All the transfer functions and gain values are stored in a separate MatLAB script shown in code snippet 10 for convenience. The reference signal is determined by the variable ID : its value ranges from 0 to 9 corresponding to a total of 10 signals as highlighted in both code snippet 10 and figure 8. Sampling time for the input signal is set to 0.01 second and simulation duration is 8 seconds.

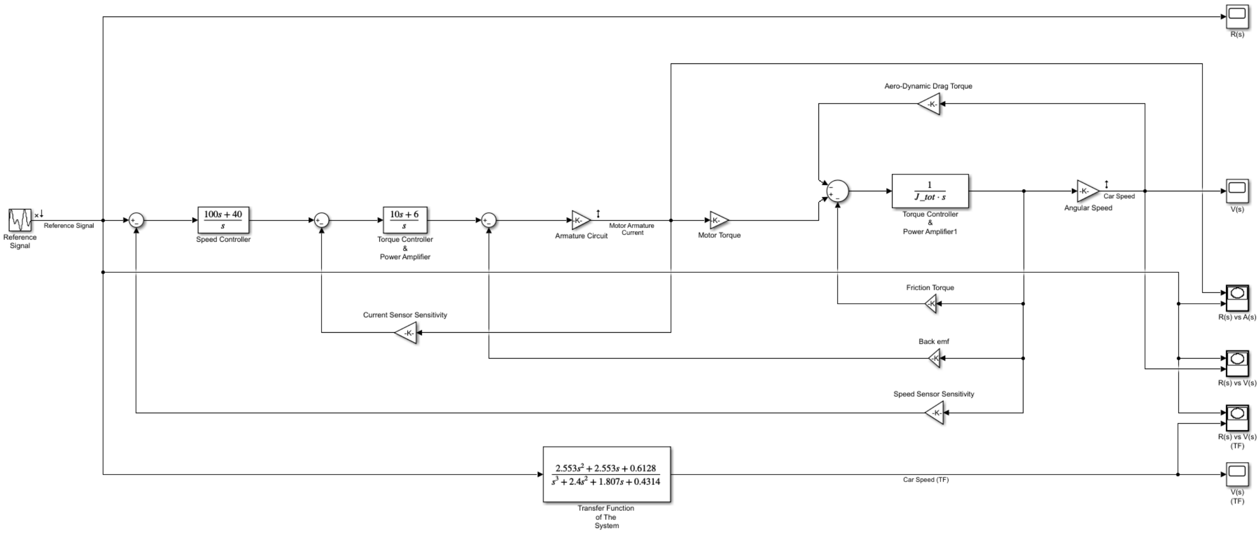

Figure 2: System Block Diagram on Simulink

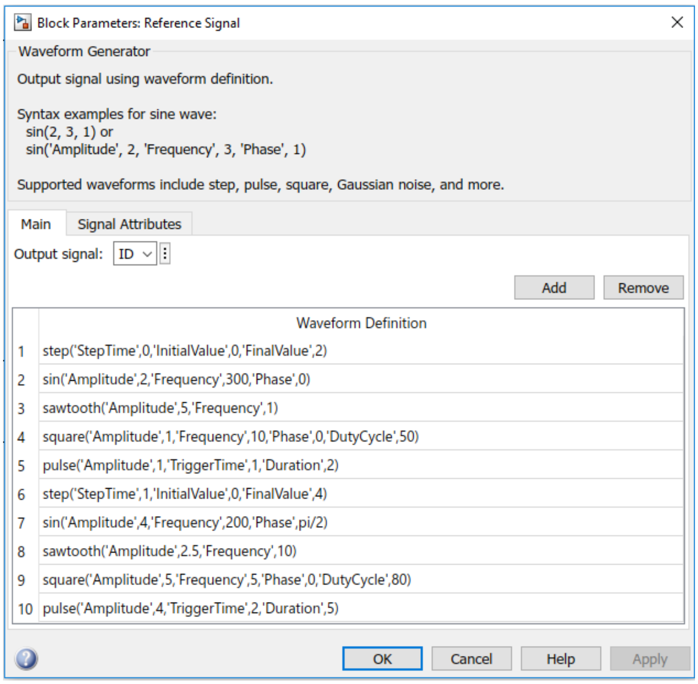

Figure 3: Reference Signals Choices

```matlab
%% Block Diagram Parameters

% Current Sensor Sensitivity
K_cs = 0.5;

% Speed Sensor Sensitivity
K_ss = 0.0433;

% Motor Inertia
J_tot = 7.226;

% Motor Resistor
R_a = 1;

% Viscous Friction
k_f = 0.1;

% Back emf Constant
k_b = 2;

% Vehicle Dynamics
r__i_tot = 0.0615;
p_Cw_A_vo_r__i_tot = 0.6154;

% Motor Torque
m_tot__K_t = 1.8;

%% Input Signal ID
% ID 0->9 chooses the input waveform
% ID =  0: step('StepTime',0,'InitialValue',0,'FinalValue',2)
% ID =  1: sin('Amplitude',2,'Frequency',300,'Phase',0)
% ID =  2: sawtooth('Amplitude',5,'Frequency',1)
% ID =  3: square('Amplitude',1,'Frequency',10,'Phase',0,'DutyCycle',50)
% ID =  4: pulse('Amplitude',1,'TriggerTime',1,'Duration',2)
% ID =  5: step('StepTime',1,'InitialValue',0,'FinalValue',4)
% ID =  6: sin('Amplitude',4,'Frequency',200,'Phase',pi/2)
% ID =  7: sawtooth('Amplitude',2.5,'Frequency',10)
% ID =  8: square('Amplitude',5,'Frequency',5,'Phase',0,'DutyCycle',80)
% ID =  9: pulse('Amplitude',4,'TriggerTime',2,'Duration',5)

choice = 2;
ID = choice + 1;
```

Code Snippet 1: MatLAB Script to Store Constants

2.2 Results

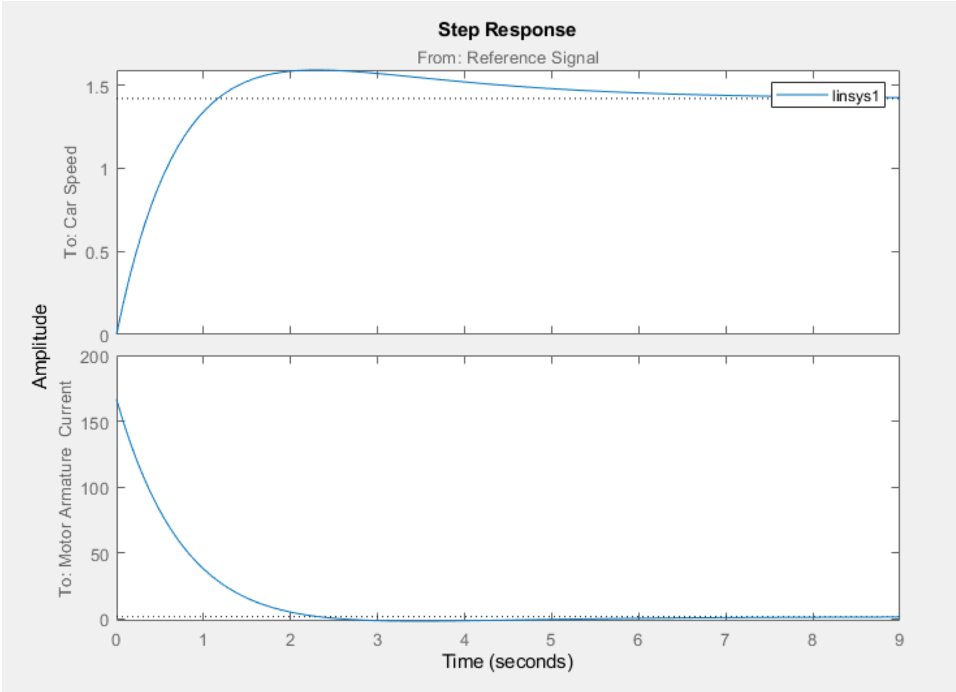

Figure 4: Step Response

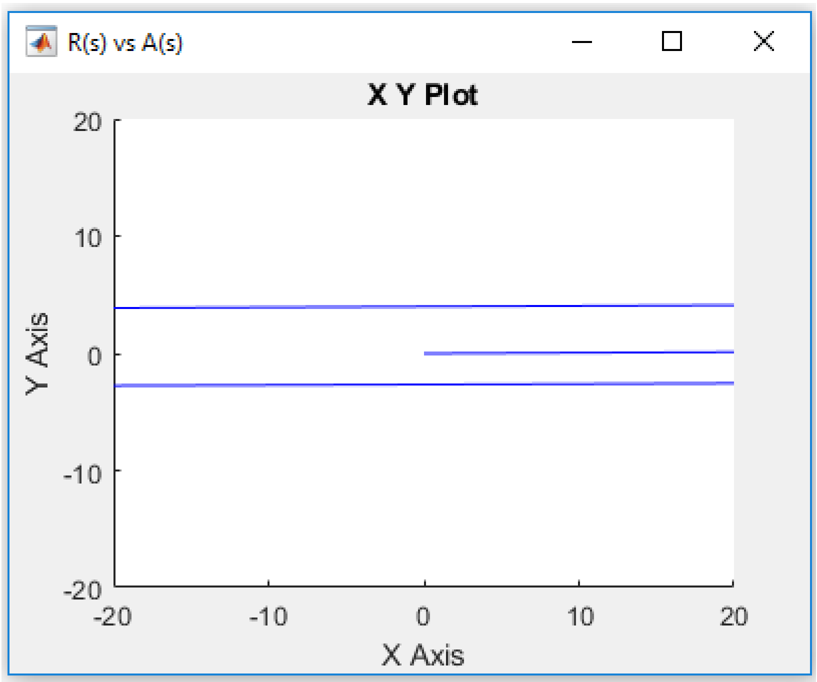

Figure 5: Armature Current vs Reference Input

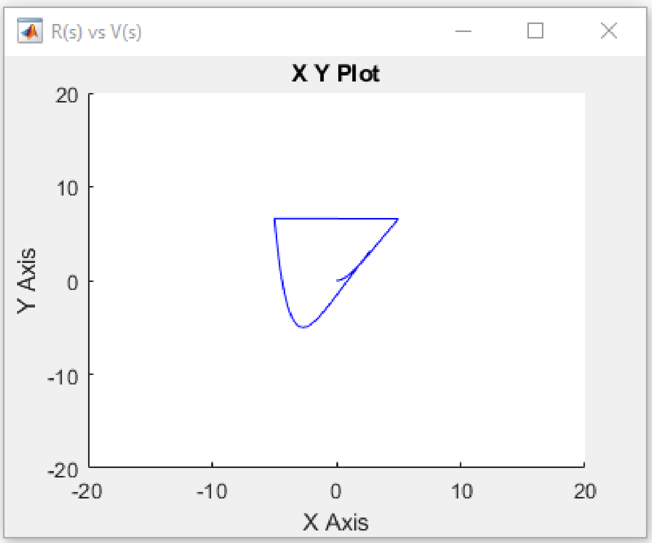

Figure 6: Vehicle Speed vs Reference Input

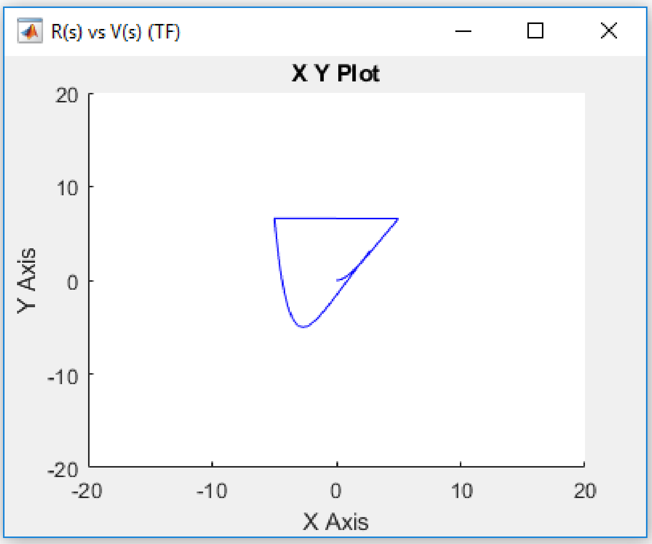

Figure 7: Vehicle Speed vs Reference Input Using Equivalent System Transfer Function

```
>> tf(linsys1)

ans =

  From input "Reference Signal" to output...
                 2.553 s^2 + 2.553 s + 0.6128
   Car Speed:  --------------------------------
               s^3 + 2.4 s^2 + 1.807 s + 0.4314

                             166.7 s^3 + 169.8 s^2 + 43.18 s + 0.7631
   Motor Armature  Current:  ----------------------------------------
                                 s^3 + 2.4 s^2 + 1.807 s + 0.4314

Name: Linearization at model initial condition
Continuous-time transfer function.
```

Figure 8: Transfer Functions Required in Lab Manual

3 System Representation on MatLAB

```
%% System Parameters

% Gain Blocks
GAIN_speedSensorSensitivity = 0.0433;                          % Kss
GAIN_currentSensorSensitivity = 0.5;                           % Kcs
GAIN_motorInertia = 7.226;                                     % Jtot
GAIN_motorResistor = 1;                                        % Ra
GAIN_viscousFriction = 0.1;                                    % kf
GAIN_backEMFConstant = 2;                                      % kb
GAIN_aeroDynamicDragTorque = 0.6154;                           % pCwAvo
GAIN_vehicleDynamics = 0.0615;                                 % r__i_tot
GAIN_motorTorque = 1.8;                                        % MtotKt

% Transfer Function Blocks
TF_speedController = tf([100 40], [1 0]);                      % Gsc(s)
TF_torqueControllerAndPowerAmp = tf([10 6], [1 0]);            % Ka.Gtc(s)
TF_angularSpeed = tf([0 1], [GAIN_motorInertia 0]);            % 1/Jtot.s
```

Code Snippet 2: System Parameters

```
%% Move the takeoff point (Angular Velocity) one block to the right
```

Code Snippet 3: Block Diagram Simplification (Move Takeoff Point)

```
%% Total Transfer Function (G1) From Motive Torque (T(s)) to Vehicle
   Speed (V(s))

TTF_feedback1_vehicleSpeedOverMotiveTorque =
   tf([GAIN_aeroDynamicDragTorque], [1]);
TTF_feedback2_vehicleSpeedOverMotiveTorque =
   series(tf([GAIN_viscousFriction], [1]), tf([1],
   [GAIN_vehicleDynamics]));
TTF_feedback_vehicleSpeedOverMotiveTorque =
   parallel(TTF_feedback1_vehicleSpeedOverMotiveTorque,
   TTF_feedback2_vehicleSpeedOverMotiveTorque); % 2.241
TTF_forward_vehicleSpeedOverMotiveTorque = series(tf([TF_angularSpeed],
   [1]), tf([GAIN_vehicleDynamics], [1]));
TTF_G1_vehicleSpeedOverMotiveTorque =
   feedback(TTF_forward_vehicleSpeedOverMotiveTorque, 2.241);
```

Code Snippet 4: Block Diagram Simplification (G1)

```
%% Total Transfer Function (G2) From Armature Current (Ia(s)) to Vehicle
   Speed (V(s))

TTF_G2_vehicleSpeedOverArmatureCurrent =
   series(TTF_G1_vehicleSpeedOverMotiveTorque, tf([GAIN_motorTorque],
   [1]));
```

Code Snippet 5: Block Diagram Simplification (G2)

```
%% Move takeoff point before G2 to the end of system
```

Code Snippet 6: Block Diagram Simplification (Move Takeoff Point)

```
%% Total Transfer Function (G3) From Amplifier Output Voltage (Ia(s)) to
   Vehicle Speed (V(s))

TTF_forward_vehicleSpeedOverAmplifierOutputVoltage = series(tf([1],
   [GAIN_motorResistor]), TTF_G2_vehicleSpeedOverArmatureCurrent);
TTF_feedback_vehicleSpeedOverAmplifierOutputVoltage =
   series(tf([GAIN_backEMFConstant], [1]), tf([1],
   [GAIN_vehicleDynamics])); % 32.52
TTF_G3_vehicleSpeedOverAmplifierOutputVoltage =
   feedback(TTF_forward_vehicleSpeedOverAmplifierOutputVoltage, 32.52);
```

Code Snippet 7: Block Diagram Simplification (G3)

```
%% Total Transfer Function (G4) From Torque Controller (Ka.Gtc(s)) to
   Vehicle Speed (V(s))

TTF_forward_vehicleSpeedOverTorqueController =
   series(TF_torqueControllerAndPowerAmp,
   TTF_G3_vehicleSpeedOverAmplifierOutputVoltage);
TTF_feedback_vehicleSpeedOverTorqueController =
   series(tf([GAIN_currentSensorSensitivity], [1]), 1 /
   TTF_G3_vehicleSpeedOverAmplifierOutputVoltage);
TTF_G4_vehicleSpeedOverTorqueController =
   feedback(TTF_forward_vehicleSpeedOverTorqueController,
   TTF_feedback_vehicleSpeedOverTorqueController);
```

Code Snippet 8: Block Diagram Simplification (G4)

```
%% Total Transfer Function For the System From Reference Signal Rv(s) to
   Vehicle Speed V(s)

TTF_forward_vehicleSpeedOverTorqueController =
   series(TTF_G4_vehicleSpeedOverTorqueController, TF_speedController);
TTF_feedback_vehicleSpeedOverTorqueController =
   series(tf([GAIN_speedSensorSensitivity], [1]), tf([1],
   [GAIN_vehicleDynamics]));
TTF_G5_vehicleSpeedOverTorqueController =
   feedback(TTF_forward_vehicleSpeedOverTorqueController,
   TTF_feedback_vehicleSpeedOverTorqueController);
```

Code Snippet 9: Block Diagram Simplification (G5)

```
%% Simplifying The Expression

TTF_G5_Output = minreal(TTF_G5_vehicleSpeedOverTorqueController);
```

Code Snippet 10: Equivalent Transfer Function of the System