Adaptive Control of DC Motor Angular Position with Full State Measurable

Student : Widya Ageng Setya Tutuko

Table of Contents

List of Tables

List of Figures

ABSTRACT

This project is to design DC Motor's angular position control using adaptive law. The DC Motor system has two sensors to measure angular position and angular velocity which both variables can be treated as system states to construct the required adaptive law. The system parameter obtained in the calibration, together with formularized adaptive law to ensure that the system is Lyapunov stable, are then simulated in Simulink to obtain the overview of performance of controlled system and find the best starting adaptive gain Γ and Q matrix in the hardware test. Due to the presence of additional dynamic variable (noises, delay, etc), the adaptive gain Γ and Q matrix are fine-tuned to reduce the effect of such variables and drive the DC motor to desired angular position as set by reference angular position/reference signal.

I. System & Calibration

A. DC Motor Configuration and Model

In this project, the controlled plant/system is a DC Motor with integrated angle position (potentiometer) and angle velocity (tachometer) sensors. Readings of the two sensors are in volts and affected by sensor gain which needs be determined or calibrate and used to obtain the actual readings by reversing the gain actions. Voltage outputs of sensors pass through NI data acquisition system with some noise to be expected. Control signal is generated through graphical codes in connected LabVIEW system which then passes through DAC and excitation port physically towards the motor system as motor drive voltage. To better illustrate, Figure 1 below shows the setup of DC motor system.

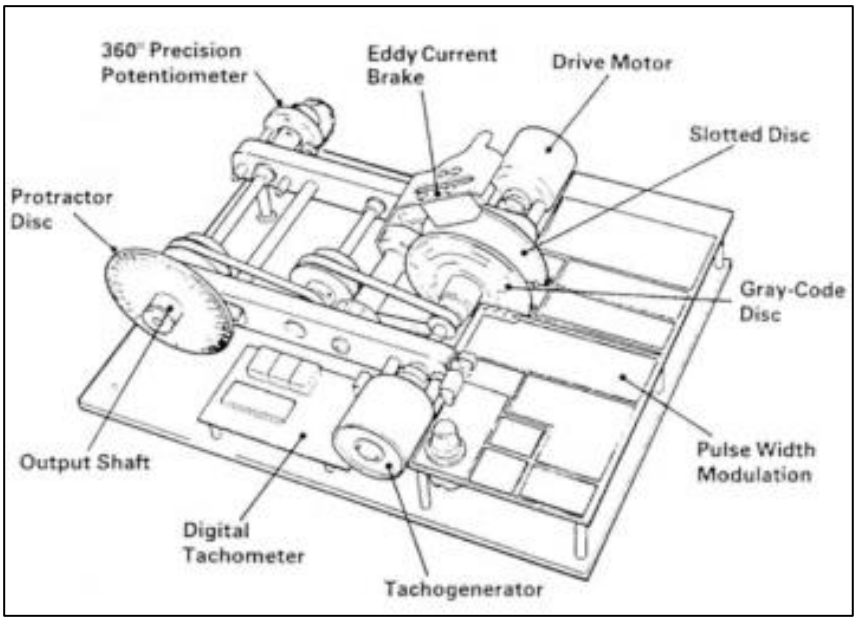


Figure 1 Physical System of Controlled DC Motor

The motor system is approached to be a first order system with DC motor voltage as system input and the angular velocity as an output. This angular velocity can then be used to determine the angular position by integrating it over time in functional block. The system block is depicted in Figure 2 below to enhance the understanding of the system in control perspective.

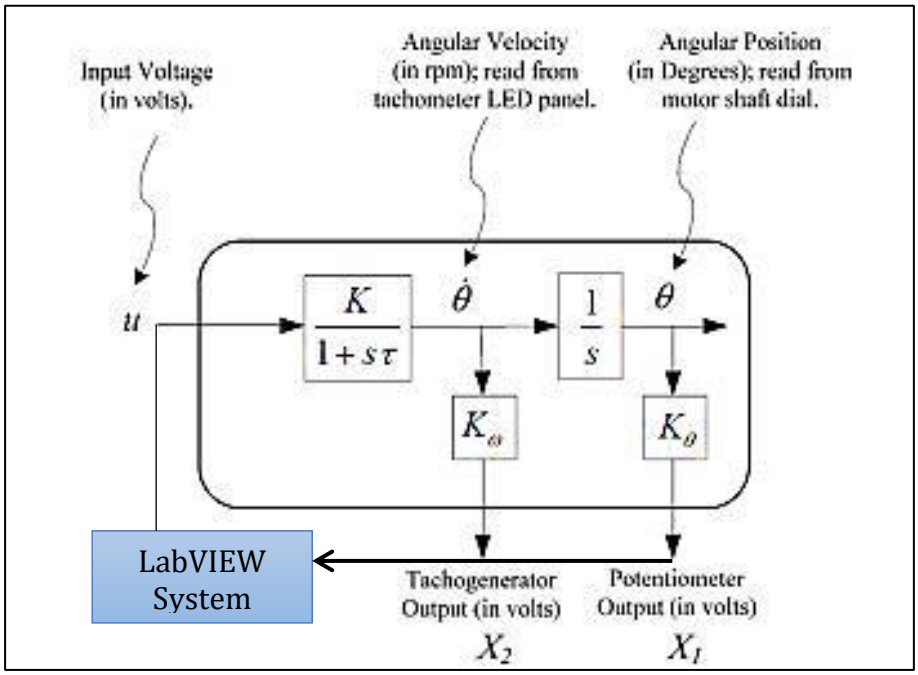


Figure 2 Simplified Diagram of DC Motor System

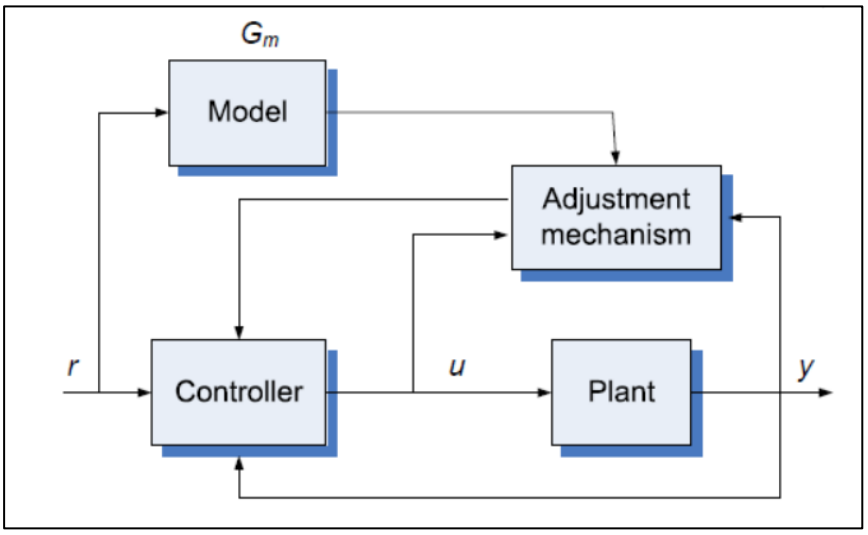


Figure 3 Block Diagram of DC Motor System with Adaptive Control within LabVIEW system

The graphical codes in LabVIEW contain functional blocks with each of them represent the reference model dynamic, adaptive controller gain block, and control signal & reference signal block. These blocks will be presented in the later chapter when required parameters are entered.

B. Sensor Gain Calibration

As elaborated earlier, the sensor readings are affected by scaling from motional units (rad & rad/sec) to volts. In order to reverse-scaling these volt readings, each sensor gains (K_ω and K_θ) are to be determined by statistic fitting. 'Polyfit' function built-in in MATLAB is utilized in this project to linearize the two-dimensional data in the sense of least-squares. The calibration of both gains will be explained in the two following sub chapters.

i. Angular Position Sensor Gain (K_θ) Calibration

Following two-dimensional data represent X_1 and angular readout (in radiant) from motor shaft dial to determine K_θ.

Shaft Dial (degree)	Shaft Dial (radiant)	X_1 (Volts)
258	4.5029	0
288	5.0265	0.84
318	5.5501	1.734
348	6.0737	2.642
378	6.5973	3.516
408	7.1209	4.445

Table 1 Calibration data to determine potentiometer gain (K_θ)

Assuming that potentiometer operates in linear region, least square linear equation can be obtained from the data above through linear regression. MATLAB has built-in function that allow linear equation to be obtained instantly by inputting the two-dimensional data given above. Snippet of codes to perform linear regression is given in Figure 4 below.

```
%Ktheta calibration using angle position, shaft dial
pos_degree = [258 288 318 348 378 408];
pos_radiant = pos_degree/(180/pi); % unit angle position to be used
in the control algorithm structure
x1 = [0 0.84 1.734 2.642 3.516 4.445]; % Labview readouts
Fittheta = polyfit(pos_radiant,x1,1);
Ktheta = Fittheta(1); % its reciprocal to be entered into labview
```

Figure 4 Polyfit (with degree 1) function to perform linear regression

Codes above generate an 1x2 array containing the gradient and y – axis intersecting point. In this case, we map angle position in radiant as x-axis and potentiometer output

voltage X_1 as y-axis. The generated fitting line and datapoint are displayed in Figure 5 below:

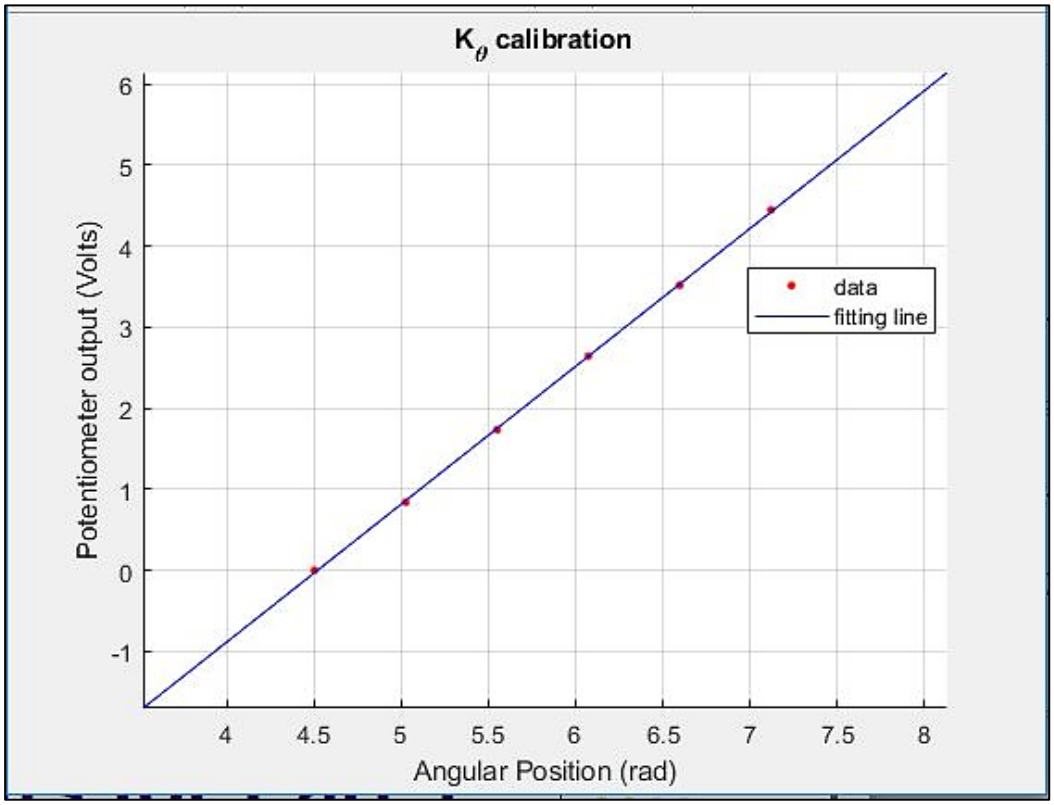


Figure 5 Linear regression of angular position vs potentiometer output

Fitting line's gradient in the figure above is the gain of potentiometer K_θ which is around 1.7004 (Volts/rad). It is understood that potentiometer output can get saturated in ADC circuitry of LabVIEW module. However, in this system, potentiometer outputs of all possible angular position are in the linear region (discrete ladder) of ADC to ensure saturation does not occur since the potentiometer is 360^o degree precision type.

It is important to check that potentiometer output are not saturated for some angular position and observe no voltage is excited to drive the motor when shaft-dial is rotated to certain angular position to obtain potentiometer reading points to construct data that is to be linearly regressed later. K_θ that we obtained is used to reverse-scaling the reading back to radiant in the control structure (LabVIEW to multiply potentiometer output with reciprocal $1/K_\theta = 0.5881$ rads/Volt).

ii. Angular Velocity Sensor Gain (K_ω) Calibration

Using part I of LabVIEW setup, DC input voltage can be excited to the motor to obtain the tachogenerator gain K_ω. The control voltage u will drive the DC motor to certain speed and direction depending on amplitude and the sign of control voltage. The two-dimensional data Table 2 below were obtained during calibration. Angular velocity (in rpm) obtained from LED digital tachometer mounted on the module while X_2 is the tachogenerator voltage output. Angular velocity in rpm has to be converted into rad/sec to align with the angular position unit in order to have consistent units in all linear fitting.

LED readout (rpm)	LED readout (rads/sec) x − axis	X_2 (Volts) y − axis
-162	-16.9646	-3.39
-104	-10.8909	-2.17
-46	-4.8171	-0.95
0	0	0
47	4.9218	0.95
104	10.8909	2.12
162	16.9646	3.39

Table 2 Calibration data to determine tachogenerator gain (K_ω)

The resulting fitting line is depicted in Figure 6 below.

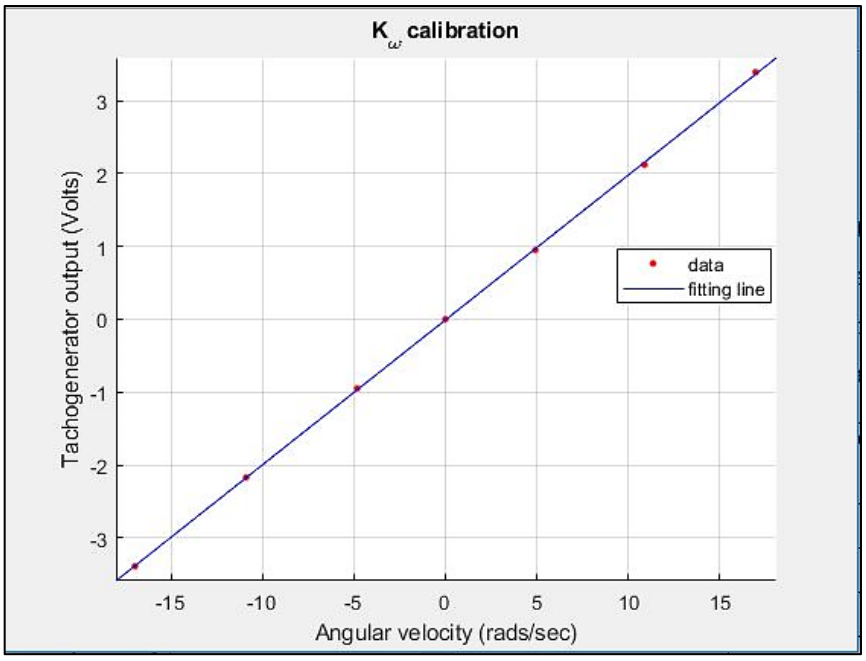


Figure 6 Linear regression of data from LED tachometer and tachogenerator output voltage

Gradient of fitting line above is around $K_\omega = 0.1988$ (Volts/rad/sec). Hence, similar to potentiometer gain, LabVIEW will use the gradient value to reverse-scale the tachogenerator reading back to rads/sec in the control system (reverse gain around $1/K_\omega = 5.0308\, rads/sec/Volt$).

iii. Determination of Static Gain K_p and Rise Time τ of DC Motor System.

The experiment done to determine K_ω in earlier subchapter is the same experiment to characterise first order system parameters of DC motor. As we approximate that DC Motor System with input voltage as input and angular velocity as output is a first order system, the follow-ups will be to determine the static gain and rise time.

Ensuring that LED readout reaching stable number is important in order to be able to obtain static gain as it tells us the ratio of output/input once the system reaches steady state. That is represented by K_p in first order model shown in Figure 2 earlier. A table consists of LED readout and input voltage (Volts) is presented below.

Input Voltage (Volts) $x-$ axis	LED readout (rpm)	LED readout (rads/sec) $y-$ axis
-3	-162	-16.9646
-2	-104	-10.8909
-1	-46	-4.8171
0	0	0
1	47	4.9218
2	104	10.8909
3	162	16.9646

Table 3 Calibration data to determine plant's static gain (K_p)

Linear fitting and calibration points are given in the Figure 7 below. The gain K_p derived from the line gradient below is around 5.5389(rad/sec/Volts). The rise time τ can be determined by measuring the time difference between times when motor starts running (assuming motor is started from rest position, or 0 rpm initial condition) to $(1 - e^{-1}) = 63.2\%$ of steady state value. The illustration of rise time is displayed in Figure 8. Empirically, through step test, it is around 0.31 seconds.

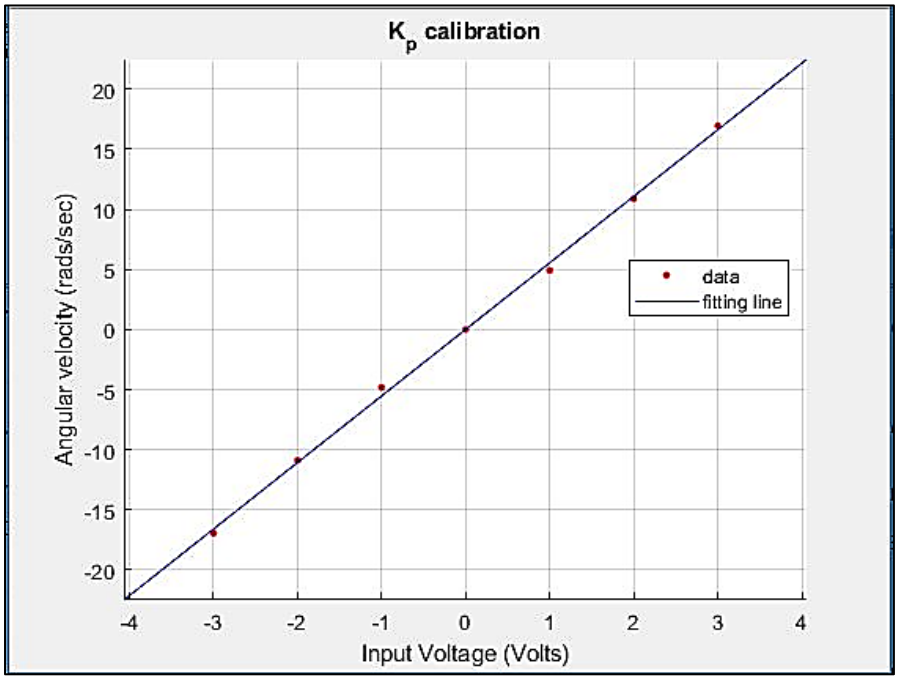


Figure 7 Linear fitting of static gain calibration data

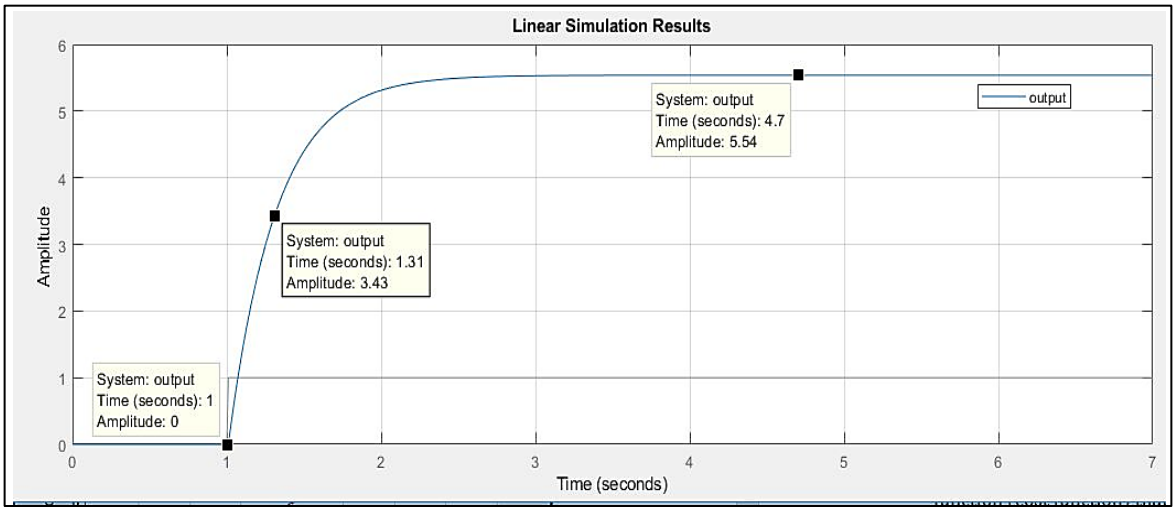


Figure 8 Step response to empirically determine rise time (1x time constant)

To summarize the findings in chapter I (calibration), below are parameter that has been obtained which will be frequently used both for control law construction and simulation.

K_θ	K_ω	K_p	τ
1.7004 (Volts/rad)	0.1988 (Volts/rad/sec)	5.5389 (rad/sec/Volts)	0.31 seconds

Table 4 Parameter obtained through calibration

II. Adaptive Controller Design with Full State Measurable

A. The State-Space Model of DC Motor System

The two states (angular position and velocity) of the system are accessible and therefore we can construct the adaptive law with all states being measurable.

From the approximated transfer function in Figure 2, we have following equations:

$$\frac{\theta}{u} = \frac{K_p}{s(\tau s + 1)} \ldots\ldots (1)$$

Let $x_1 = \theta$ (the angular position) and $x_2 = \dot{\theta} = s\theta = \omega$ (assuming zero initial condition, then we will have following equations and its representation in state-space model:

$$s^2\theta\tau + s\theta = K_p u \equiv s x_2 + \frac{x_2}{\tau} = \frac{K_p u}{\tau}$$

$$\dot{x_2} = -\frac{1}{\tau}x_2 + \frac{K_p}{\tau}u \ldots\ldots (2)$$

$$\dot{x_1} = x_2 \ldots\ldots\ldots (3)$$

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ 0 & -\frac{1}{\tau}\end{bmatrix}}_{A_p}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{\frac{K_p}{\tau}\begin{bmatrix}0\\ 1\end{bmatrix}}_{gB} u \ldots\ldots (4)$$

Substituting the numerical values that have been obtained from calibration part, we have the state space model of the specific DC motor system that was used in the experiment:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ 0 & -3.2258\end{bmatrix}}_{A_p}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{17.8675\begin{bmatrix}0\\ 1\end{bmatrix}}_{gB} u \ldots\ldots (5)$$

Since our interest is to control the angular position, we can set the output matrix C_p to be as follow:

$$y = \underbrace{[1 \quad 0]}_{C_p}\begin{bmatrix}x_1\\ x_2\end{bmatrix} \ldots\ldots\ldots (6)$$

B. Reference 2nd Order System

As we have obtained the state-space of the model, we will need a reference model that generates reference output y_m and generate an output error that we can use to construct adaptive law. Let the reference 2nd order system be defined:

$$H_m(s) = \frac{\theta}{r} = \frac{\omega_m^2}{s^2 + 2\xi\omega_m s + \omega_m^2}$$

$$s^2\theta + 2\xi\omega_m s\theta + \omega_m^2\theta = \omega_m^2 r$$

$$\dot{x_{m2}} = -2\xi\omega_m x_{m2} - \omega_m^2 x_{m1} + \omega_m^2 r$$

In the state-space form, with 'm' subscript indicates it's a reference state,

$$\begin{bmatrix} \dot{x_{m1}} \\ \dot{x_{m2}} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -\omega_m^2 & -2\xi\omega_m \end{bmatrix}}_{A_m} \begin{bmatrix} x_{m1} \\ x_{m2} \end{bmatrix} + \underbrace{\omega_m^2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{g_m B} r \quad \ldots (7)$$

$$y_m = \underbrace{[1 \quad 0]}_{C_m} \begin{bmatrix} x_{m1} \\ x_{m2} \end{bmatrix} \ldots (8)$$

The reference signal r is set to be a step waveform with proper period to ensure enough settling window for the output (angular position). With no overshoot on angular position is expected, the reference system must be critically damped and respond fast enough to reach steady state value relatively quick compared to period of reference step signal.

Note that the plant's time constant is around 0.31 seconds, the reference model's time constant should be set such that the plant's output follow the dynamics of reference output. That is, reference model time constant should be set faster than plant's time constant in order for reference output reach steady state faster and give better output error for better adaptation.

For critically damped system $\xi = 1$, the time domain response (for a step input) is given by:

$$\mathcal{L}^{-1}\left\{\frac{\theta}{r} = \frac{\omega_m^2}{s(s+\omega_m)^2} = \frac{1}{s} - \frac{1}{s+\omega_m} - \frac{\omega_m}{(s+\omega_m)^2}\right\}$$

$$\theta = r - re^{(-\omega_m t)} - r\omega_m t e^{(-\omega_m t)}$$

The DC motor plant that is approximated as first order has following equation (for t equals time constant) if we sub them into 2nd order time domain response:

$$\mathrm{t} = \tau, \theta = \ 0.6321\mathrm{r} \rightarrow \mathrm{r} - 0.6321\mathrm{r} = re^{(-0.31\omega)} + 0.31r\omega e^{(-0.31\omega)}$$

Solving above equation using Newton-Raphson or another non-linear solver, we obtain that ω is around 1.3962. Let the natural frequency of reference model ω_m is twice faster, around $\omega_m = 2$,then the reference model is set with $\xi = 1; \omega_n = 2$. The state space model in equation (7),(8) reformed to be:

$$\begin{bmatrix} \dot{x_{m1}} \\ \dot{x_{m2}} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -4 & -4 \end{bmatrix}}_{A_m} \begin{bmatrix} x_{m1} \\ x_{m2} \end{bmatrix} + 4\underbrace{\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{g_m B} r \ldots . (9)$$

$$\mathrm{y_m} = \underbrace{[1 \quad 0]}_{C_m} \begin{bmatrix} x_{m1} \\ x_{m2} \end{bmatrix} \ldots (10)$$

C. Lyapunov Stability & Adaptive Law Design

Recall equation (4),(6),(7),(8) and comparing them to obtain output error $\mathrm{e_{out}} = \mathrm{y} - \mathrm{y_m}$ and state error $\mathrm{e} = \mathrm{x} - \mathrm{x_m}$, we get:

$$\dot{x} - \dot{x_m} = \begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} - \begin{bmatrix} \dot{x_{m1}} \\ \dot{x_{m2}} \end{bmatrix} = A_p x - A_m x_m + gBu - g_m Br$$

Let the control law to be in the form: $u = \theta_x^{*T} x + \theta_r^* r \ldots . (11)$,

$$\dot{e} = \left(A_p + gB\theta_x^{*T}\right)x - A_m x_m + gB\theta_r^* r - g_m Br \ldots . . (12)$$

If θ_x^{*T} and θ_r^* are tuned in such a way that matching conditions below achieved:

$$\left(A_p + gB\theta_x^{*T}\right) = A_m$$

$$g\theta_r^* = g_m \ (scalar)$$

The error equation (12) can be simplified into:

$$\dot{e} = A_m e$$

Provided that A_m is a selectable matrix, we can choose such that A_m is negative definite, i.e.: eigenvalues of matrix A_m are negative. However, in most practical cases, θ_x^{*T} and θ_r^* are not easily determined as A_p and g of original plant are not accurately known. This leads to adaptation of control law and instead of taking the form in equation (11), it will have the following form of function of time:

$$u = \theta_x^T x + \theta_r r \ldots\ldots (13)$$

Substitute this into error equation (12),

$$\dot{e} = \left(A_p + gB\theta_x^T\right)x - A_m x_m + gB\theta_r r - g_m B r$$

Let the parametric error $\Phi = \begin{bmatrix}\Phi_x \\ \Phi_r\end{bmatrix} = \begin{bmatrix}\theta_x - \theta_x^* \\ \theta_r - \theta_r^*\end{bmatrix}$, then the error equation changes into:

$$\dot{e} = \left(A_p + gB\theta_x^{*T}\right)x - A_m x_m + gB\theta_r^* r - g_m B r + gB\Phi^T \begin{bmatrix}x \\ r\end{bmatrix}$$

Using the matching condition obtained previously, the new error equation now has included a dynamic of parametric error that evolves with time:

$$\dot{e} = A_m e + gB\Phi^T \begin{bmatrix}x \\ r\end{bmatrix} \ldots\ldots (14)$$

To ensure that state error e is bounded and decaying to zero, we need to construct Lyapunov function that is function of both state error e and parametric error Φ.

Given a candidate of such Lyapunov function:

$$\mathrm{V}(\mathrm{e}, \Phi) = e^T P e + |g|\Phi^T \Gamma^{-1}\Phi \ldots\ldots (15)$$

Note that Φ is 3x1 parametric error since we have two states and one reference signal. And Γ is positive definite symmetrical matrix, while P is positive definite & symmetric matrix to guarantee that Lyapunov function is a quadratic function and thus $\mathrm{V}(\mathrm{e}, \Phi) \geq 0$. To further see that Lyapunov function above is non-decrescent and both upper-bounded and lower-bounded by a scalar function, the following checks and simulation are performed.

Let the lower-bound scalar function a(x) is defined as :

$$\mathrm{a}(\mathrm{e}, \Phi) = (e^T P e + |g|\Phi^T \Gamma^{-1}\Phi)\left(1 - e^{-e^T e - \Phi^T \Phi}\right),$$

letting vector x consist of state error and parametric error values.

While for upper-bound scalar function b(x) is set:

$$\mathrm{b}(\mathrm{e}, \Phi) = (e^T P e + |g|\Phi^T \Gamma^{-1}\Phi)\left(1 + e^{-e^T e - \Phi^T \Phi}\right)$$

It is clear that, $a(e,\Phi) \leq V(e,\Phi) \leq b(e,\Phi)$ and both scalar functions are non-decrescent too. Simulation of these functions below show that indeed Lyapunov function above is bounded (upper and lower) by non-decrescent scalar functions.

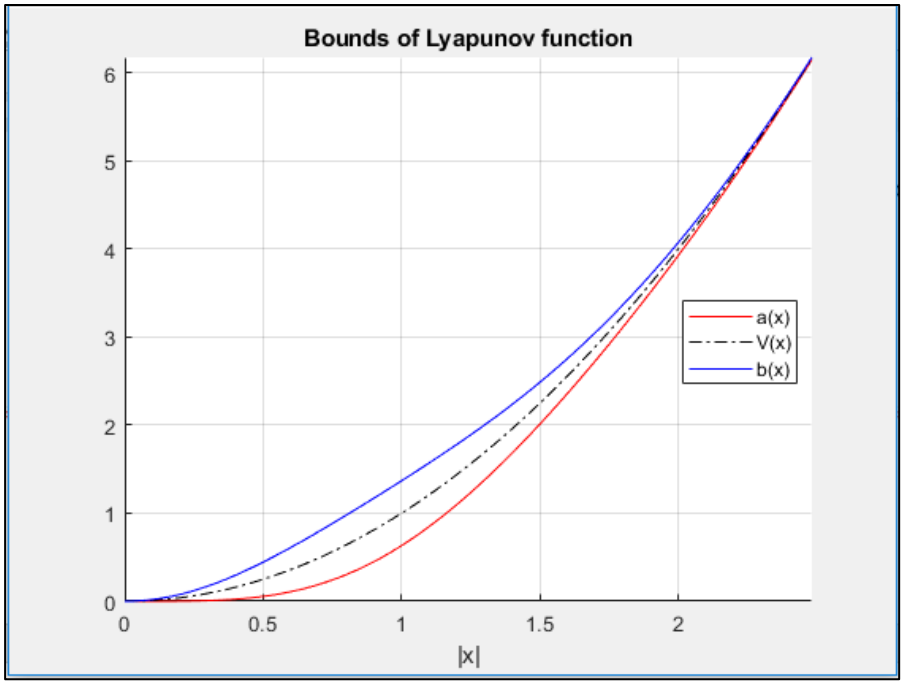


Figure 9 Plot of bound functions of Lyapunov function V(x)

It is also noted that Lyapunov function is radially unbounded, i.e.: $V(e,\Phi) \to \infty$ as $e \to \infty$ and $\Phi \to \infty$. Now the only check left is ensuring that time gradient of Lyapunov function is negative (decrescent with time). Differentiating equation (15) with respect to time, we obtain:

$$\dot{V} = 2e^{T}P\dot{e} + 2|g|\Phi^{T}\Gamma^{-1}\dot{\Phi}$$

Recall error equation (14) and sub into above equation:

$$\dot{V} = 2e^{T}PA_m e + 2e^{T}PgB\Phi^{T}\begin{bmatrix} x \\ r \end{bmatrix} + 2|g|\Phi^{T}\Gamma^{-1}\dot{\Phi}$$

$e^{T}PA_m e$ is a scalar, thus its transpose is its equivalent and equation above changes into:

$$\dot{V} = e^{T}(PA_m + A_m^{T}P)e + 2e^{T}PgB\Phi^{T}\begin{bmatrix} x \\ r \end{bmatrix} + 2|g|\Phi^{T}\Gamma^{-1}\dot{\Phi}$$

$PA_m + A_m^{T}P$ is ARE equation which equalizes $-Q$ with Q being positive definite matrix also. Thus we have,

$$\dot{V} = -e^T Qe + 2e^T PgB\Phi^T \begin{bmatrix} x \\ r \end{bmatrix} + 2|g|\Phi^T \Gamma^{-1}\dot{\Phi} \ldots . (16)$$

As analysed before, $\dot{V}$ should be negative to ensure Lyapunov function is decaying towards zero, implying that state and parametric error decay to zero. The first term $-e^T Qe$ already guarantees negative value, thus the remaining terms should be designed in such a way that it cancels each other:

$$2e^T PgB\Phi^T \begin{bmatrix} x \\ r \end{bmatrix} + 2|g|\Phi^T \Gamma^{-1}\dot{\Phi} = 0 \rightarrow 2|g|\Phi^T \Gamma^{-1}\dot{\Phi} = -2e^T PgB\Phi^T \begin{bmatrix} x \\ r \end{bmatrix}$$

Note that $\Phi^T \begin{bmatrix} x \\ r \end{bmatrix}$, $e^T PB$ and g are scalar and their position can be interchanged, with that we have:

$$g = sign(g)|g| \rightarrow |g|\Phi^T \Gamma^{-1}\dot{\Phi} = -sign(g)|g|\Phi^T \begin{bmatrix} x \\ r \end{bmatrix} e^T PB$$

Front term $|g|\Phi^T$ cancels out, leaving:

$$\Gamma^{-1}\dot{\Phi} = -sign(g) \begin{bmatrix} x \\ r \end{bmatrix} e^T PB \rightarrow \dot{\Phi} = -sign(g)\Gamma \begin{bmatrix} x \\ r \end{bmatrix} e^T PB \ldots . (17)$$

It implies that if the dynamic of parametric error is defined using equation (17), the time gradient of Lyapunov function candidate above is stable.

Thus, $V(e, \Phi)$ is decrescent with respect to time indicating that it is bounded. Boundedness of $V(e, \Phi)$ further prove that state error e and parametric error Φ are bounded, which also implies that states x and controller gain (θ_x and θ_r) are bounded. As reference signal r is user-defined, thus ė is bounded as deduced by equation (14): $\dot{e} = A_m e + gB\Phi^T \begin{bmatrix} x \\ r \end{bmatrix}$.

From equation (16), once the last two terms are cancelled after implementing adaptive law (17), the Lyapunov function gradient is given by:

$$\dot{V} = -e^T Qe$$

Integrating both sides over a range of $t = [0, \infty)$ gives:

$$V(0) - V(\infty) = \int_0^\infty e^T Qe \, dt \leq k$$

Since V are bounded with any t. Together with gradient-boundedness implication from equation (14), we know that $\lim_{t \to \infty} e(t) = 0$. In addition to that, state error

cannot have impulses with infinite gradient as shown by example in Figure 10 below:

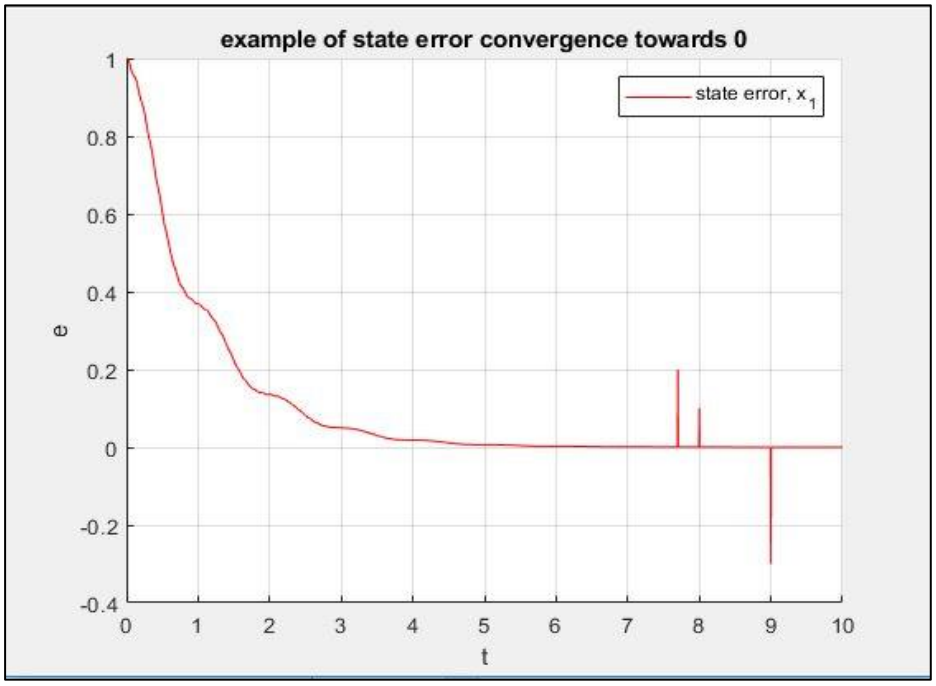


Figure 10 Impulses on state error during steady state

III. MATLAB & Simulink Simulation

A. Building Blocks and Numerical Model

To construct block diagram in Simulink, all states-space equations and adaptive law has to be in numerical values. Recollecting past information of the models, we have following sets of numerical models with some parameters obtained during calibration:

DC Motor state-space representation:

$$\begin{bmatrix}\dot{x_1}\\ \dot{x_2}\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ 0 & -3.2258\end{bmatrix}}_{A_p}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \underbrace{17.8675\begin{bmatrix}0\\ 1\end{bmatrix}}_{gB}u$$

Reference model state-space representation:

$$\begin{bmatrix}\dot{x_{m1}}\\ \dot{x_{m2}}\end{bmatrix} = \underbrace{\begin{bmatrix}0 & 1\\ -4 & -4\end{bmatrix}}_{A_m}\begin{bmatrix}x_{m1}\\ x_{m2}\end{bmatrix} + \underbrace{4\begin{bmatrix}0\\ 1\end{bmatrix}}_{g_mB}r$$

Adaptive Law:

$$\dot{\Phi} = -sign(g)\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ r\end{bmatrix}e^T\begin{bmatrix}1.125 & 0.125\\ 0.125 & 0.1563\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix}$$

$$\dot{\Phi} = -\begin{bmatrix}x_1\\ x_2\\ r\end{bmatrix}(0.125e_1 + 0.1563e_2)$$

for Γ and Q being identity matrix and $P = lyap(Am', Q)$, solved using MATLAB built-in ARE solver. Sign(g) is positive since value of K_p calibrated earlier is positive.

From calibration parts, we have:

$1/K_\theta$	$1/K_\omega$
0.5881 (rad/Volts)	5.0308 (rad/sec/Volts)

And last, reference signal r is square wave with period 10 seconds, 50% duty cycle, 2unit peak-to-peak with offset -1. Each of the building blocks in Simulink Simulation are given below:

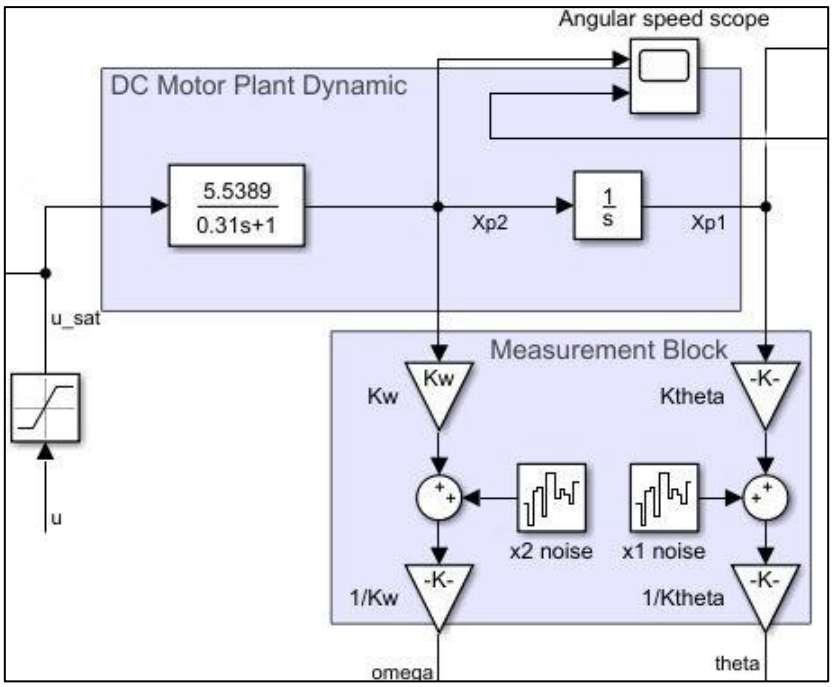


Figure 11 DC Motor plant and measurement section

Note that there are noises expected in the measurement. The PSD of this white noise can be set in Simulink interface to simulate noisy and noise-free measurement.

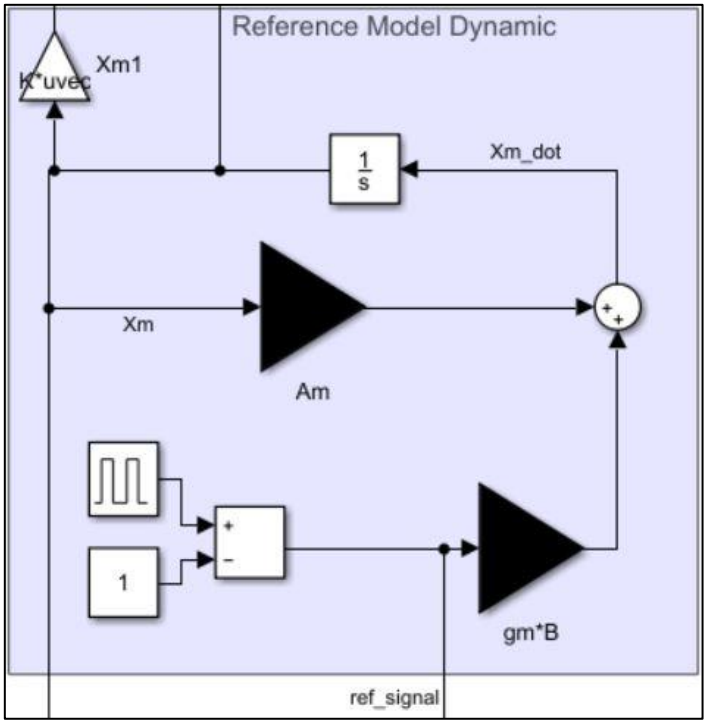


Figure 12 Reference model dynamics

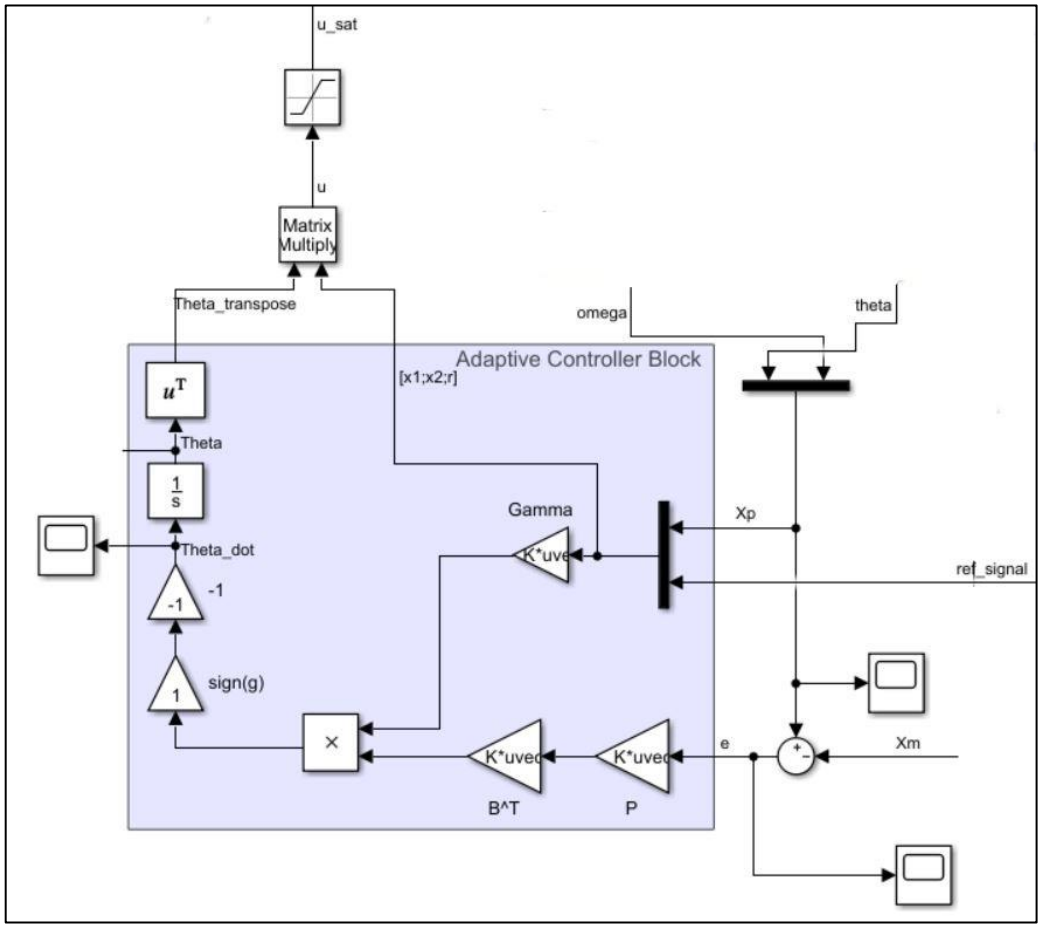


Figure 13 Adaptive Control law construction block

There is limitation on the size of control signal available in LabVIEW system. Thus, in this Simulink simulation, it is preferred to place saturation block to limit the control signal to always fall within range [-5,5].

B. Simulation Results and Discussion

Using the models and parameter above, we have following simulation results.

i. Case of $\Gamma = diag(1,1,1); Q = diag(1,1)$

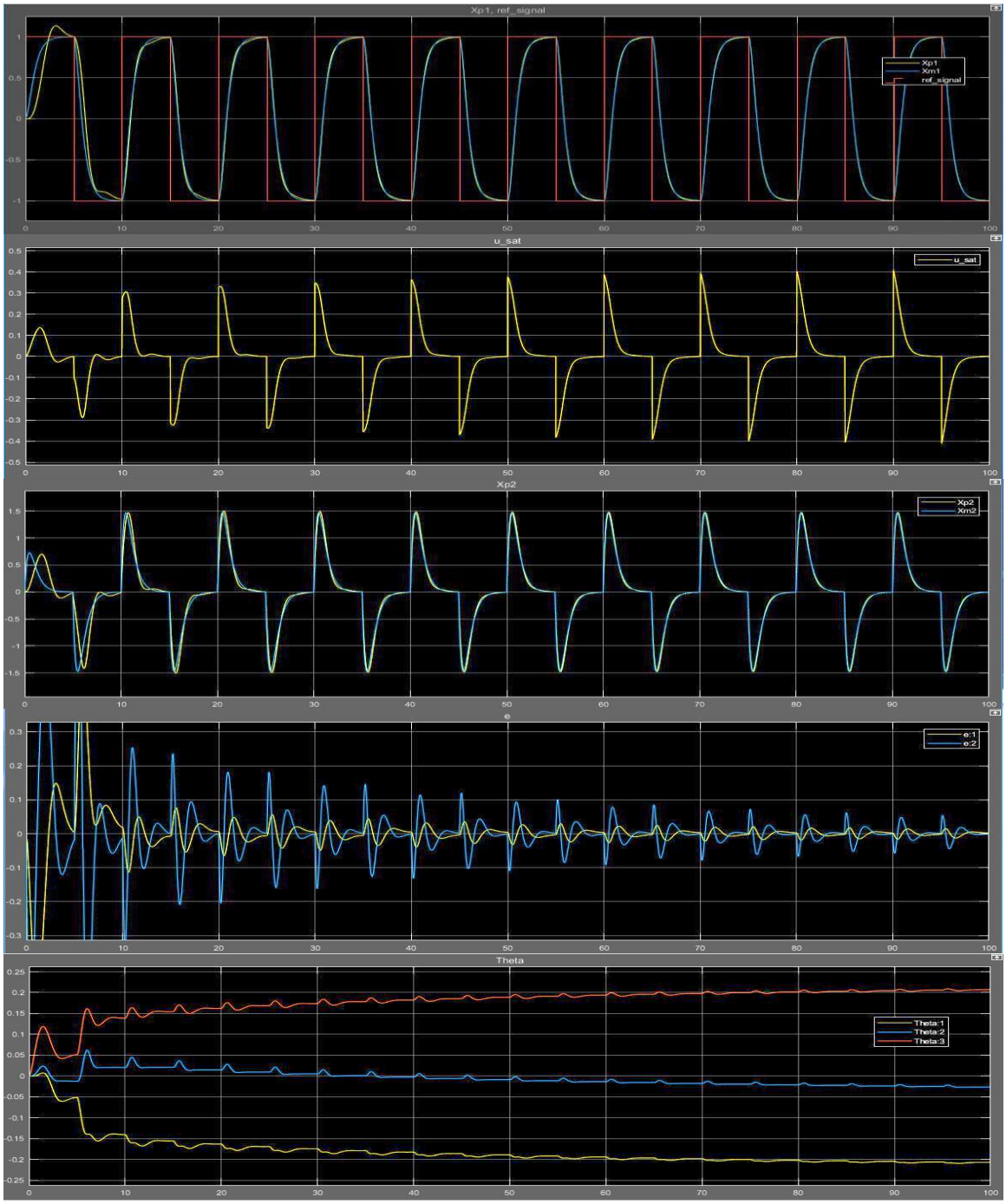


Figure 14 Top to bottom: angular position, control signal, angular velocity, state error, parameter path for $\Gamma = diag(1,1,1); Q = diag(1,1)$

Comparison of controller parameter (θ_x, θ_r) and exact controller parameter (θ_x^*, θ_r^*) are shown in figure below:

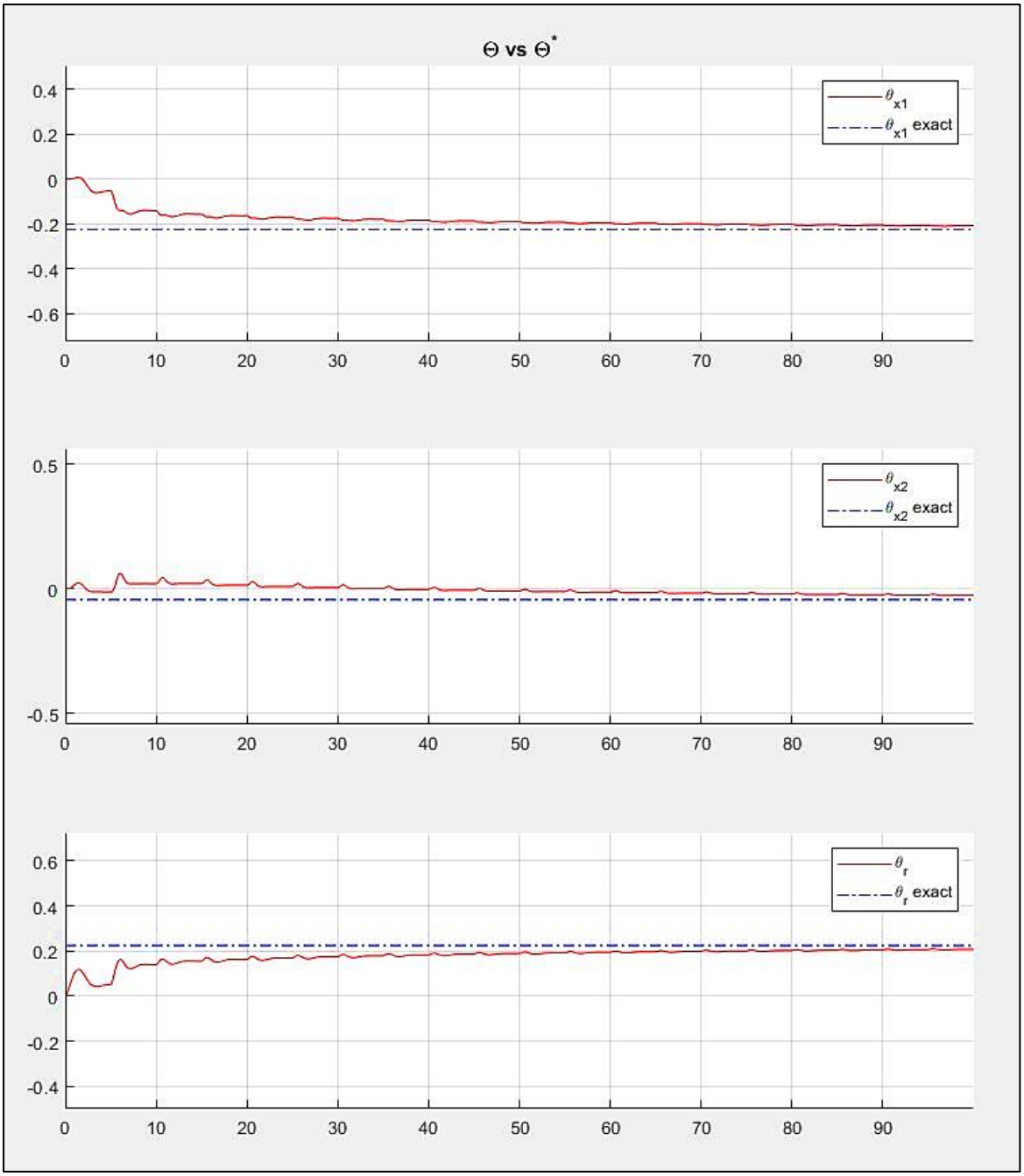


Figure 15 Comparison of exact controller parameter vs real-time controller parameter when $\Gamma = diag(1,1,1); Q = diag(1,1)$

From Figure 14, it can be observed that the adaptive controller helps the angular position and velocity to follow the same signals from reference model. The absolute value of control signal is still under saturation limit. Also note that with zero initial condition for all states and parameter, the real-time controller parameter converges

towards the exact parameter approximated from the constants obtained from matching condition, i.e.: $\theta_x^{*T} = \frac{B^T(A_m - A_p)}{gB^TB}; \theta_r^* = \frac{g_m}{g}$

ii. Case of $\Gamma = \text{diag}(10,10,10); Q = \text{diag}(1,1)$

Figure 16 Top to bottom: angular position, control signal, angular velocity, state error, parameter path for $\Gamma = diag(10,10,10); Q = diag(1,1)$

Notice that state error decays faster to zero compared to when $\Gamma = \text{diag}(1,1,1)$. However, control signal is much larger in peak-to-peak amplitude.

Following figure is the comparison between exact controller parameter vs real-time controller parameter trajectories.

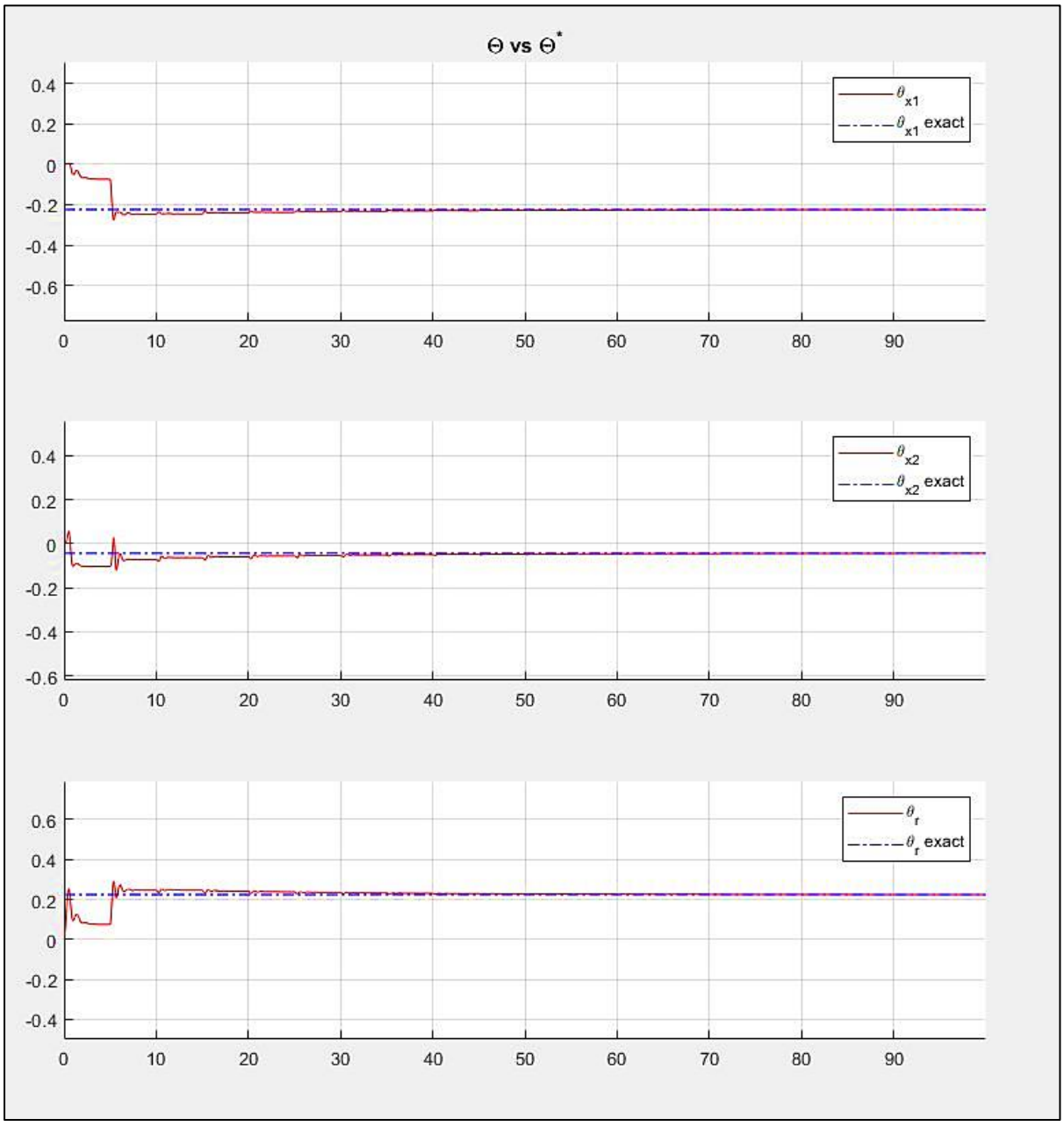


Figure 17 Convergence of controller parameter towards exact controller parameter when $\Gamma = diag(10,10,10); Q = diag(1,1)$

The larger the adaptive gain Γ, the faster the convergence of controller parameter towards exact ones. As a result, the state error reduces to zero rather quickly but at the cost of increasing the control signal size. One needs to pay attention to not set up too

large adaptive gain otherwise the saturation limit is reached and saturation of control signal will not guarantee the Lyapunov function to decrease. In addition, if the system is noisy, large adaptive gain will make noise induced become larger.

iii. Case of $\Gamma = diag(1,1,1); Q = diag(10,10)$

Figure 18 Top to bottom: angular position, control signal, angular velocity, state error, parameter path for $\Gamma = diag(1,1,1); Q = diag(10,10)$

The response is similar to previous case where $\Gamma = \text{diag}(1,1,1)$. This is because, overall gain in adaptive law of case (ii, 10Γ) and (iii, $10Q \equiv 10P$) is multiplied by 10 from the first case.

iv. Case of $\Gamma = diag(1,1,1); Q = diag(1,1)$ with Controller Parameter set at Exact Parameter values (θ^*)

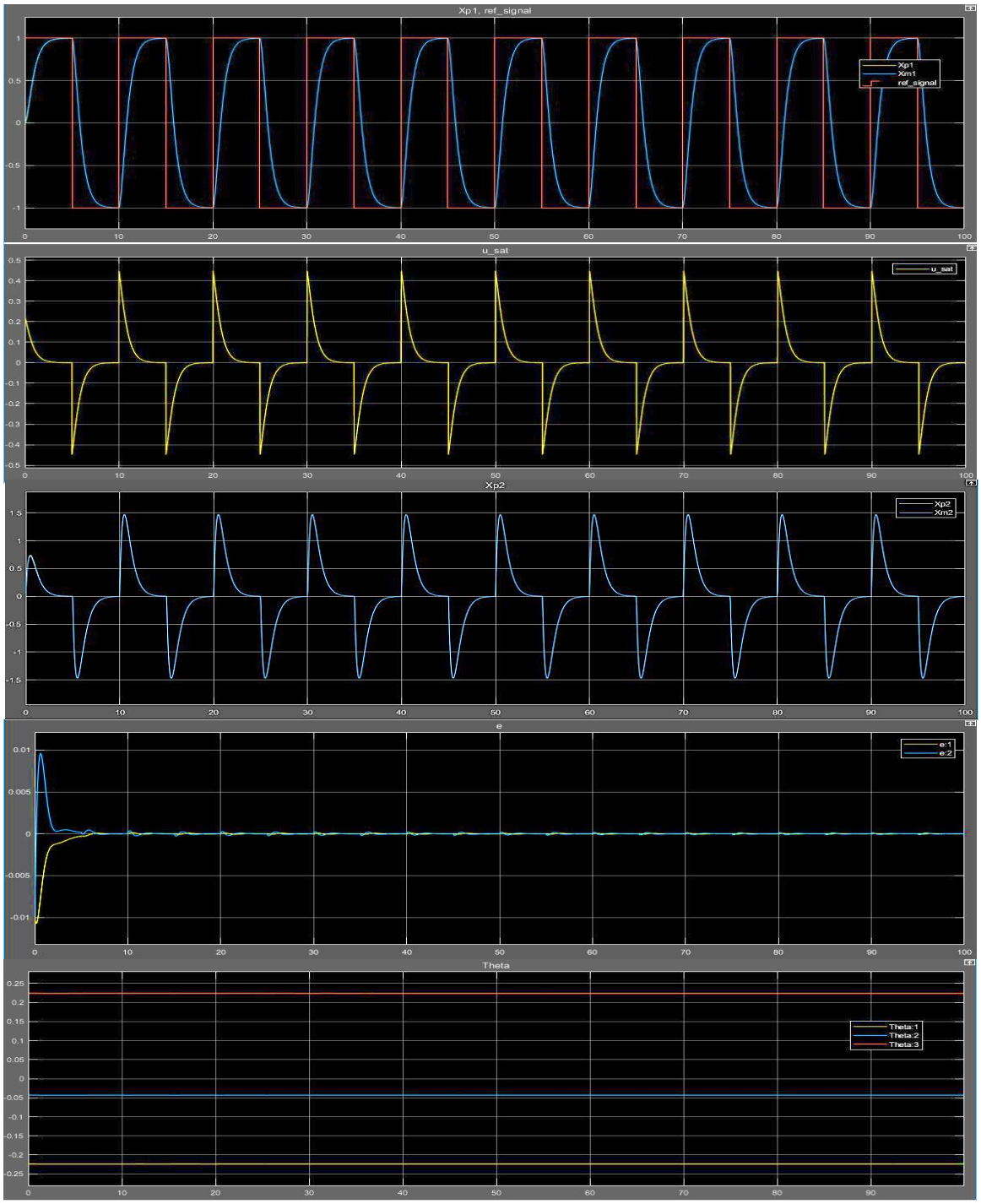


Figure 19 System response when exact parameter as initial condition

The angular position instantly converges to output of reference model unlike the response when the initial condition is zero. The controller parameter shown in the most bottom figure shows that the values are kept throughout the process, indicates that adaptation reaches steady state since the beginning.

i. Case of $\Gamma = diag(1,1,1); Q = diag(1,1)$ with Measurement Noises

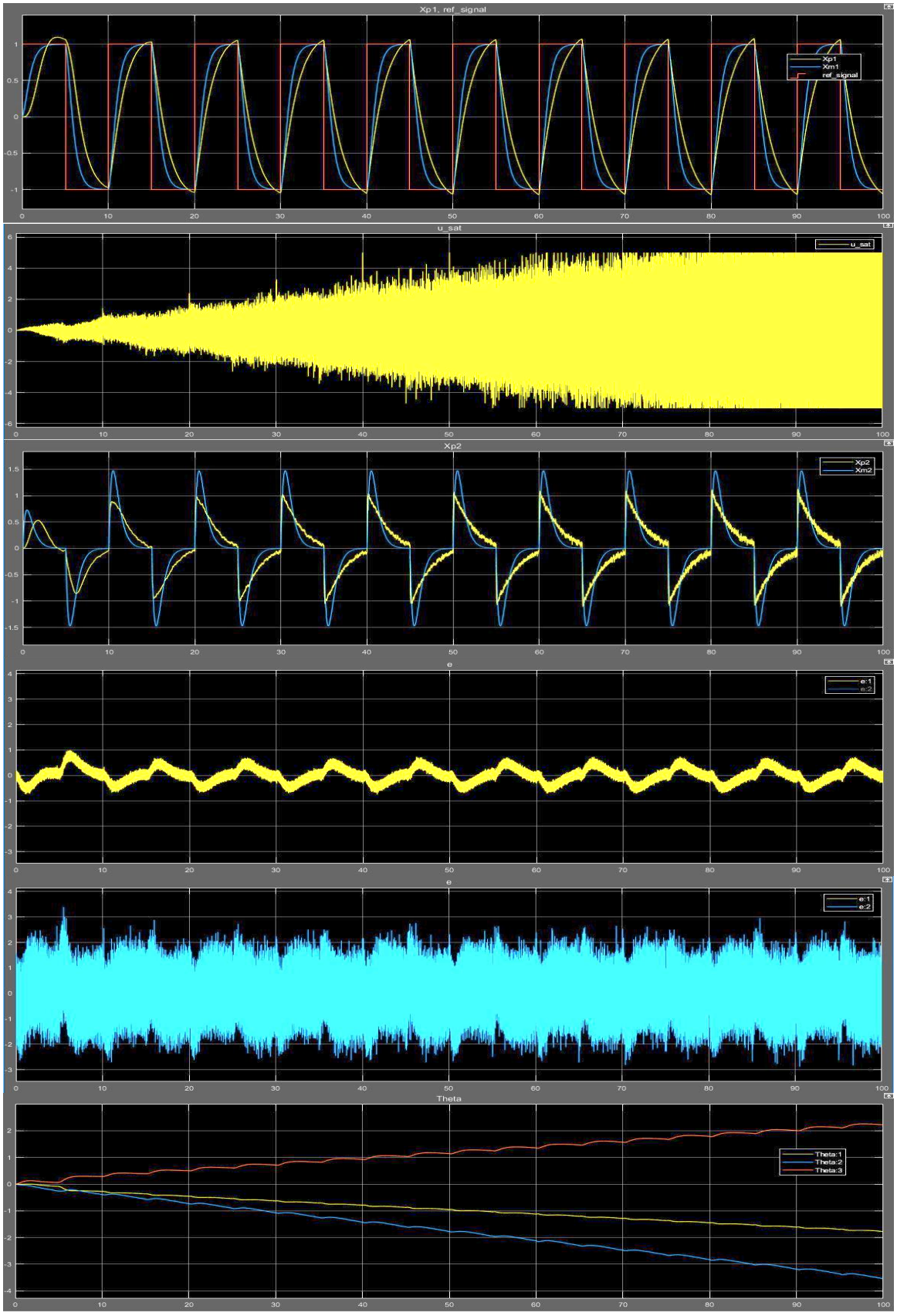


Figure 20 Top to bottom: angular position, control signal, angular velocity, positional error, velocity error, controller parameter

The noises affect the error signal since the sensing of plant's angular position and velocity are blurred by these measurement noises. As a result, the size of control signal touches saturation limit and the controller parameter magnitudes keep growing due to the fact that it is also integrating the noises:

$$\int \dot{\Phi} = \int -sign(g)\Gamma \begin{bmatrix} x \\ r \end{bmatrix} (e + \aleph)^T PB$$

To counter the effect of measurement white noises, a band pass filter might need to be implemented before the measurement used in the adaptive control law block. With the band pass filter, only noises in the system's frequencies will remain but the effect will be less severe since the wide-sense noises' width now reduced.

IV. LabVIEW Setup, Hardware Test and Discussion

In LabVIEW graphical codes, we have to register some values needed to simulate the response for different Γ and P(from different Q). Below are building blocks of LabVIEW with some parameter to be registered in fill-in boxes.

A. Reference Model Block

$$\begin{bmatrix} \dot{x_{m1}} \\ \dot{x_{m2}} \end{bmatrix} = \underbrace{\begin{bmatrix} am11 & am12 \\ am21 & am22 \end{bmatrix}}_{A_m} \begin{bmatrix} x_{m1} \\ x_{m2} \end{bmatrix} + \underbrace{\begin{bmatrix} gm1 \\ gm2 \end{bmatrix}}_{g_m B} r$$

$$\begin{bmatrix} \dot{x_{m1}} \\ \dot{x_{m2}} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 \\ -4 & -4 \end{bmatrix}}_{A_m} \begin{bmatrix} x_{m1} \\ x_{m2} \end{bmatrix} + \underbrace{4\begin{bmatrix} 0 \\ 1 \end{bmatrix}}_{g_m B} r$$

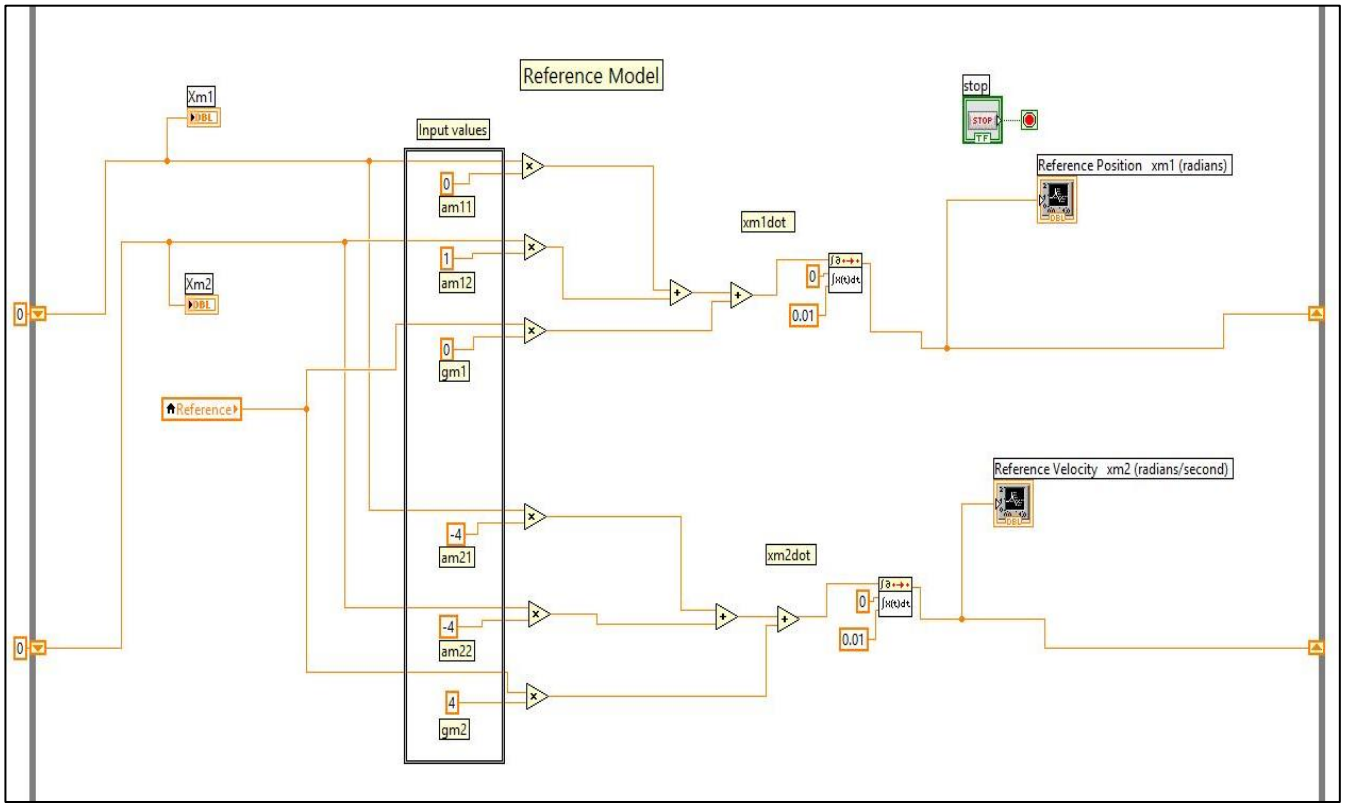


Figure 21 Reference model graphical codes in LabVIEW

B. Control Law and Reverse-Scaling Measurement Block

Control signal building block:

$$sat(u) = \begin{cases} u = 5, for\ u \geq 5 \\ u = \left([\theta_{x1} \quad \theta_{x2} \quad \theta_r] \begin{bmatrix} x_1 \\ x_2 \\ r \end{bmatrix} \right), for\ 5 < u < -5 \\ u = -5, for\ u \leq -5 \end{cases}$$

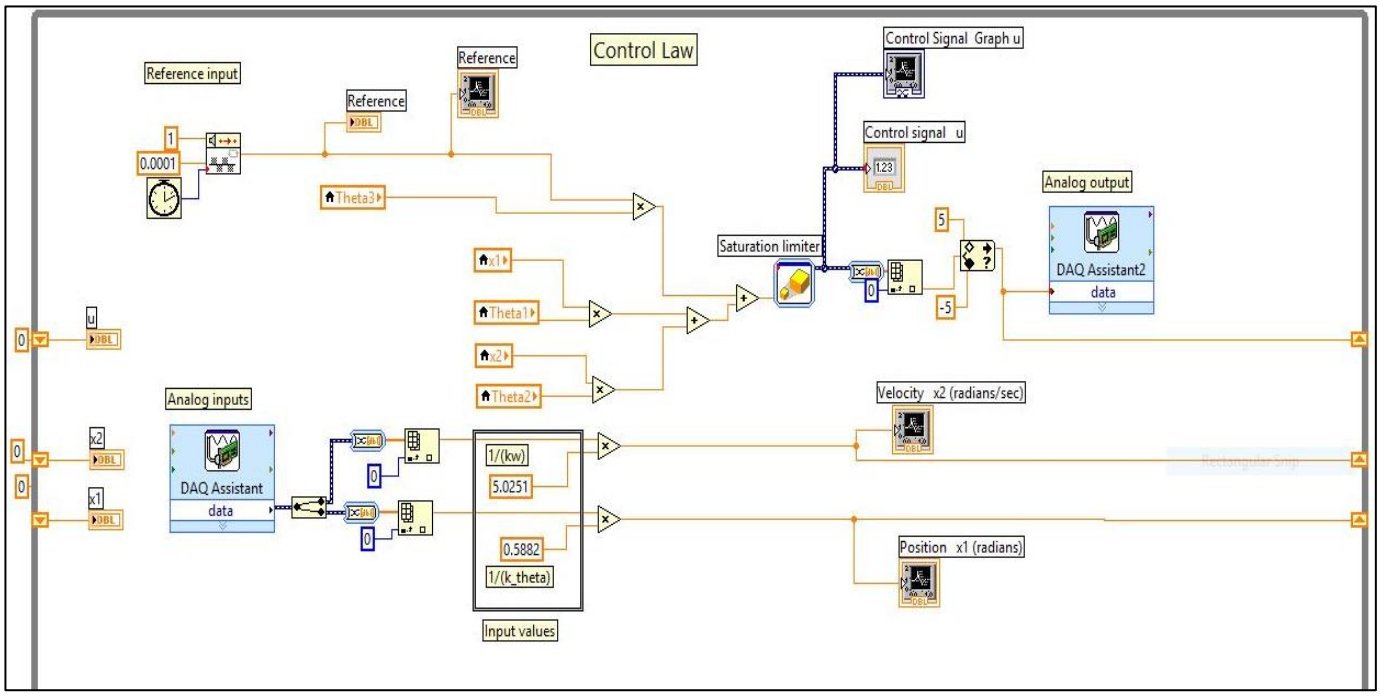


Figure 22 LabVIEW graphical code for measurement block and control law

Note the saturation limiter [-5,5] in the building block as well as the reverse scaling gain to recover the physical angular position and velocity:

$1/K_\theta$	$1/K_\omega$
0.5881 (rad/Volts)	5.0308 (rad/sec/Volts)

C. Adaptive Law Building Block

From equation (17), $\dot{\Phi} = -sign(g)\Gamma \begin{bmatrix} x \\ r \end{bmatrix} e^T PB$. The term $e^T PB$ in adaptive law can be reversed in this block since it is a scalar. $B^T P^T$ will be equal to $[0 \quad 1]\begin{bmatrix} p_{11} & p_{21} \\ p_{12} & p_{22} \end{bmatrix} = \begin{bmatrix} p_{12} \\ p_{22} \end{bmatrix}$ and obviously $e = \begin{bmatrix} x_1 - x_{m1} \\ x_2 - x_{m2} \end{bmatrix}$.

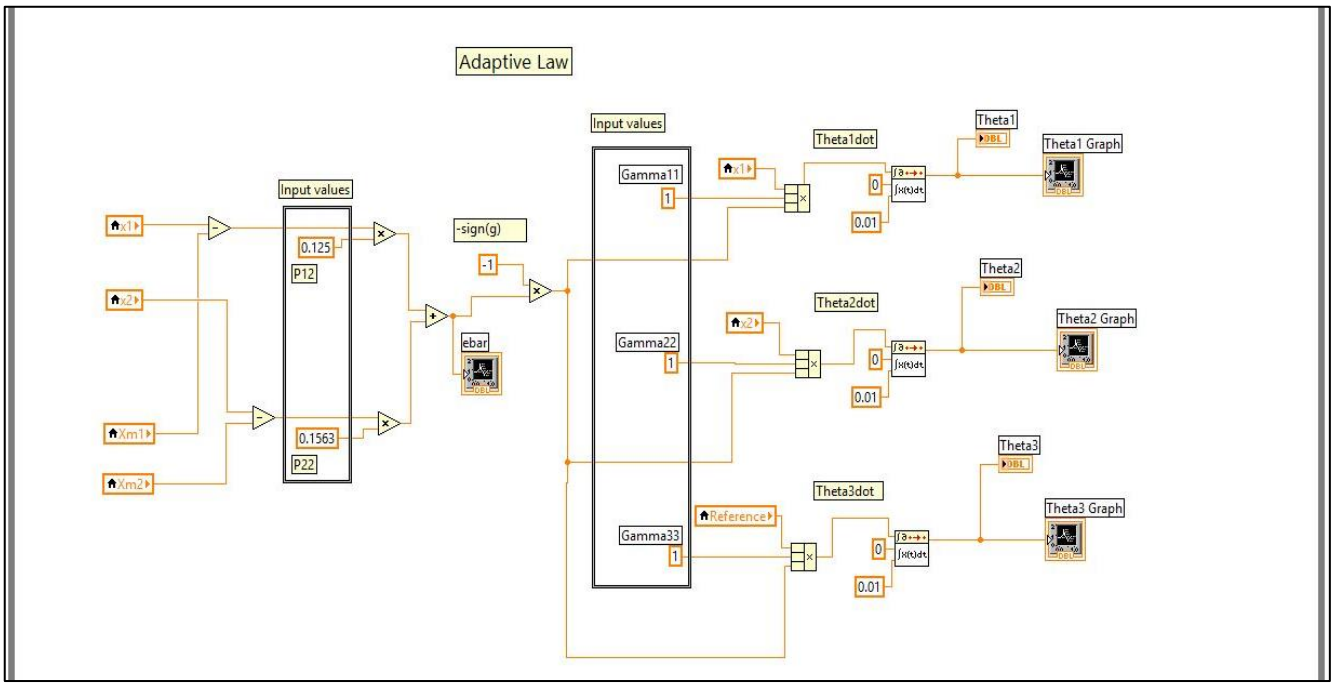


Figure 23 LabVIEW building block for Adaptive Law section

D. Hardware Test Result and Discussion

i. Case of $\Gamma = diag(1,1,1); Q = diag(1,1)$

From simulation earlier, setting up $\Gamma = diag(1,1,1); Q = diag(1,1)$ suffices to force the plant output to follow reference model as shown in Figure 14 with reasonably low size of control signal. However in the real hardware test, the plant output does not quite follow the reference output as shown in figures below:

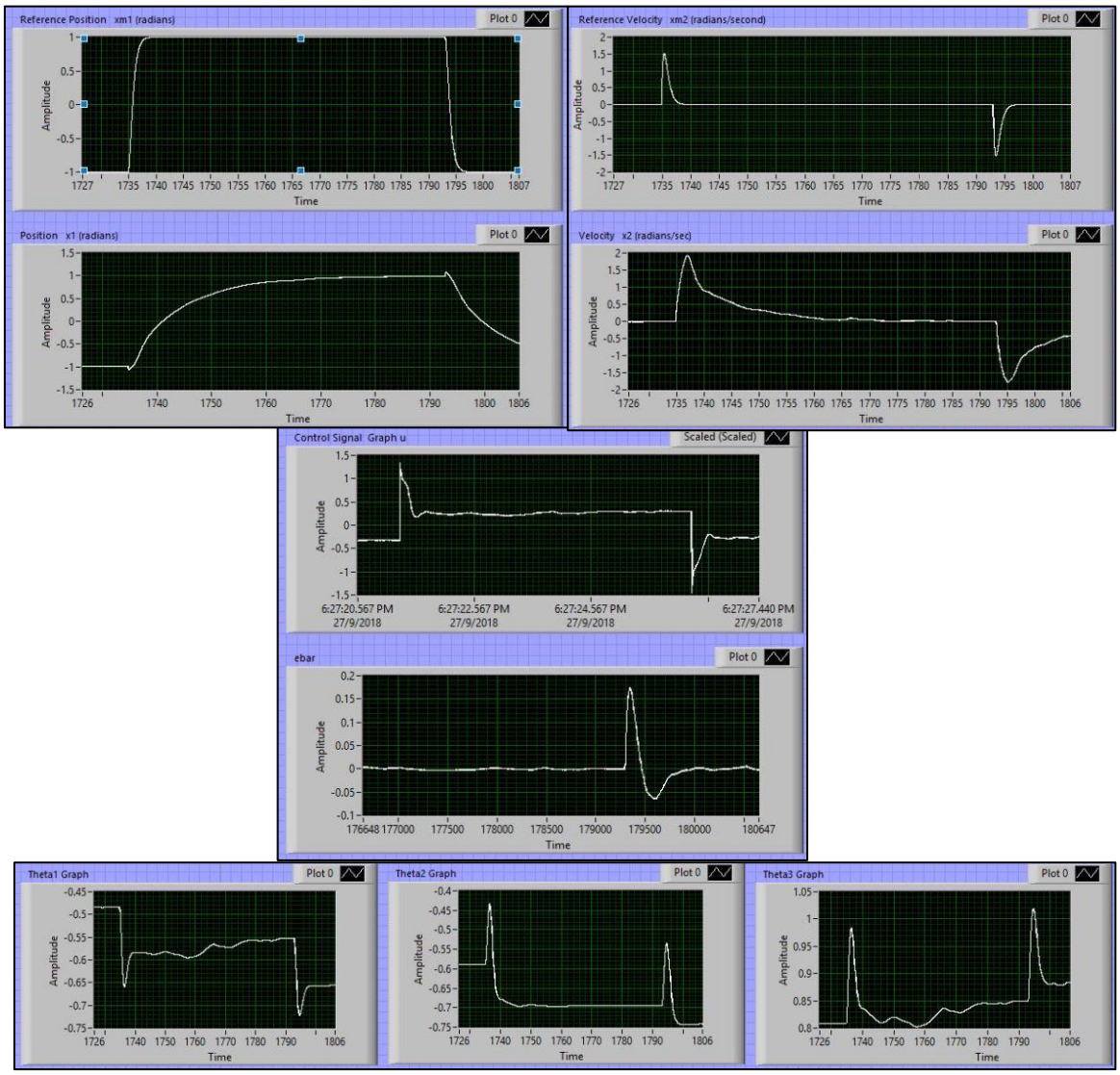


Figure 24 Test Result of setting up $\Gamma = \text{diag}(1,1,1); \text{Q} = \text{diag}(1,1)$

The control signal is much larger than the simulation as expected due to the noise in the system (could be from the motor like friction, commutation, modulation etc) or the measurement interface between the motor module and LabVIEW interface. Transient response on angular position lasts longer than in simulation. In

simulation, we did not introduce system delay that surely has, thus it appears in this real hardware test and it's not anticipated in the control law and simulation earlier.

ii. Case of $\Gamma = diag(10,10,10); Q = diag(1,1)$

For the setting given above, the results are shown as follows:

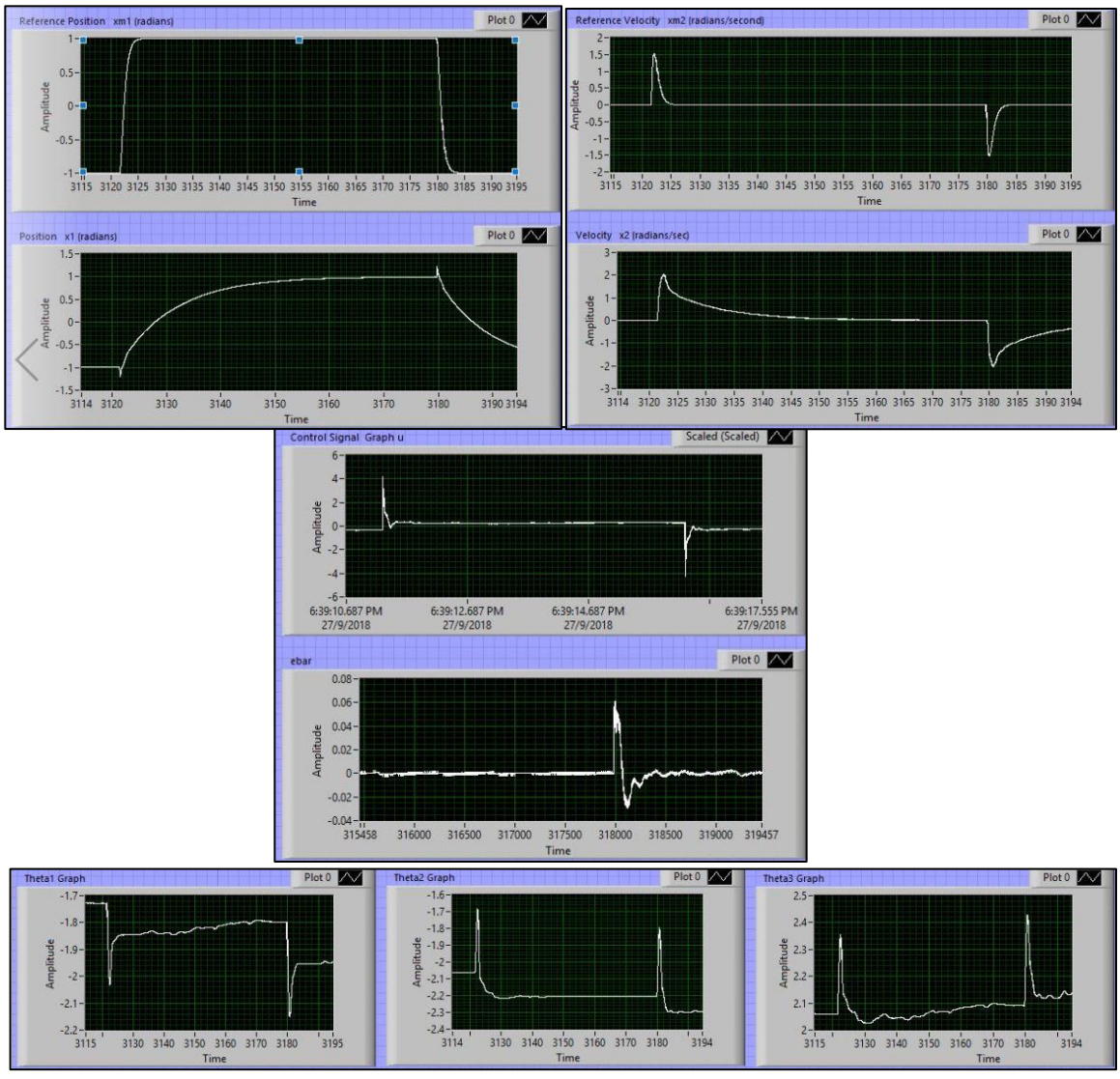


Figure 25 Test Result of setting up $\Gamma = diag(10,10,10); Q = diag(1,1)$

The control signal size is definitely larger because Γ is scaled up. However, not much improvement on the angular position is seen. It is also observed that angular velocity's transient has sharper peaks (overshoot-like) and its transient is stretched slightly longer than before. Reducing γ_{22} or q_{22} will eventually reduce the adaptation gain for angular velocity, which also implies that size of u will be lesser since it is the input to first order system that generates this angular velocity.

iii. Case of $\Gamma = diag(1,1,1); Q = diag(4,0.5)$

With setting above, we have the experimental results as follows:

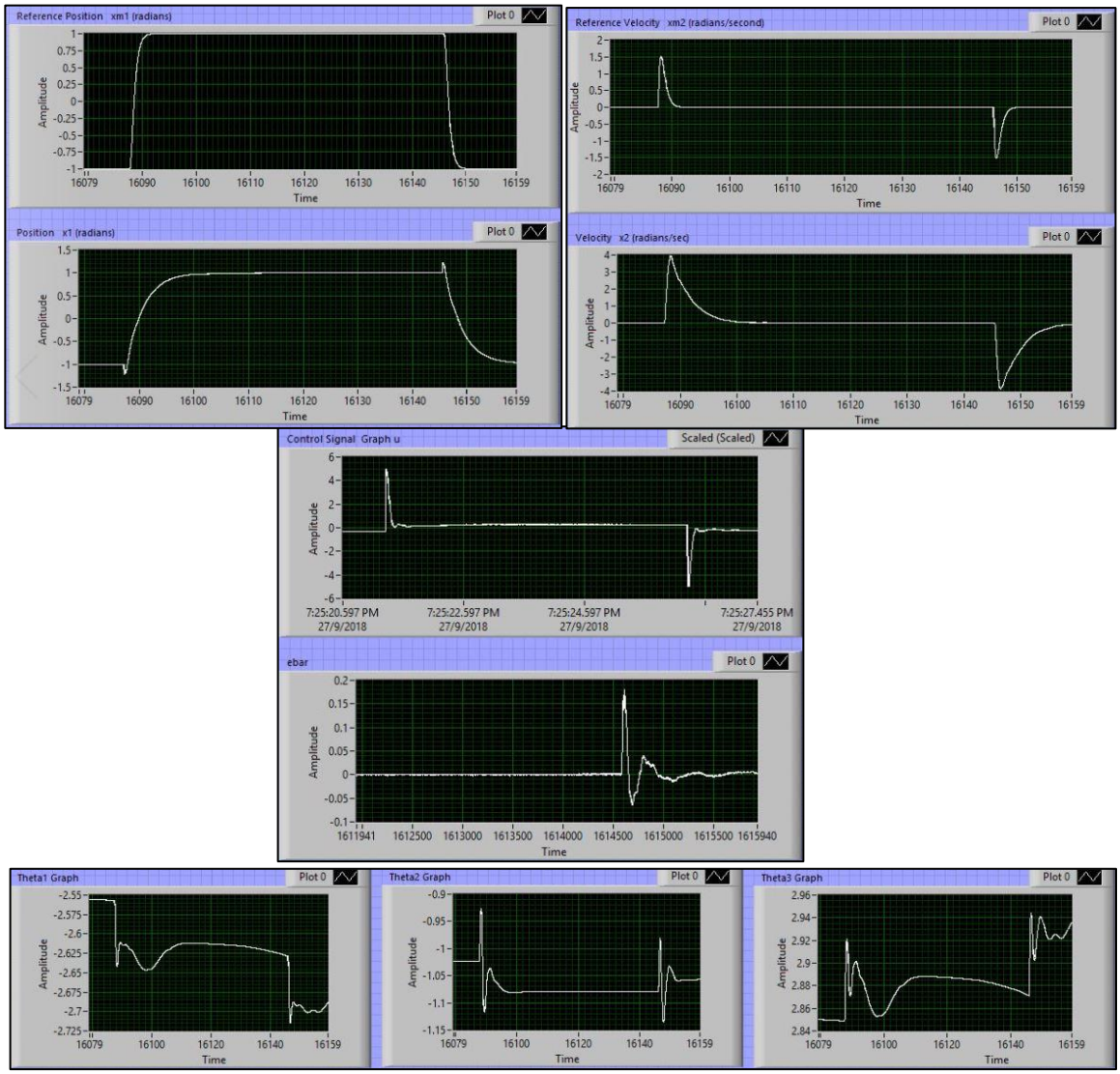


Figure 26 Test Result of setting up $\Gamma = diag(1,1,1); Q = diag(4,0.5)$

Results with the setting in this 3rd case looks better on transient-wise and having lesser state error. The decline in gain setting values related with the angular velocity reduce the width of transient time as shown in top right of Figure 26. As a result, the angular position's transient time is also reduced which eventually reduce state error.

iv. Best case found during experiment, $\Gamma = diag(3,0.5,2); Q = diag(4,0.5)$

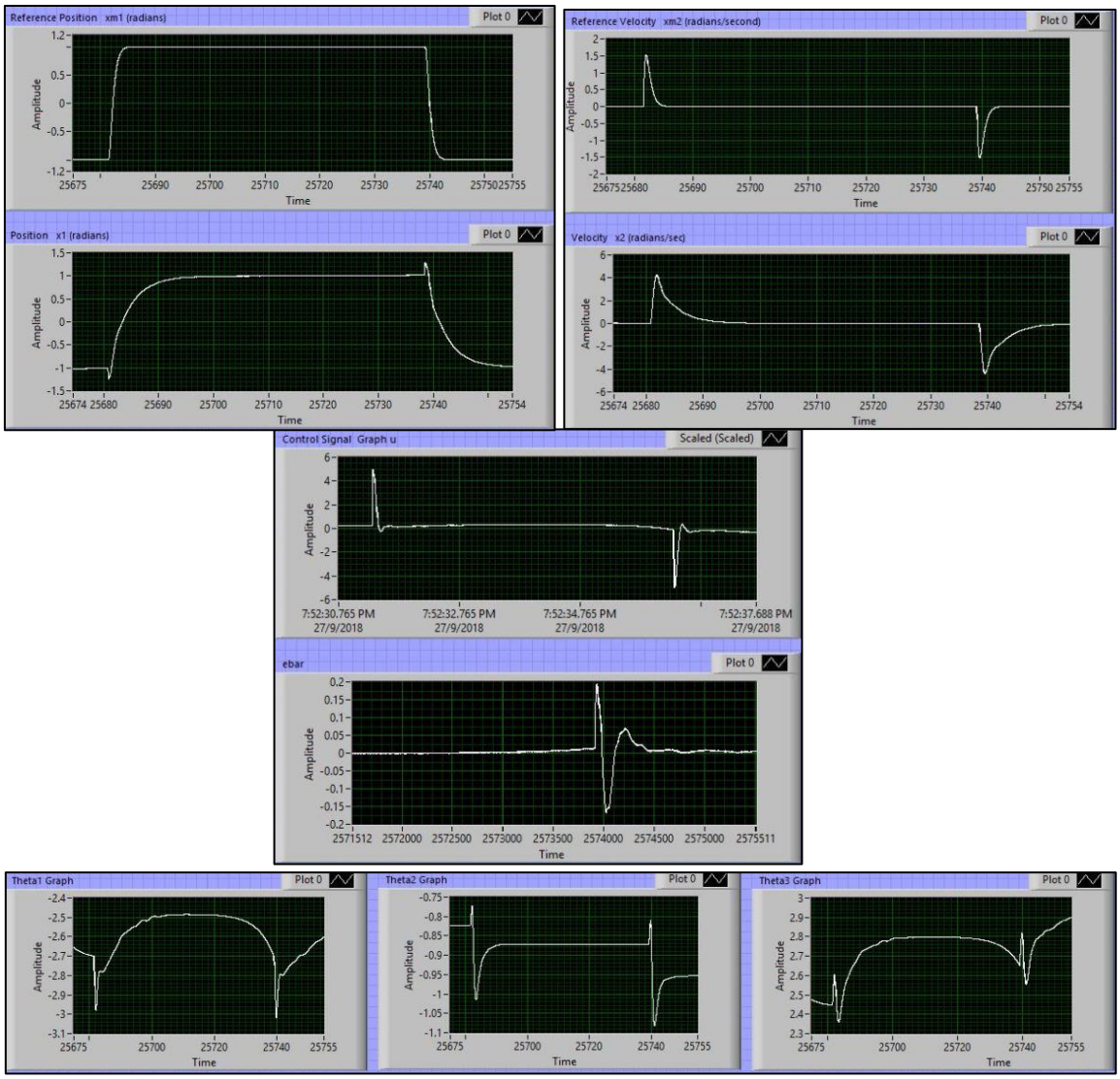


Figure 27 Test Result of setting up $\Gamma = diag(3,0.5,2); Q = diag(4,0.5)$

Controller parameter (θ) has smoother dynamics compared to all previous cases. There is a little overshoot observed on both ends of cycle, which most likely due to the control signal touching the saturation limit (5,-5) during starting up an impulse shown in the middle of Figure 27. The control size is not enough to counter-act these pulses when almost every states require high gradient (increment) to follow reference states. Nevertheless, the DC motor's angular position reach the steady state value the same as the reference steady values (-1, or 1) within reasonable settling time (around 1 seconds).

V. Conclusion

Adaptive controller for DC motor system to control angular position has been designed. The approximation of second order parameter of DC motor system in the form:

$$\frac{\theta}{u} = \frac{K_p}{s(\tau s + 1)}$$

were obtained by performing calibration and linear regression (least mean squares). Chosen 2nd order reference model in the state-space form was then compared with the state-space form of the DC motor system above. The state error and parametric error for the DC motor to follow the behaviour of reference model were then formularized into quadratic Lyapunov function $V(e, \Phi) = e^T Pe + |g|\Phi^T \Gamma^{-1} \Phi$. Ensuring the defined Lyapunov function to conform uniform stability requirement (with respect to e, Φ and time t) implied that $V(e, \Phi)$ eventually approached zero in decreasing manner, indicating e, Φ also approached zero when t is sufficiently large. To guarantee such stability in full-state measurable system, adaptive law $\dot{\Phi} = -sign(g)\Gamma \begin{bmatrix} x \\ r \end{bmatrix} e^T PB$ and control law $u = \theta_x^T x + \theta_r r$ came into the picture.

Simulating the overall functional diagram with the given adaptive and control law showed that indeed the state errors (including the angular positional error) reduced to zero after some time.

In the real-hardware check, it is found that the performance was not as ideal as simulation. The limitation of control signal as well as noises presence in motor system, modulation, and measurement affect the dynamic of controller parameter. As a result, the controller parameter would not converge to steady state value (exact controller parameter) in a short time. In fact, due to noise presence, the controller parameter slowly integrate the noise into its magnitude, making the convergence hard to achieve. Gain setting was therefore critical in order to reduce the amplification of these noises. Setting up adaptive gains and Q matrix slightly lower on velocity side (γ_{22} or q_{22}) helped to reduce the severity of noises and reduce the control size as well as the transient time of both angular velocity and position. The best setting of Γ and Q were found at $diag(3,0.5,2)$ and $diag(4,0.5)$ respectively.

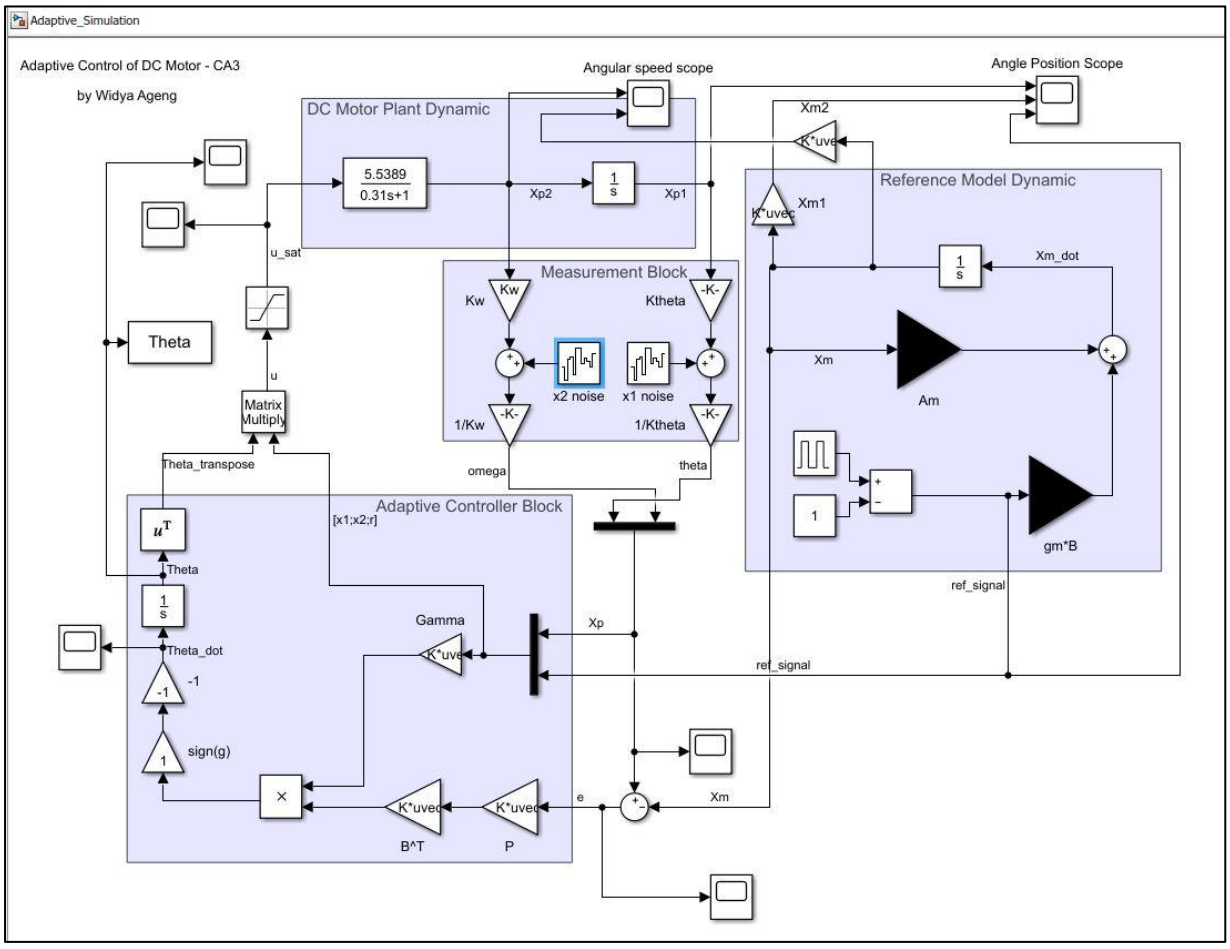
VI. Appendix: Code and Model

A. Adaptive Controller Code

```
%-------------------------------------------------------
% Adaptive Control,Full State Measurable
%-------------------------------------------------------
clear;clc;

%% Labview Sensor constants calibration, Ktheta, Kw

%Ktheta calibration using angle position, shaft dial
pos_degree = [258 288 318 348 378 408];
pos_radiant = pos_degree/(180/pi); % unit angle position to be used in the
control algorithm structure
x1 = [0 0.84 1.734 2.642 3.516 4.445]; % Labview readouts
Fittheta = polyfit(pos_radiant,x1,1);
Ktheta = Fittheta(1); % its reciprocal to be entered into labview block

%Fitting plot of K theta
figure;hold on;grid;
plot(pos_radiant,x1,'.r','MarkerSize',10);
minplot_theta = ((min(x1)-Fittheta(2))/Fittheta(1))-1;
maxplot_theta = ((max(x1)-Fittheta(2))/Fittheta(1))+1;
plot(minplot_theta:0.01:maxplot_theta,Fittheta(1)*(minplot_theta:0.01:maxpl
ot_theta)+Fittheta(2),'-b');
axis tight; legend('data','fitting line');
xlabel('Angular Position (rad)');
ylabel('Potentiometer output (Volts)');
title('K_{\theta} calibration');hold off;

%Kw calibration using digital tachometer on the module
w_rpm = [-162 -104 -46 0 47 104 162];
w_radps = w_rpm*2*pi/60; % unit angle velocity to be used in the control
algorithm structure
x2 = [-3.39 -2.17 -0.95 0 0.95 2.12 3.39];
Fitw = polyfit(w_radps,x2,1);
Kw = Fitw(1); % its reciprocal to be entered into labview block

%Fitting plot of Kw
figure;hold on;grid;
plot(w_radps,x2,'.r','MarkerSize',10);
minplot_w= ((min(x2)-Fitw(2))/Fitw(1))-1;
maxplot_w = ((max(x2)-Fitw(2))/Fitw(1))+1;
plot(minplot_w:0.01:maxplot_w,Fitw(1)*(minplot_w:0.01:maxplot_w)+Fitw(2),'-
b');
axis tight; legend('data','fitting line');
xlabel('Angular velocity (rads/sec)');
ylabel('Tachogenerator output (Volts)');
title('K_{\omega} calibration');hold off;


%% Reference model selection

%second order parameter
damping_rat = 1;
nat_freq = 2;

%state space representation of reference model
```

```
Am = [0 1; -(nat_freq^2) -2*damping_rat*nat_freq]; % to be entered into
labview block
B = [0;1];% to be entered into labview block
gm = nat_freq^2;% to be entered into labview block


%% Other Adaptive Controller Gain Matrices Setup & Plant Characterisation

% Adaptive Gain Gamma
Gamma = [3 0 0;0 0.5 0;0 0 2]; % to be entered into labview block
Gamma = [1 0 0;0 1 0;0 0 1];

% Q +ve definite matrix
Q = [4 0; 0 0.5];
Q = [1 0; 0 1];

% Calculating ARE equation Am'P + PAm = -Q
P = lyap(Am',Q); % to be entered into labview block

%Pretest to determine DC gain of motor system, Kp
u_labview = [-3 -2 -1 0 1 2 3]; % Experiment
Kpvec = polyfit(u_labview,w_radps,1); % Fitting
Kp = Kpvec(1);

%Fitting plot of Kp
figure;hold on;grid;
plot(u_labview,w_radps,'.r','MarkerSize',10);
minplot_u = ((min(w_radps)-Kpvec(2))/Kpvec(1))-1;
maxplot_u = ((max(w_radps)-Kpvec(2))/Kpvec(1))+1;
plot(minplot_u:0.01:maxplot_u,Kpvec(1)*(minplot_u:0.01:maxplot_u)+Kpvec(2),
'-b');
axis tight; legend('data','fitting line');
xlabel('Input Voltage (Volts)');
ylabel('Angular velocity (rads/sec)');
title('K_{p} calibration');hold off;

% Plant's parameter empirical determination through step response test
tau = 0.31; % Experiment
Mot_system = tf(Kp,[tau 1]);

% Plant's state space representation
Ap = [0 1;0 -1/tau];
g = Kp/tau;

% Approximation of perfect controller gain Theta star
% from Matching Condition
Theta_x_star = (B'*(Am-Ap)/(g*(B'*B)))';
Theta_r_star = gm/g;

%augmented theta
aug_theta_star = [Theta_x_star;Theta_r_star];
```

B. Adaptive Control Simulation



C. Lyapunov Function Checks

```
%-------------------------------------------------------
%-- LYAPUNOV CHECK & BOUNDEDNESS OF VARIOUS SIGNALS---
%-------------------------------------------------------

clear;clc;

%% Initiating state error and parameter error vectors
state_err = [0;0];
param_err = [0;0;0];

% x uniform stability theorem, page 15 Lect Note part 1
aug_err = [state_err;param_err];

P = eye(2);
Gamma = eye(3);
aug_Ly_mat = [P zeros(2,3);zeros(3,2) Gamma];


%% Bounding Lyapunov function with scalar functions
% increasing element in state vector x, means the L2 norm (||x||) increases
t = (0:0.01:2.5/sqrt(5))';
abs_t = sqrt(5)*t;
state_inc_mat = ones(size(aug_err,1),1)*t';

%Lower bound non-decreasing scalar function, a(x)=Lyap*(1-exp(-e'e));
low_a_vec = state_inc_mat(:,1)'*aug_Ly_mat*state_inc_mat(:,1)*(1-exp((-
1)*state_inc_mat(:,1)'*state_inc_mat(:,1)));

%Lyapunov function value, x'Px + phi'(Gamma^-1)phi
Lyap_val = state_inc_mat(:,1)'*aug_Ly_mat*state_inc_mat(:,1);

%Upper bound non-decreasing scalar function, b(x)=Lyap*(1+exp(-e'e));
upp_b_vec = state_inc_mat(:,1)'*aug_Ly_mat*state_inc_mat(:,1)*(1+exp((-
1)*state_inc_mat(:,1)'*state_inc_mat(:,1)));

%iterating along state axis,|x|
for i=2:length(t)
    low_a_vec =
[low_a_vec;(state_inc_mat(:,i)'*aug_Ly_mat*state_inc_mat(:,i))*(1-exp((-
1)*state_inc_mat(:,i)'*state_inc_mat(:,i)))];
    Lyap_val =
[Lyap_val;state_inc_mat(:,i)'*aug_Ly_mat*state_inc_mat(:,i)];
    upp_b_vec =
[upp_b_vec;state_inc_mat(:,i)'*aug_Ly_mat*state_inc_mat(:,i)*(1+exp((-
1)*state_inc_mat(:,i)'*state_inc_mat(:,i)))];
end

% Plotting of bounding functions and Lyapunov function
figure;
hold on;grid;
plot(abs_t,low_a_vec,'-r');
plot(abs_t,Lyap_val,'-.k');
plot(abs_t,upp_b_vec,'-b');
hold off;axis tight;
xlabel('|x|');
title('Bounds of Lyapunov function');
legend('a(x)','V(x)','b(x)');
```

```
%% Example of error spike during asymptotic tracking against zero which can
be avoided using adaptive law

% create exponential graph with spikes
sig_t = (0:0.005:10)';
sig_e = exp((-1)*sig_t);
noi_e = (0.1*sin(2*pi*1*sig_t)).*(exp(-0.9*sig_t));
noi_e = noi_e + 0.005*sin(2*pi*(10*rand()*sig_t)).*(exp(-0.9*sig_t));
spikes = 0.1*(sig_t==8)-0.3*(sig_t==9)+0.2*(abs(sig_t-7.7)<1e-10);
sig_e = sig_e + spikes + noi_e;
figure;hold on;grid;
plot(sig_t,sig_e,'-r');
ylabel('e');
xlabel('t');
title('example of state error convergence towards 0');
legend('state error, x_1');
hold off;
```

D. Plotting of Controller Parameter against Exact Parameters

```
%% Plotting theta trajectories and exact theta
%% Calculating exact theta

%Reference Model
%second order parameter
damping_rat = 1;
nat_freq = 2;

%state space representation of reference model
Am = [0 1; -(nat_freq^2) -2*damping_rat*nat_freq]; % to be entered into
labview block
B = [0;1];% to be entered into labview block
gm = nat_freq^2;% to be entered into labview block

% Calibration data
w_rpm = [-162 -104 -46 0 47 104 162];
w_radps = w_rpm*2*pi/60; % unit angle velocity to be used in the control
algorithm structure
u_labview = [-3 -2 -1 0 1 2 3]; % Experiment
Kpvec = polyfit(u_labview,w_radps,1); % Fitting
Kp = Kpvec(1);

% Plant's parameter empirical determination through step response test
tau = 0.31; % Experiment
Mot_system = tf(Kp,[tau 1]);

% Plant's state space representation
Ap = [0 1;0 -1/tau];
g = Kp/tau;

% Approximation of perfect controller gain Theta star
% from Matching Condition
Theta_x_star_plot = [(Theta.Time>=0) (Theta.Time>=0)]*diag((B'*(Am-
Ap)/(g*(B'*B)))');
Theta_r_star_plot = (Theta.Time>=0)*gm/g;

% finding axes bounds
min_y_plot1 = min(min(Theta.Data(:,1)),min(Theta_x_star_plot(:,1)));
max_y_plot1 = max(max(Theta.Data(:,1)),max(Theta_x_star_plot(:,1)));
min_y_plot2 = min(min(Theta.Data(:,2)),min(Theta_x_star_plot(:,2)));
max_y_plot2 = max(max(Theta.Data(:,2)),max(Theta_x_star_plot(:,2)));
min_y_plot3 = min(min(Theta.Data(:,3)),min(Theta_r_star_plot(:,1)));
max_y_plot3 = max(max(Theta.Data(:,3)),max(Theta_r_star_plot(:,1)));

figure;grid;
subplot(3,1,1);hold on;grid;title('\Theta vs \Theta^*');
plot(Theta.Time,Theta.Data(:,1),'-r');
plot(Theta.Time,Theta_x_star_plot(:,1),'-.b');
axis([min(Theta.Time) max(Theta.Time) min_y_plot1-0.5 max_y_plot1+0.5]);
legend('\theta_{x1}','\theta_{x1} exact');hold off;

subplot(3,1,2);hold on;grid;
plot(Theta.Time,Theta.Data(:,2),'-r');
plot(Theta.Time,Theta_x_star_plot(:,2),'-.b');
axis([min(Theta.Time) max(Theta.Time) min_y_plot2-0.5 max_y_plot2+0.5]);
legend('\theta_{x2}','\theta_{x2} exact');hold off;
```

```
subplot(3,1,3);hold on;grid;
plot(Theta.Time,Theta.Data(:,3),'-r');
plot(Theta.Time,Theta_r_star_plot(:,1),'-.b');
axis([min(Theta.Time) max(Theta.Time) min_y_plot3-0.5 max_y_plot3+0.5]);
legend('\theta_r','\theta_r exact');hold off;
```

E. Newton-Raphson Iteration to determine Second Order Time Domain Response Solution

```
%% Solving non linear equation using Newton raphson
syms x;
f = 0.3679 - exp(-0.31*x) + 0.31*x*exp(-0.31*x);
fdiv = diff(f);

x_prev = 0;
x_next = x_prev - (subs(f,x_prev)/subs(fdiv,x_prev));
while abs(x_prev-x_next)>1e-8
    x_prev = vpa(x_next,8)
    x_next = x_prev - (subs(f,x_prev)/subs(fdiv,x_prev));
end
```