ELEC6228 Design Project
Applied Control Systems

Karthik Sathyanarayanan
ks6n19@soton.ac.uk

I. Part A

A. Mathematical Modelling of the Robot Dynamics

The mathematical model for the Lego Mindstorms NXT robot is used as given in the Appendix. On this exercise, only the body pitch angle and the average of the wheel angle are being considered as the state variables.

B. Modelling of the Robot Dynamics in Simulink

To model the robot dynamics in Simulink, the state-space model given from the MATLAB are used to generate the matrices vector.The state space model has initial conditions as [0 0.1 0 0] representing each of the outputs as shown in appendix [2]. If the conditions were [0 -0l 0 0], the impulse response would go to negative infinity , while for [0 0.1 0 0], the response approaches positive infinity. This represent the robot falling in either directions.At [0 0 0 0], the robot remains perfectly balanced.

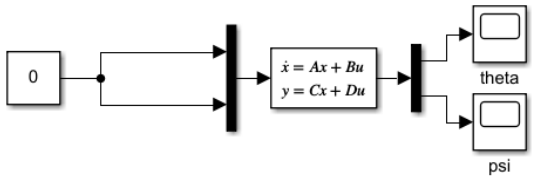

Fig. 1. State Space model [1]

This model is further simulated in Matlab to get an understanding of the system dynamics of the robot. We expect the robot to behave in the manner as showcased in figure 2. The first impulse response is that of θ and ψ while the second impulse response graph is of ϕ. These graphs match with the expected behaviour of the robot system dynamics.

C. A Simple PID Controller

Maybe the most famous control method is the Proportional–Integral–Derivative (PID) control. A PID controller implements a feedback control loop which continuously calculates an error e(t), which is the difference between the system's output y(t) and the reference signal r(t), and then applies a correction on that error based on Proportional, Integral and Derivative terms

There are a lot of different ways to calculate the three terms of the PID controller with the most famous one being the Ziegler-Nichols. The first step is to set I and D gains at zero and then P is increased until the output starts to oscillate.

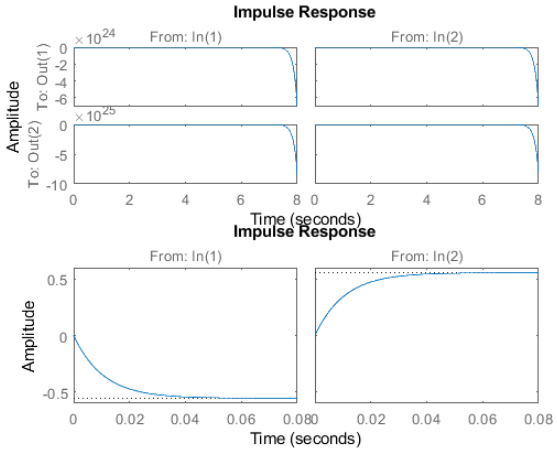

Fig. 2. Implulse response of model.

This K is called K_u and the period of the oscillations T_u. Then the three parameters are: $K_p = 0.6K_u$, $K_i = 1.2K_u/T_u$ and $K_d = 3K_uT_u/40$.The output of the robot are the higher order values of ψ and the control variables are higher order values of C. We linearise the variables used in the robot system dynamics and control C. Initially the angle ψ is controlled with a PID controller but it is observed that angleθ cannot be controlled. This means that the robot may be balanced but since θ is very unstable , the robot will be spinning in circles as the wheels keep spinning. This can be resolved by using a cascade PID approach. Since the desired outcome is a balanced robot, the objective is to stabilize ψ at zero. In order to stabilise ψ at zero and to drive ψ through the negative feedback loop and since this angle is not the system's output anymore, a Kalman filter is used as an observer to observe θ and ψ. The Kalman filter provides an estimation of all available states. Kalman filter uses the inputs, the plant's inputs and the ouput to estimate all the states. These states are then demuxed to get the four outputs and we observe just θ and ψ

The figure 3 shows the value of angle ψ over time. Due to the noise that we put in the system dynamics to mimic real world environment, we se that ψ has a small jitter but it is negligible and the range of the jitter is close to zero. Hence the controller works.

The figure 4 showcases the value of θ over time. We are not overtly controlling the angle θ but we do not want the angle to keep on increasing or decreasing sharply. This behaviour shows the gentle rolling of the robot in its effort to stabilise

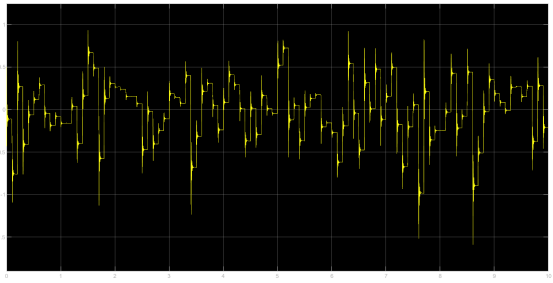

Fig. 3. Value of angle ψ (PID) over time.

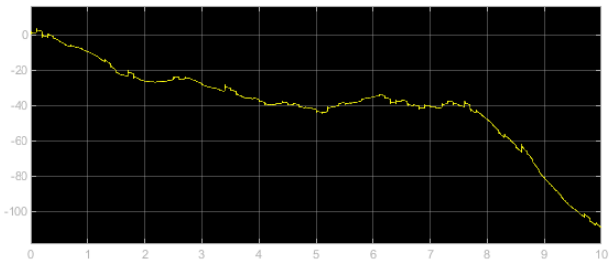

Fig. 4. Value of angle θ (PID) over time.

itself.

D. Model Predictive Control

In this task , the MPC controller replaces the PID controller to balance the robot. The same state space are used with the outputs ψ and θ.In order to stabilise ψ, we extract the second state x2 from the plant and a Kalman filter estimator is used with the same initial conditions of the plant [0 0.2 0 0]. In order to implement the MPC controller , a MPC block is used. This block receives the current measured output, the reference signal and measured disturbance signal. The controller then computes the optimally manipulated variables by solving a quadratic complexity programming problem. These optimally manipulated variables are then used as inputs to the plant.

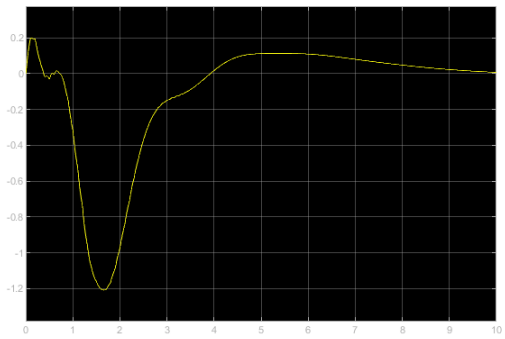

Fig. 5. Value of angle θ (MPC) over time.

The values of θ vary with the changing inputs until the robot stabilises and the angle reaches zero. Compared to the θ values of PID controller , we can control the angle in a better manner using MPC controller. The values of ψ follow

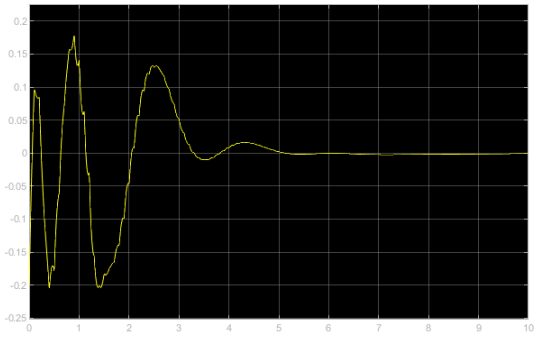

Fig. 6. Value of angle ψ (MPC) over time.

the same expected path and reach zero. However, when we compare this graph with that of PID, we can see that there is no jitter in the steady state value. The robot remains balanced in a better fashion when a MPC controller is used.

E. Discussion

The design seems to be working in an acceptable manner and successfully balances the robot. The design could be further improved if the entire controller was discretised. The problem that occurred while using discrete controllers could be due to the mismatch of sampling times within the various blocks. We can conclude that the MPC design works best for the task of balancing the robot. Though the PID design achieves the purpose is not as robust as the MPC design. The trade off of using an MPC design is that it increases the complexity of the control systems when a simple PID controller an balance the robot. Practical considerations such as the environment the robot would be , the noises it would encounter should be taken into consideration while deciding to use a particular design.In order to make things even more close to the real Robot, Zero order Holders(ZOH) should have been used in the inputs and outputs of the system according to the manufacturer's sample times(i.e. the rotary encoder for angle θ has maximum sample:1000 and the gyro sensor for ψ has maximum sample: 300). The reason this improvement were not used in this preject is that the use of these ZOH drove some derivatives to infinity even after the use of appropriate saturators.

II. Part B

A. Configuring the Quanser QUBE-Servo

For this exercise, both inertia disk and rotary pendulum were tested for all control system. For LQR control the rotary pendulum is accomplished with reasonable resistant from small push/force.

B. Mathematical Modelling of the Robot Dynamics

The mathematical model of the system are derived from the parameter given below:

Symbol	Description	Value
Inertia Disc		
m_d	Disc mass	0.053 kg
r_d	Disc radius	0.0248 m
Rotary Pendulum		
m_r	Rotary arm mass	0.095 kg
L_r	Rotary arm length	0.085 m
m_p	Pendulum link mass	0.024 kg
L_p	Pendulum link length	0.129 m

Fig. 7. QUBE Module parameter [1]

1) DC motor and Inertia Disk Mathematical Modelling

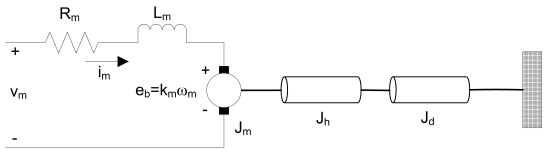

Fig. 8. Qube-Servo 2 DC motor schematic [1]

The Quanser QUBE-servo 2 is based on a direct-drive rotary DC motor system. the electric motor circuit is shown below and the parameters used are from the Figure 2 [1] and Figure 3 [1]. The back electromotive voltage $e_b(t)$ is determined by the speed of the shaft rotation, w_m and the electromotive constant k_m. There the back electromotive which go against current flow is given by:

$$e_b(t) = k_m \omega m(t) \tag{1}$$

By using Kirchhoff's second law, the following equation is generated:

$$v_m(t) - R)mi_m(t) - Lm\frac{di_m(t)}{dt} - k_m \omega m(t) = 0 \tag{2}$$

The motor inductance L_m will be ignored on this case because it is smaller than the resistance. Hence,

$$v_m(t) - R)mi_m(t) - k_m \omega m(t) = 0 \tag{3}$$

To find the motor current, the equation above can be solve in term of the current i_m:

$$i_m(t) = \frac{v_m(t) - k_m \omega m(t)}{R_m} \tag{4}$$

and because the motor shaft equation can be given as

$$J_{eq}\dot{\omega}_m(t) = \tau_m(t) \tag{5}$$

Where as τ_m is the applied torque based on the DC motor with J_{eq} is the total moment of inertia about the motor shaft, the torque equation can be found as:

$$\tau_m = k_m i m(t) \tag{6}$$

And The moment of inertia of the disk are given as:

$$J = \frac{1}{2}mr^2 \tag{7}$$

Finally with the DC motor transfer function for voltage-to-position given as shown below the state-space model can be found.

$$P(s) = \frac{{}_m(s)}{V_m(s)} = \frac{K}{s(\tau s + 1)} \tag{8}$$

2) Rotary Pendulum Mathematical Modelling

The equations of motion (EOM) for the pendulum system were developed using the Euler-Lagrange equation. This systematic method is second-order system equation often utilized to model complicated model dynamics. In the event that the total kinetic and potential energy of the system is obtained, then the Lagrangian can be found. A number of derivations are then calculated to yield the non linearized EOMs.:[]

$$\Big(m_p L_r^2 + \frac{1}{4}m_p L_p^2 - \frac{1}{4}m_p L_p^2 \cos(\alpha)^2 + J_r\Big)\,\ddot{\theta} + \Big(\frac{1}{2}m_p L_p L_r \cos(\alpha)\Big)\,\ddot{\alpha} + \Big(\frac{1}{2}m_p L_p^2 \sin(\alpha)\cos(\alpha)\Big)\,\dot{\theta}\dot{\alpha} - \Big(\frac{1}{2}m_p L_p L_r \sin(\alpha)\Big)\dot{\alpha}^2 = \tau - D_r\dot{\theta}. \tag{9}$$

and

$$\frac{1}{2}m_p L_p L_r \cos(\alpha)\,\ddot{\theta} + \Big(J_p + \frac{1}{4}m_p L_p^2\Big)\,\ddot{\alpha} - \frac{1}{4}m_p L_p^2 \cos(\alpha)\sin(\alpha)\dot{\theta}^2 + \frac{1}{2}m_p L_p g \sin(\alpha) = -D_p\dot{\alpha}. \tag{10}$$

The applied torque are coming from the base of the rotary arm. Hence gave the equation:[]

$$\tau = \frac{k_m\big(V_m - k_m\dot{\theta}\big)}{R_m} \tag{11}$$

To simplified the intricacy of nonlinear system, it is better to linearized the system resulting as:[]

$$(m_p L_r^2 + J_r)\,\ddot{\theta} + \frac{1}{2}m_p L_p L_r \ddot{\alpha} = \tau - D_r\dot{\theta}. \tag{12}$$

$$\frac{1}{2}m_p L_p L_r \ddot{\theta} + \Big(J_p + \frac{1}{4}m_p L_p^2\Big)\,\ddot{\alpha} + \frac{1}{2}m_p L_p g\alpha = -D_p\dot{\alpha}. \tag{13}$$

In terms of acceleration the linearized equations can described as:[]

$$\ddot{\theta} = \frac{1}{J_T}\Big(-\Big(J_p + \frac{1}{4}m_p L_p^2\Big)\,D_r\dot{\theta} + \frac{1}{2}m_p L_p L_r D_p\dot{\alpha} + \frac{1}{4}m_p^2 L_p^2 L_r g\alpha + \Big(J_p + \frac{1}{4}m_p L_p^2\Big)\tau\Big). \tag{14}$$

$$\ddot{\alpha} = \frac{1}{J_T}\Big(\frac{1}{2}m_p L_p L_r D_r\dot{\theta} - (J_r + m_p L_r^2)\,D_p\dot{\alpha} - \frac{1}{2}m_p L_p g(J_r + m_p L_r^2)\,\alpha - \frac{1}{2}m_p L_p L_r \tau\Big). \tag{15}$$

By calculating the moment of inertia from the pendulum link J_p and rotary arm J_r:

$$J_r = \frac{M_r L_r^2}{12} \tag{16}$$

$$J_p = \frac{M_p L_p^2}{12} \tag{17}$$

Resulting in the equation for the total inertia of

$$J_T = J_p m_p L_r^2 + J_r J_p + \frac{1}{4} J_r m_p L_p^2. \tag{18}$$

Finally, the linear state-space model of the systems can be determine as the state-space equations are given as:

$$\dot{x} = Ax + Bu$$

Where as $\dot{x}$ is the next state of the system and u is the control input

$$y = Cx + Du$$

Which can be defined accordingly for the rotary pendulum system as,

$$x = \begin{bmatrix} \theta & \alpha & \dot{\theta} & \dot{\alpha} \end{bmatrix}^T$$

$$y = \begin{bmatrix} \theta & \alpha \end{bmatrix}^T$$

and the state-space matrices representation of the linear system can be formatted as:

$$A = \frac{1}{J_T} \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & A1 & A2 & A3 \\ 0 & A4 & A5 & A6 \end{bmatrix}$$

$$A1 = m_p^2 \Big(\frac{L_p}{2}\Big)^2 L_r g$$

$$A2 = -D_r (J_p + m_p) \Big(\frac{L_p}{2}\Big)^2$$

$$A3 = -m_p \Big(\frac{L_p}{2}\Big) L_r D_p$$

$$A4 = m_p g \Big(\frac{L_p}{2}\Big) (J_r + mpL_r^2)$$

$$A5 = -m_p \Big(\frac{L_p}{2}\Big) L_r D_r$$

$$A6 = -D_p (J_r + m_p {L_r}^2)$$

$$B = \frac{1}{J_T} \begin{bmatrix} 0 & 0 & B1 & B2 \end{bmatrix}^T$$

$$B1 = J_p + m_p \Big(\frac{L_p}{2}\Big)^2$$

$$B2 = m_p \Big(\frac{L_p}{2}\Big) L_r$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$$

Symbol	Description	Value
DC Motor		
R_m	Terminal resistance	8.4 Ω
k_t	Torque constant	0.042 N.m/A
k_m	Motor back-emf constant	0.042 V/(rad/s)
J_m	Rotor inertia	4.0 x 10^{-6} kg.m^2
L_m	Rotor inductance	1.16 mH

Fig. 9. DC Motor parameter [1]

Where A is the state matrix, B is the input matrix, C is the output matrix and D is the feed-forward matrix and all matrices are constant. The matrices vector is chosen accordingly from the linear EOM equations shown above. Lastly, to convert the state-space matrices to be in terms of voltage the actuator dynamics are added for the state matrix and input matrix.

$$B = \frac{k_m B}{R_m}$$

$$A2 = A2 - \frac{k_m^2}{R_m B1}$$

$$A5 = A5 - \frac{k_m^2}{R_m B2}$$

C. Linear Quadratic Optimal Control

1) Background

The LQR optimal control algorithm is used to computes the control inputs u such that the performance index or the cost function is minimized optimally. The typical cost function used in LQR theory is shown below:

$$J = \sum_{t=0}^{\infty} (x_{ref} - x(t))^T Q (x_{ref} - x(t)) + u(t)^T R u(t)$$

The Q and R matrices control how expensive the cost of state variables either from the reference and the control measurements. If an element of matrix Q is increased, the performance index increases respectively with any difference from the desired setpoint/reference of the state variable, therefore the control gain K will become larger. On the other hand, if the values of element R are increased, the control will become expensive, however the control gains will be decrease. As the state reference are defined as,

$$x_{ref} = \begin{bmatrix} \theta & 0 & 0 & 0 \end{bmatrix}^T$$

2) Implementation

The LQR state-feedback controller for the rotary pendulum on the labVIEW as shown below:

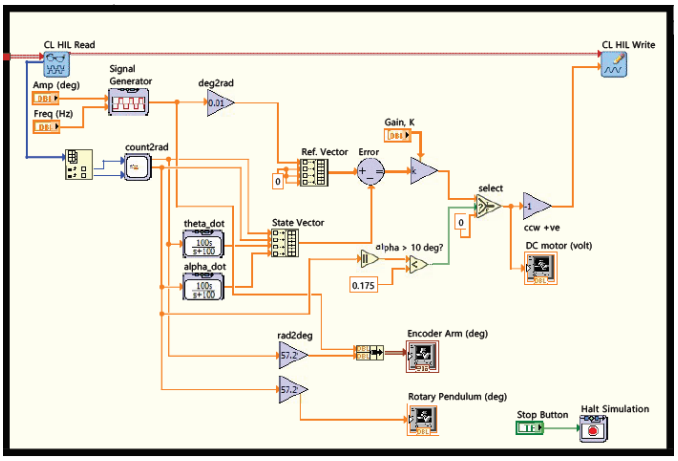

Fig. 10. LQR state-feedback controller block diagram

For this exercise, MATLAB function lqr is applied to obtain the optimal gain K which then can be applied to the labVIEW for the controller. The optimal gain K generated from the MATLAB are: $\begin{bmatrix} -1 & 34.2418 & -1.2254 & 3.0770 \end{bmatrix}$

To make it as simple as possible the Q matrix is just an identity matrix with the ρ weight is 1.

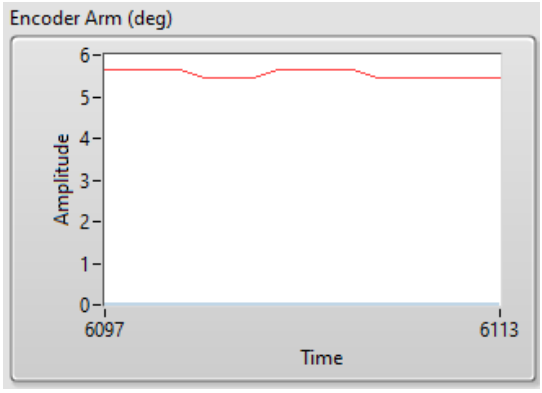

Fig. 11. Encoder Arm in degree

It can be observed that the encoder arm cannot steady on one place, this is due to the cable that power the encoder is quite firm and sometimes it moved the encoder.

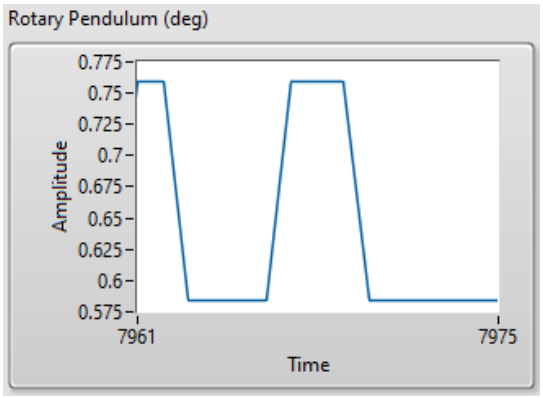

Fig. 12. Rotary pendulum in degree

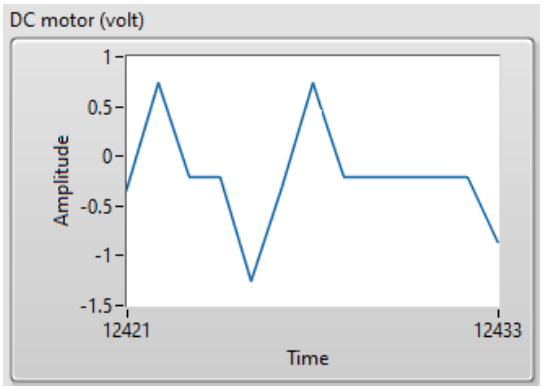

Fig. 13. DC motor voltage

For the rotary pendulum on Figure 6, it can be observed that the pendulum stay almost close to 0°degree. Additionally, the DC motor voltage can also handle the output voltage with acceptable value.

D. Model Predictive Control

1) Background

Model Predictive controller, as the name suggests, is used for predicting the consequetive behaviour of the model. It quite efficiently solves the optimization problem, and thus helps in steering the predicted output to the required reference. Due to it's constrained control of the system, which leads to significant improvements in the performance of the controller, MPC is quite widely used in industry. Compared to PID controllers, MPC has a better performance result since it overcomes the time delay feedback problem, due to its predictive nature. PID, on the other hand, relatively do not perform well with time delays. MPC consist of a prediction as well as control horizon for depicting future output of the controller. It is a discrete system model which achieves potential control signal values by reducing the cost function of the given state. There are many Industries, the most prominent being aero and automative industry, that quite evidently knows the importance of MPC and uses it in its many applications, such as autonomous ground vehicles, air handling control of diesel engines etc.

2) Implementation

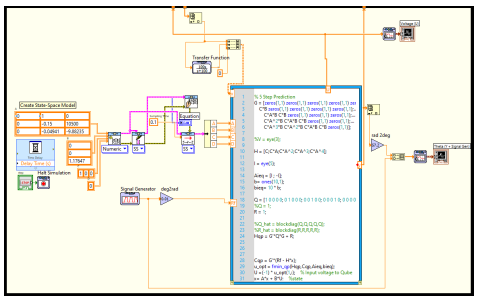

Fig. 14. Model Predictive controller block diagram

This MPC controller uses DC servo motor model to choeve MPC. In mathscript block, for constrained MPC, iterative prediction horizon of five steps has been used. The voltage limit has been set to +/- 10 V whereas the design parameters Q R has been provided the value in the form of matrix (5*5) and constant value 1, respectively, by iterative methodology. In LabVIEW mathscript, $fmin_{qp}$ is used to find the optimum input for the Qube. It is basically used to solve the quadratic programming problem which cannot be solved analytically and performs the similar functionality as quadprog in MATLAB. The weight matrix, Q, is used to tune the MPC controller by iterative methodology. The obtained waveform is a square signal of amplitude 60 degree for 4 sec as required by the task. Due to Kalman filter, the output signal was able to track the reference signal quite accurately.

The MPC design is created with labVIEW tool, for which the mathematical model has been previously described. Next, according to the depicted model, a continuous state space system is created with A being a matrix of size (3*3), B matrix of size (3*1), C matrix of size(1*3) and D having constant 0 value. Furthermore, in order to transform a continuous time model to a discrete one, a continuous to discrete converter along with sampling time 0.1 is applied. This process is essential for digital implementation on the digital devices. In order to display the state space equation, 'state name to models' as well as 'draw state space equation' blocks has been used accordingly.

```
% 5 Step Prediction
G = [zeros(1,1) zeros(1,1) zeros(1,1) zeros(1,1) zeros(1,1);...
     C*B zeros(1,1) zeros(1,1) zeros(1,1) zeros(1,1);...
     C*A*B C*B zeros(1,1) zeros(1,1) zeros(1,1);...
     C*A^2*B C*A*B C*B zeros(1,1) zeros(1,1);...
     C*A^3*B C*A^2*B C*A*B C*B zeros(1,1)];

H = [C;C*A;C*A^2;C*A^3;C*A^4];

I = eye(5);

Aieq = [I ; -I];
b= ones(10,1);
bieq= 10 * b;

Q = [1 0 0 0 0; 0 1 0 0 0; 0 0 1 0 0; 0 0 0 1 0; 0 0 0 0 1]
R = 1;

Hqp = G'*Q*G + R;
Cqp = G'*(Rf - H*x);
u_opt = fmin_qp(Hqp,Cqp,Aieq,bieq);
U =(-1) * u_opt(1,:);   % Input voltage to Qube
x= A*x + B*U;   %state
y = C*x;        %Theta (angle of rotation)
K = U/x;
```

Fig. 15. Model Predictive controller block diagram using MathScript

subsequently, all the matrix vectors are then unbundled and provided to mathscript block, which consist of the information/programme to form constrained MPC controller. For the requirement to rotate the Quanser inertia disk upto 60 degree for 4 period of seconds, the demand reference for tracking MPC has been provided by the signal generator, in the form of R_f, to mathscript block. Also, to validate the proper tracking of the system, discrete Kalman filter is used. The input of the controller, U, which is the voltage for the Qube, is provided from the mathscript block to waveform function to demonstrate the wave, whereas output Y is provided to waveform function θ angle.

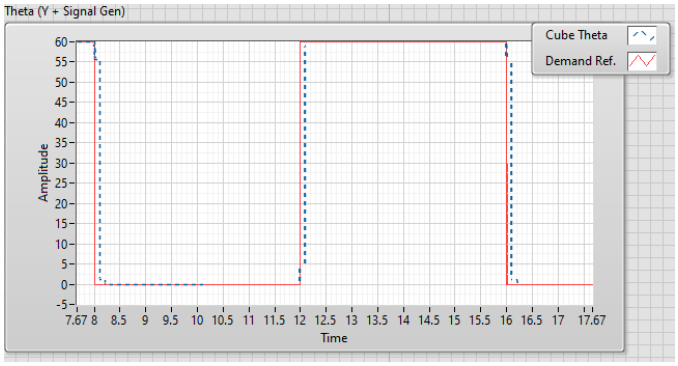

Fig. 16. Output θ results compare with the reference signal

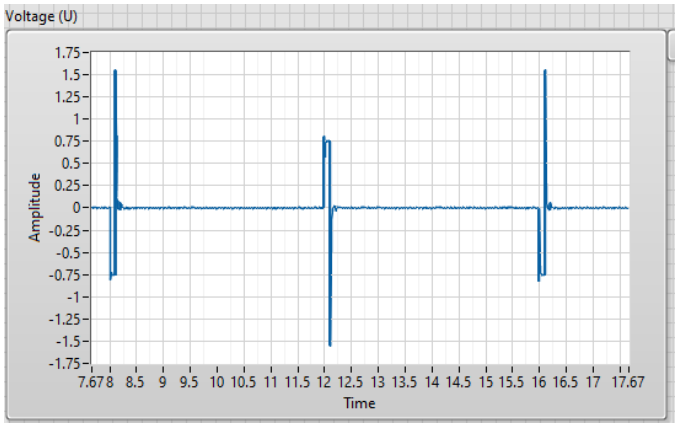

Fig. 17. Model Predictive controller block diagram using MathScript

III. Iterative Learning Control

A. Background

The system is defined as

$$x_k(t+1) = Ax_k(t) + Bu_k(t)$$

$$y_k(t) = Cx_k(t)$$

Here A = $\begin{bmatrix} 0 & 1 & 0 \\ 0 & -0.15 & 10500 \\ 0 & -0104941 & -9.88235 \end{bmatrix}$, B = $\begin{bmatrix} 0 \\ 0 \\ -1.17647 \end{bmatrix}$, C = $\begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$. y_k is denoted as

$$y_k = Gu_k$$

The matrix is written using the lifted form of G.

$$\begin{bmatrix} y_k(1) \\ y_k(2) \\ . \\ y_k(N) \end{bmatrix} = \begin{bmatrix} CB & & & \\ CAB & CB & & \\ . & . & & \\ CA^{N-1}B & CA^{N-2}B & . \; . & CB \end{bmatrix} \begin{bmatrix} u_k(0) \\ u_k(1) \\ . \\ u_k(N-1) \end{bmatrix}$$

u_{k+1} is governed by the following equation :

$$u_{k+1} = u_k + \gamma_k G^T e_k$$

However, since we are using norm optimal ILC algorithm and a fixed value of gamma as the graph for error converges faster and takes less computing prowess in every trial .

$$u_{k+1} = u_k + inv((R + R * Q * G) * G + Q * e)'$$

e_{k+1} is reference signal subtracted by Gu_{k+1}. Substituting and simplifying we would arrive at the following equation.

$$e_{k+1} = (I - \gamma_k G)e_k$$

$$G = \begin{bmatrix} CB & & & \\ CAB & CB & & \\ . & . & & \\ CA^{N-1}B & CA^{N-2}B & . \; . & CB \end{bmatrix}$$

if we used a fixed γ_k, then the fastest convergence would require minimising $||I - gammaG' * G||^2$ with respect to γ_k. This has the solution below

$$gamopt = 1/norm(G)^2$$

The aspiration for an iterative learning controller is that it should converge the error to zero and this convergence should happen in a monotonic fashion for the controller to work within a reasonable limit.

B. Implementation

The approach to the implementation of iterative learning controller has been to use mathscript blocks to implement most of the controller. The above figure shows the snapshot of the mathscript blocks used in the design of the controller.The mathscript blocks are placed inside of the control blocks. The first mathscript block gives us a G matrix as discussed earlier and a reference signal for the inertia disk given as:

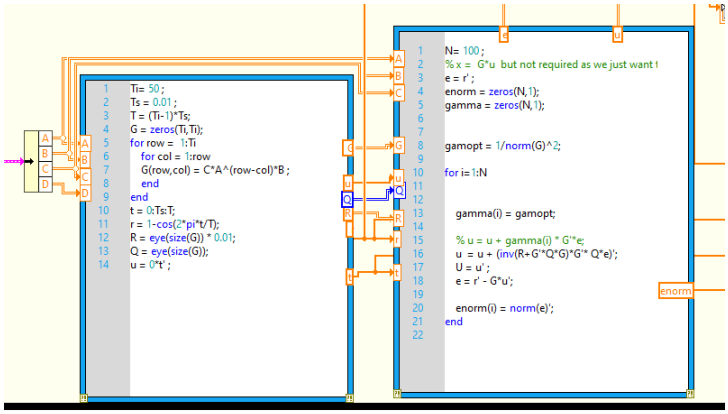

Fig. 18. Snapshot of VI.

$$r = 1 - cos(2 * pi * t/T);$$

where t is the time and T is the total time. The second mathscript block makes the controller by making 100 trials of iteration. Initially the error is equal to the reference signal. We calculate the gamma optimal and using a for loop calculate theinput gain for each trial k. The error is calculated with each and every trial and stored in array e. The following figures will illustrate the behaviour of various parameters of the controller.

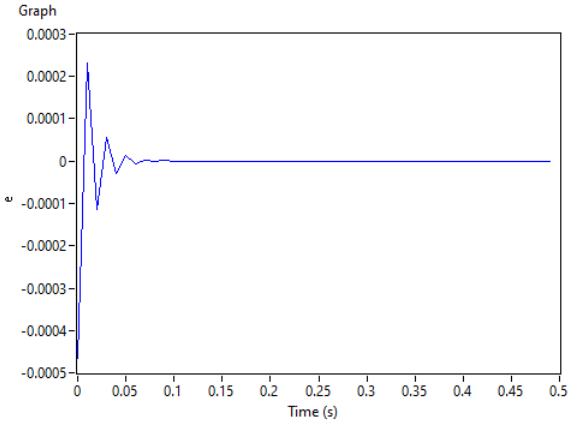

Fig. 19. Error as a function of time.

The figure 13 shows the error being calculated for every sample time till time T. We can see that the error approaches zero is less than 0.1 seconds.

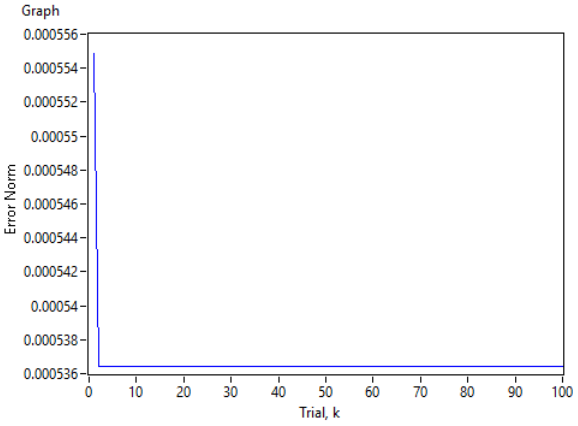

Fig. 20. Tracking error norm as a function of time.

The figure 14 shows the monotonic convergence of error norm. Since both e_k and norm of e_k converge, the controller is working up to our aspired standards.

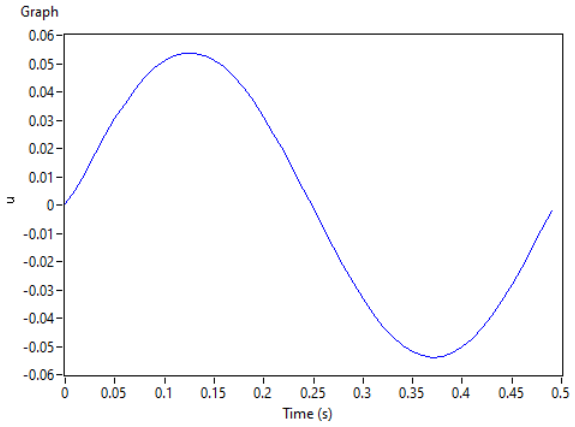

Fig. 21. Input u as a function of time.

The input gain in figure 15 shows the values reaching + 0.051 to -0.051 in a sinusoidal curve as expected based on the reference signal given. However the time period is decreased by the controller and the input gain changes more vigorously than the reference signal

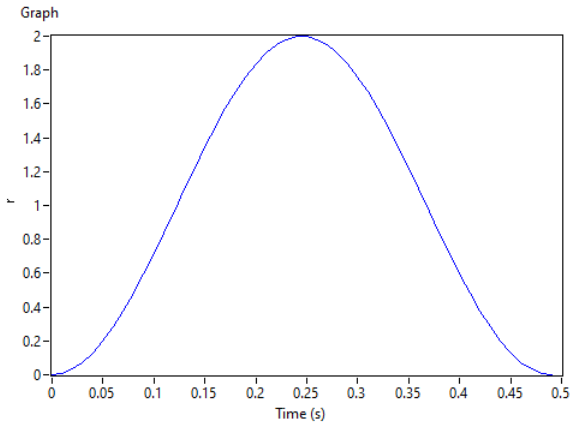

Fig. 22. Reference signal as a function of time.

The reference curve is figure 16 illustrates the way in which the inertial disk moves .

IV. Comparison and Discussion

The LQR controller was the only one that could successfully achieve the rotary pendulum balance. It did that with a good degree of robustness. The MPC controller did successfully manage to implement the design onto the QUBE servo motor. The ILC controller worked well and was very aggressive but was not tested for either in HIL block simulations or on the QUBE servo motor.

The LQR controller is a versatile controller and has proven itself to work in myriad variety of tasks. The MPC controller is a robust controller .Compared to PID controllers, MPC has a better performance result since it overcomes the time delay feedback problem, due to it's predictive nature. The ILC controller has the potential to be the most robust and highly precise controller as the computation time for error minimisation is less than 0.1 seconds and should be able to control the disk very reliably and accurately.

V. Conclusion and Future Work

In the LQR controller , the controller is working perfectly on the QUBE and the pendulum remains balanced as expected.The rotary pendulum almost stays close to zero degrees and the DC motor handles the output voltage at an acceptable value. However the Q matrix was 1st value is identity matrix and we should be able to change the value of 1st row and 1st column. The system crashes when this is attempted. The model also used Matlab functions to generate the gain and a labview VI model could have been used.

The MPC controller works as desired and the inertial disk does follow the specification as desired except the delay of 4 seconds. in future an accurate tracking of MPC is desired and a bit of delay is desired as the delay of 4 seconds is not replicated in the servo motor.

The ILC controller seems to be working in an acceptable manner. However this could not be verified due to the absence of HIL simulation modules. The controller also does not follow the specification of a 60 degree swing with a 4 second hold. It rather behaves as an oscillating disk. However in the real world, a delay would be there in the oscillations and it is very difficult to accurately measure the delay the disk would have during oscillation.

References

[1] Quanser QUBE-Servo 2 Experiment, Setup and Configuration, 2016. Accessed on: April. 12, 2020. [Online] Available: https://www.made-for-science.com/en/quanser/?df=made-for-science-quanser-qube-servo-2-usermanual.pdf

[2] $https://secure.ecs.soton.ac.uk/notes/elec6228/ELEC6228_Project.pdf$