Control and Simulation Tasks Report

By

Abdullah Abdelgalel Abouda

Table of Contents

List of Figures and Tables

Introduction

In the realm of engineering and technology, ensuring the reliability and safety of systems is crucial. Control and simulation tools are incredibly valuable for analyzing, designing, and improving complex systems across various fields, including aerospace, automotive, robotics, and manufacturing. This report delves into the specifics of each task, outlining the methods employed and our approach to them.

Task I- Physical Modeling

Required: *Model The Suspension System in Figure 1 Using Simulink.*

Expected Output*: The Difference Between X1 and X2 over Time with specified Inputs.*

Assumptions:

- ***W*** *is the ground profile (i.e., road disturbance). If W = 0, this means that the road is flat and there are no disturbances. It could be modeled as* ***step input.***
- *The control force,* ***u****, could be modeled as a* ***step input***
- *parameters to run the simulation:*

Parameter	Value
M_1	2500 Kg
M_2	320 Kg
K_1	80000 N/m
B_1	350 N.s/m
K_2	500000 N/m
B_2	15020 N.s/m

Table 1: System Parameters

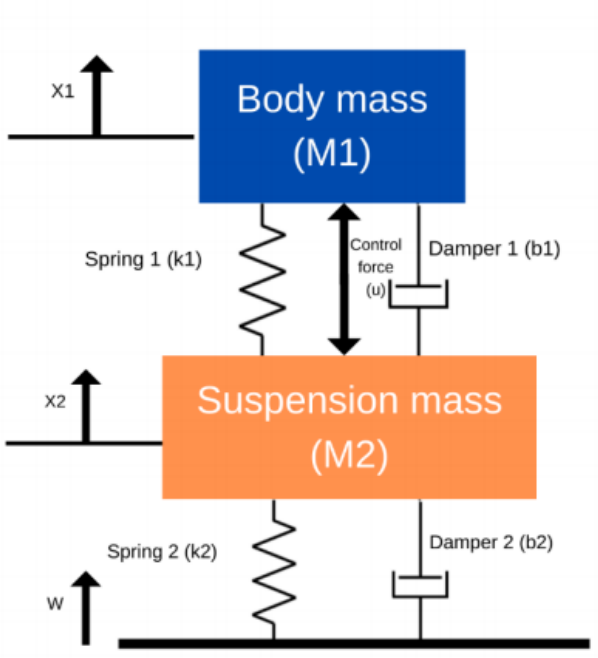


Figure 1: Suspension System Model

Solution Approach

We begin by creating the mechanical translation equations that describe our suspension system. Next, we'll use these equations to build blocks in a Simulink model. Lastly, we'll fine-tune the parameters and run simulations to generate and analyze the results.

Mechanical Translation Equations:

We'll formulate equations for all systems that influence a particular mass. These equations will be in the form of differential equations in the **time domain**.

Note: $\dot{x}_n = \frac{dx_n}{dt}$, $\ddot{x}_n = \frac{d^2x_n}{dt}$

Equation 1 *(Influencing Body Mass or M1)*

$$M_1\ddot{x}_1 + K_1(x_1 - x_2) + b_1(\dot{x}_1 - \dot{x}_2) = u(t)$$

Equation 2 *(Influencing Suspension Mass or M2)*

$$M_2\ddot{x}_2 + K_1(x_2 - x_1) + b_1(\dot{x}_2 - \dot{x}_1) + K_2(x_2 - W) + b_2\left(\dot{x}_2 - \dot{W}\right) = -u(t)$$

Rearrange Equations 1 & 2 to make both $\ddot{x}_1$ & $\ddot{x}_2$ in the L.H.S

Equation 1 *(Influencing Body Mass or M1)*

$$\ddot{x}_1 = \frac{1}{M_1}[u(t) - K_1(x_1 - x_2) - b_1(\dot{x}_1 - \dot{x}_2)]$$

Equation 2 *(Influencing Suspension Mass or M2)*

$$\ddot{x}_2 = \frac{1}{M_2}[K_1(x_1 - x_2) + b_1(\dot{x}_1 - \dot{x}_2) + K_2(W - x_2) + b_2\left(\dot{W} - \dot{x}_2\right) - u(t)]$$

Simulink Model:

Once the equations are written, we'll begin constructing our model blocks. Since we're aiming for x_1 and x_2 outputs with u and W as inputs, we'll start from $\ddot{x}$ and work our way to x by adding integrator blocks, leveraging the equations for $\ddot{x}$ to connect the blocks and create the final model.

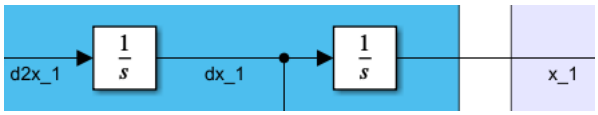


Figure 2: Integrator Blocks

Following that, we'll compute the difference between the signals of x_1 and x_2 attaching a scope to Visualize the Output Signal, as well as between $\dot{x}_1$ and $\dot{x}_2$.Then, we'll extract nodes from these signals and apply the gains of their respective systems, such as Spring 1and Damper 1 through multiplication.

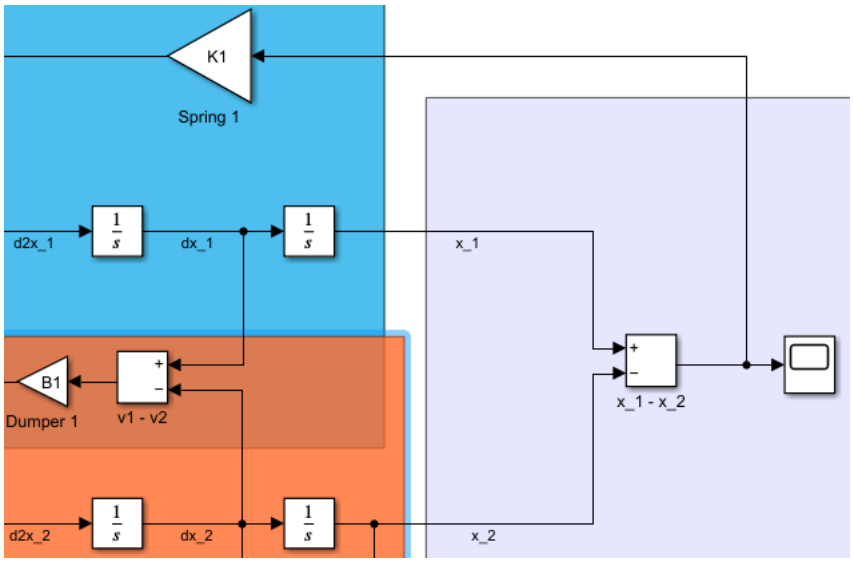


Figure 3: Subtract Blocks, Gain Blocks, and Scope

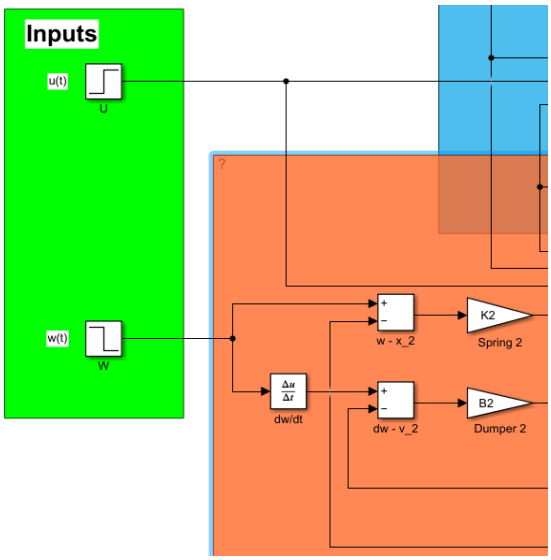
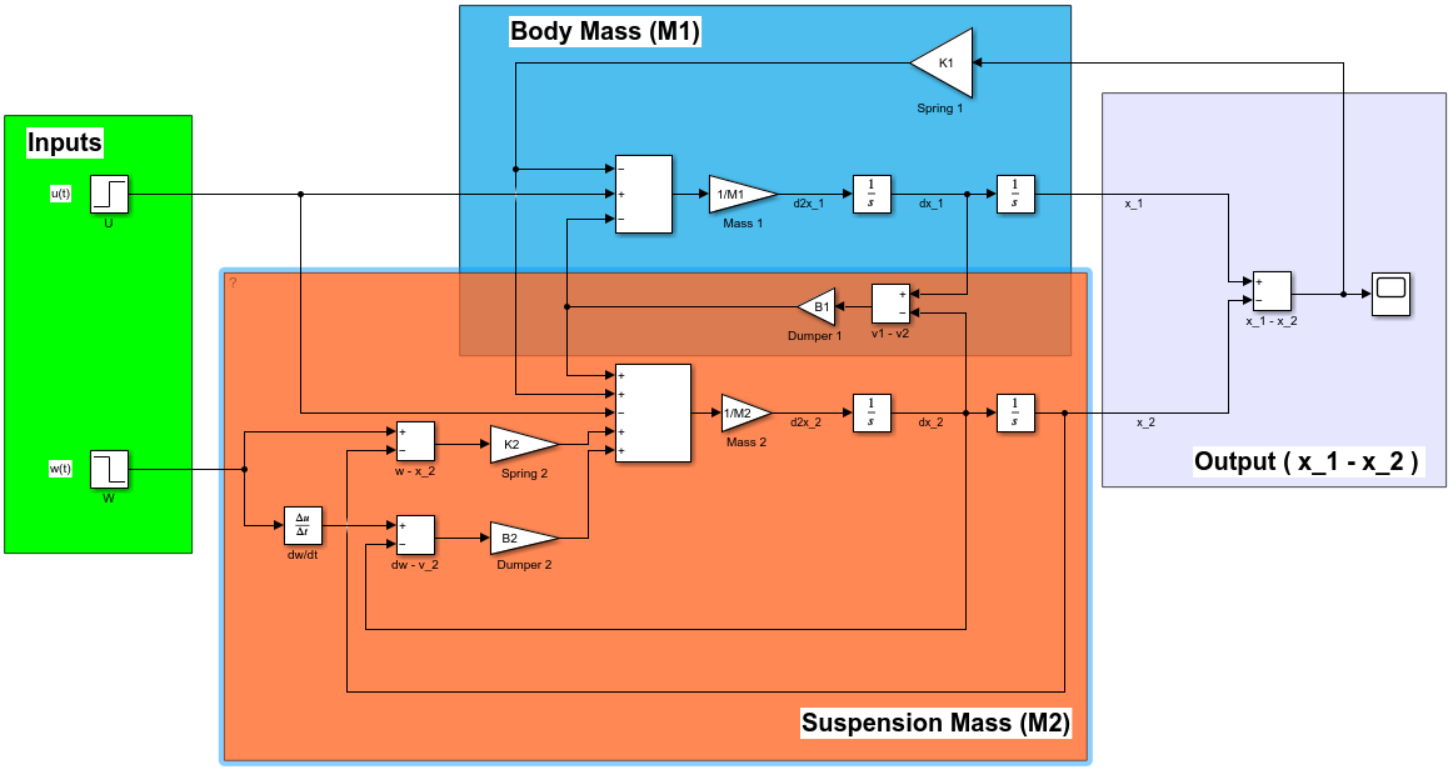


Figure 4: Subtract, Gain, and Step Input Blocks

Following that, we'll compute the difference between the signals of W and x_2, as well as between $\dot{W}$ and $\dot{x}_2$. To derive $\dot{W}$ we will add a derivative block. Input blocks will be added with the given parameters outlined in the Recruitment Report; all *U* parameters will be set to zero, and for *W* parameters, the step time will be 5 seconds with a final value of -0.1. Then, we'll extract nodes from these signals and apply the

gains of their respective systems, such as Spring 2 and Damper 2 through multiplication.

In conclusion, we'll link these blocks with sum blocks and apply the gains through multiplication to compute $\ddot{x}_1$ and $\ddot{x}_2$ thereby finalizing the model by Using Equations 1 and 2.



Note: Spring System 1 and Damper System 1 are shared between the body mass and the suspension mass, affecting both masses simultaneously.

Figure 5: Final Simulink Model

Each gain is assigned a variable name corresponding to the parameters of the physical model, such as *M1, K1, B1*, and so on. All these parameters listed in Table 1 are stored in a MATLAB file named "Physical_Modeling_Parameters," which is linked with the Simulink model file named "Landing_Gear_Suspension_System." If you want to test the same system with different parameters, you can modify the parameters in the MATLAB file named "Physical_Modeling_Parameters."

Simulink Model Output:

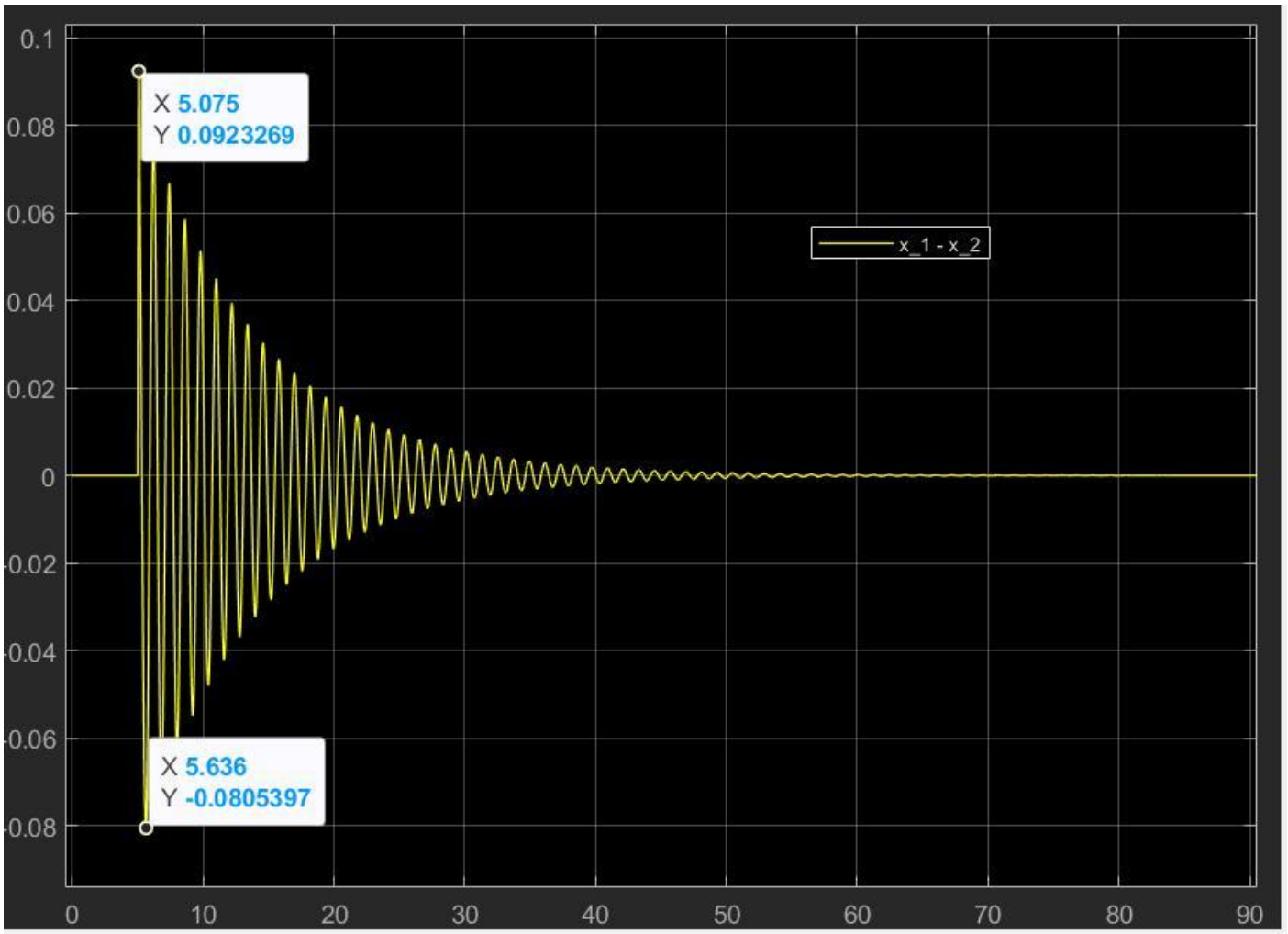


Figure 6: Simulation Output

After the landing gear suspension system impacts the 10 cm pothole at $t = 5$ seconds, we notice that the resulting disturbance causes the distance between the body mass $M1$ and suspension mass $M2$, or $x_1 - x_2$, to oscillate between values of $9.2\ cm$ and $-8.05\ cm$. These oscillations decay over time and return to zero after approximately 60 seconds.

Task II- System Identification

Required: *For the Given Input/Output data in the file "RampTest_60hz.csv" it is required to obtain the values of K and τ which describes the system.*

Assumptions:

- *The System has only a gain and one pole with no zeros*
- *The System can be modeled as the following:* $\frac{output}{input} = \frac{K}{\tau s+1}$

Solution Approach

First, we import the data file into MATLAB. Then, we slice the data into inputs and outputs, saving them as variables. Next, we import these variables into the PID Tuner tool.

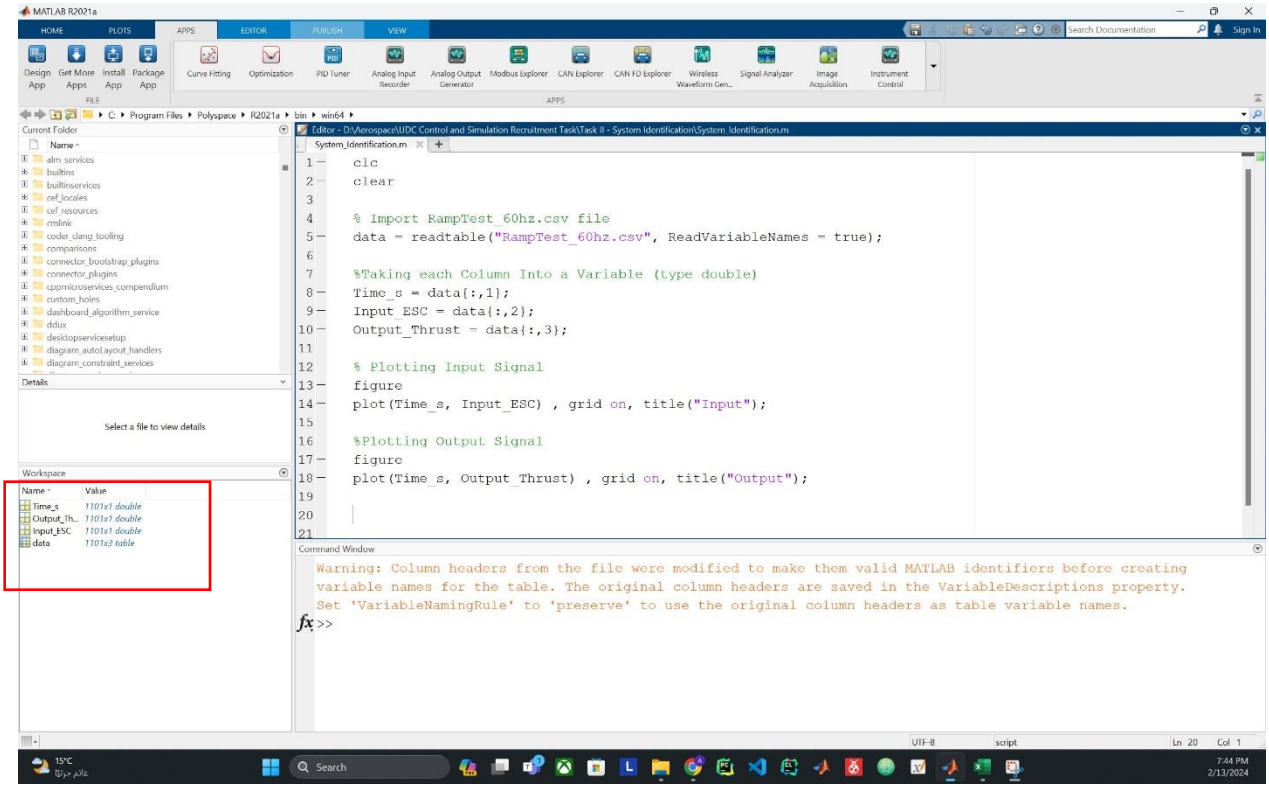


Figure 7: Sorting Input/Output data from the file to single variables

T To import the input/output variables, we access the PID Tuner Tool and designate a new Plant (System). Next, we select "Get I/O Data" followed by "Arbitrary I/O Data" to bring in our data variables, aiding in the identification of the Plant (System).

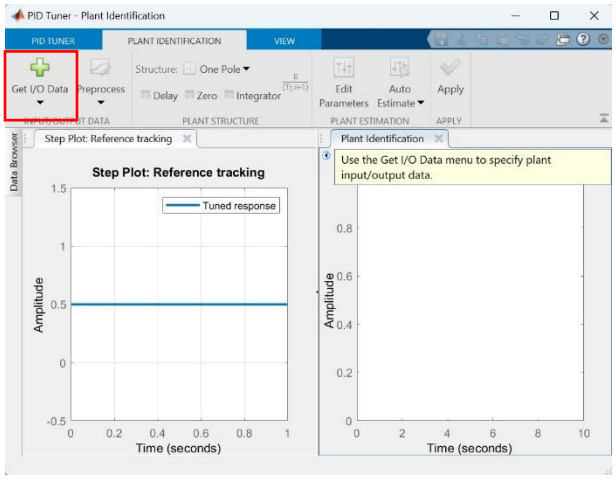


Figure 8: PID Tunner APP Interface

A window will appear prompting you to enter the names of the output and input variables that are already present in the Workspace.

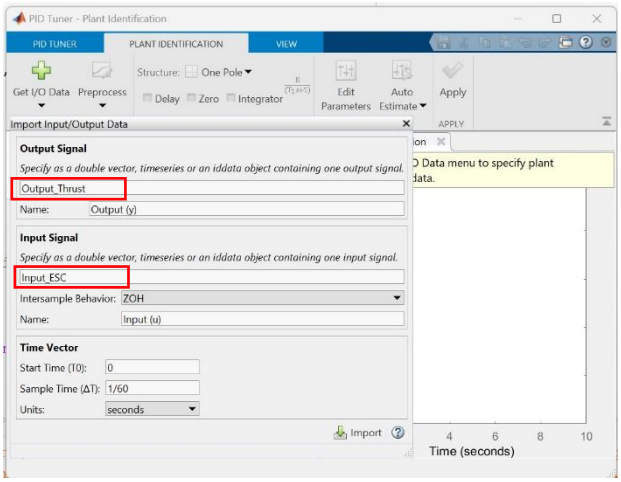


Figure 9: I/O data Window

Once you return to the PID Tuner tool window, the plant output data will be displayed as a green line, while the solver, identifying the plant, will be represented by a blue line. We can edit The Transfer Function Structure from the Plant Structure Section. To achieve the best fitting, we'll utilize the auto-estimate button, employing an optimization algorithm to obtain the optimal fit for the output data.

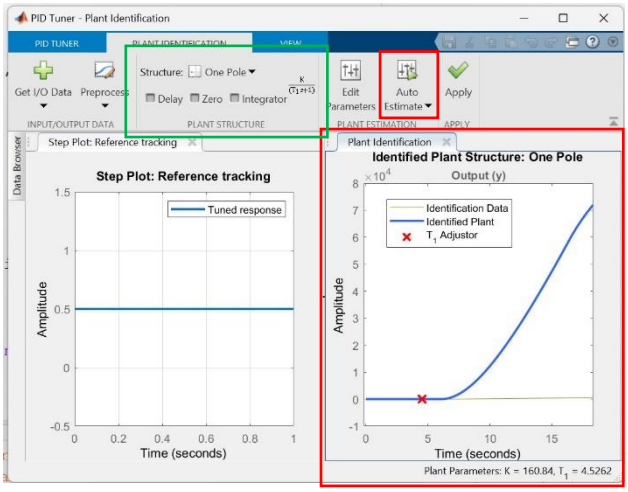


Figure 10: PID Solver Window

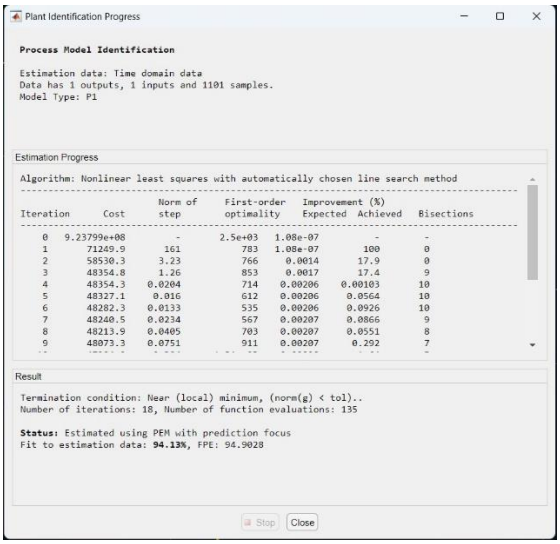


Figure 11: Plant Identification Progress Window.

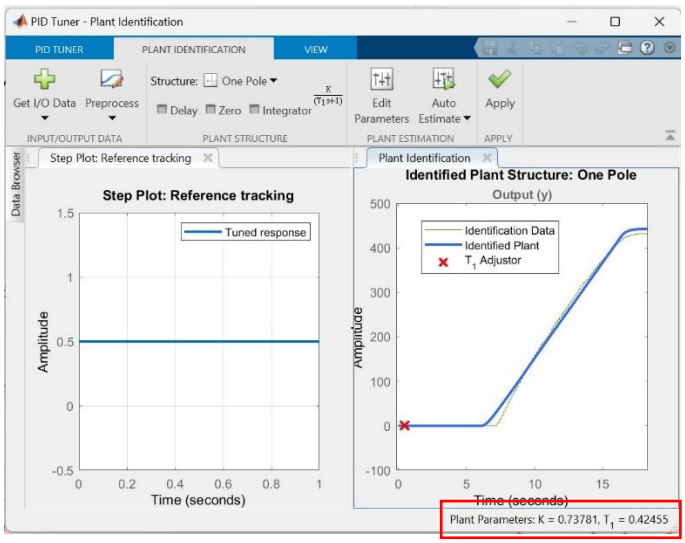


Figure 12: Plant after fitting Process showing the Values of K and T1 in the bottom right Corner.

The results of the System Identification process using the PID Tuner Tool yielded $K = 0.73781$ and $\tau = 0.42455$. These analyzed plant values can be saved in the workspace (see Figure 13). Additionally, the plant parameters can be viewed via the "Plant -> Inspect" option (see Figure 14). Note since MATLAB files will have new workspace at every run a workspace variable file "System_Identification_Workspace_Variables.mat" is saved to have access to all results generated in this task.

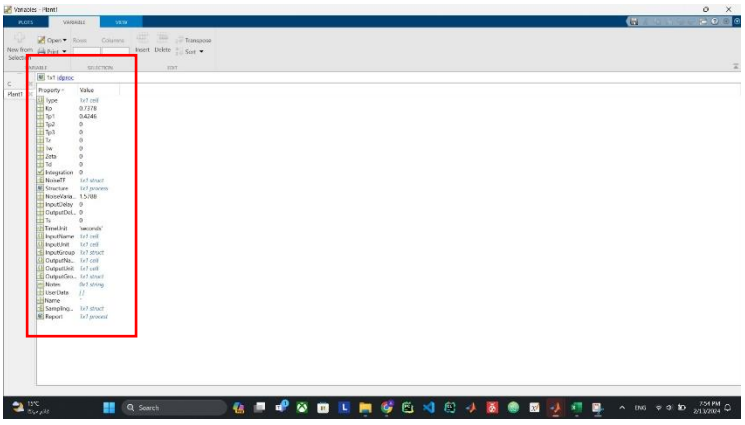


Figure 13: Plant 1 Variable in Workspace

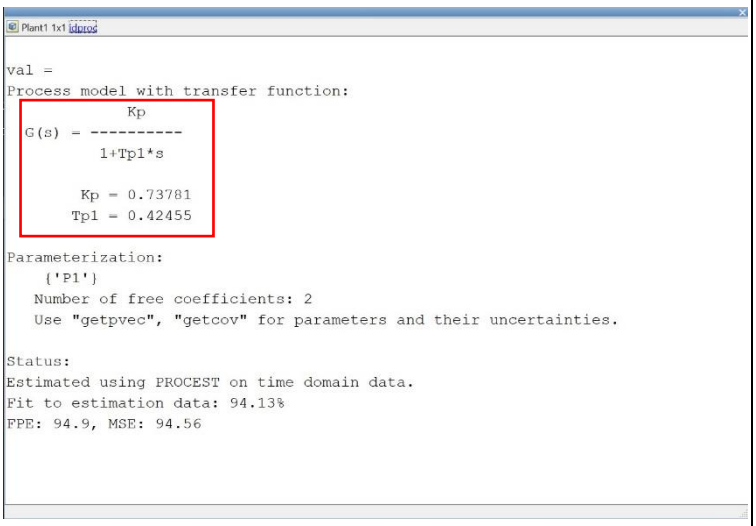


Figure 14: Plant 1 Process Model

Task III- Linux

Required: *Install Ubuntu Linux version 20.04.6 on WSL2(Windows subsystem for Linux),.*

Installation and Testing

Initially, navigate to the Windows Store and search for "Ubuntu 20.04.6". Proceed to download the application. Ensure that Virtualization is enabled from the BIOS of the computer if it's not already enabled. Upon opening the app, it will initiate the installation process and prompt the user to input their username and password.

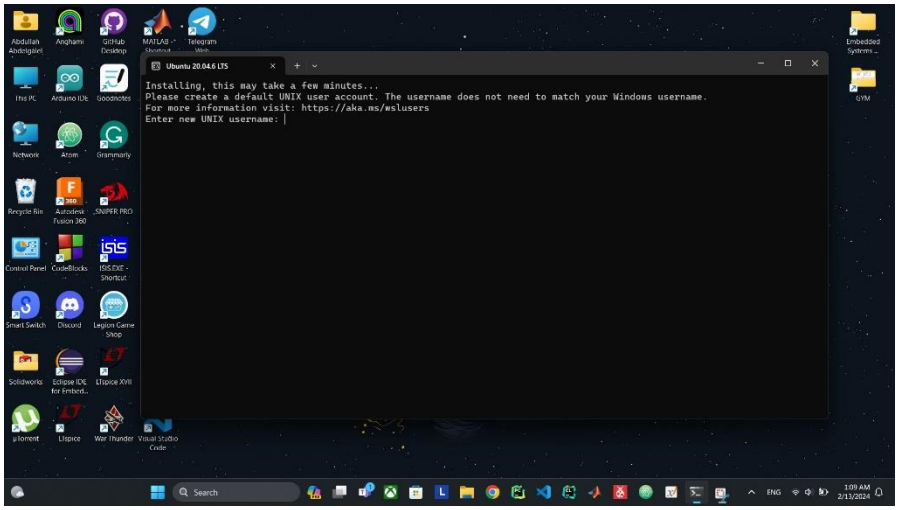


Figure 15: Username Creation Window

The images below depict a virtual machine environment created earlier on VM Workstation, running "Ubuntu 20.04.03".

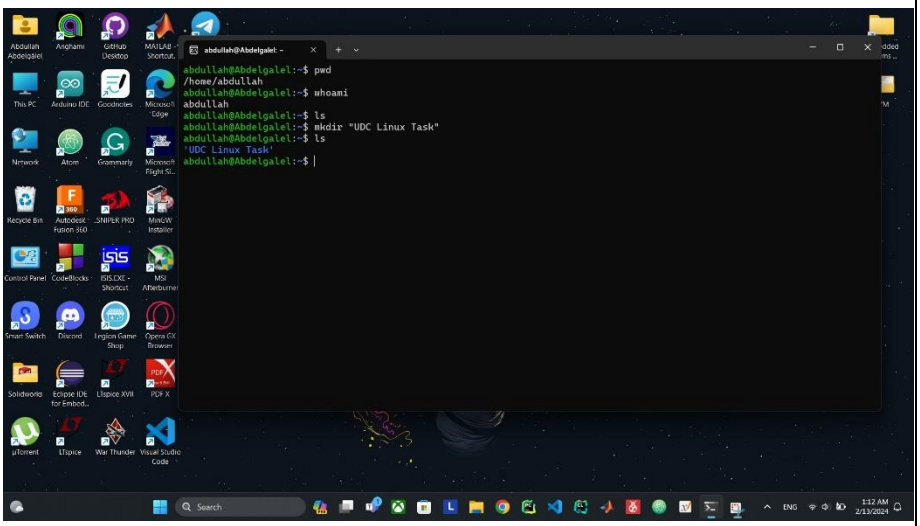


Figure 16: Some Linux Terminal Commands

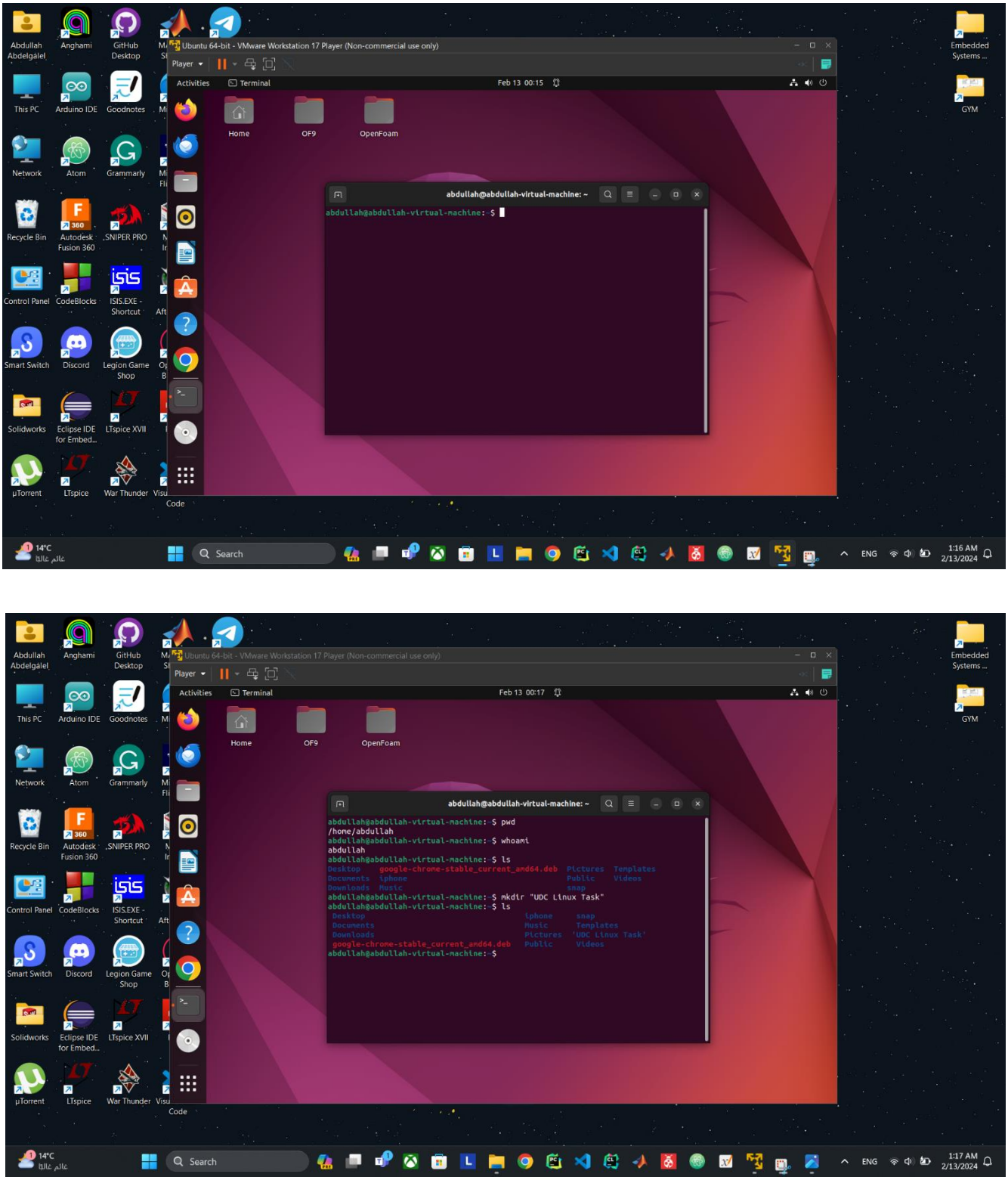


Figure 17: Ubuntu OS with some Terminal Commands