APPLICATION OF LQR CONTROL FOR TWO-WHEEL SELF-BALANCING ROBOT

Lwin Mg Mg Thwin[1], Ye Chan[2]

Abstract

The concept of two-wheel self-balancing robot is based on an Inverted pendulum (IP) on a cart. The control of an IP has been the most popular benchmark, among others, for teaching and researches in control theory and robotics. The implemented control technique is full state feedback control, Linear Quadratic Regulator (LQR). The LQR algorithm is essentially an automated way of finding an appropriate state-feedback controller. The full state system information about the position and velocity of the cart are obtained from encoder on dc motor and angular position and angular velocity of the pendulum are obtained from IMU sensor MPU6050. A complimentary filter was used to weigh the accelerometer and gyro signals together to determine the pitch angle. This two-wheel self-balancing robot was implemented by using the Arduino Mega microcontroller. The microcontroller can be programmed with the Mathworks® Simulink® program using the Rensselaer Arduino Support Package.

Keywords: Linear Quadratic Regulator (LQR), complementary filter, microcontroller

Introduction

Control theory and its applications continue to grow and expand into new areas of our lives. From driverless cars to electric grid management to financial modeling we see new applications almost daily. These advances require that education grows and adapts as well. The content of control theory curriculum also has to include these new topics and applications to keep up with technology [Brian Howard, (2015)].

Two wheel self-balancing robot is a complicated non-linear system. It has also become great consideration as a research entity because of the unstable character of the system. The two wheel self-balancing robot is based on the fundamental principle of Inverted pendulum. Inverted pendulum has many practical applications such as human walking robots, missile launchers, earthquake resistant building design etc. Development of control system for a two-wheel self-balancing robot has been a huge area of research for the past few years. This is mainly due to its nonlinear dynamics. It became an important test platform for the design and development of missiles, automobiles, space crafts, robots. The simplest method of control system is by using a PID controller [Keerthi Prakash, (2016)].

Simple PID controller cannot give the efficient solution to inverted pendulum single-mass system or inverted pendulum double-mass system, because these systems include nonlinearities, coupling, and uncertain dynamics [Yizhu Li, (2018)]. In this work, Linear Quadratic Regulator (LQR) was applied for two-wheel self-balancing robot. LQR is a well know method to determine the feedback gains of a dynamic system. It's assumed that we have an optimal full-state feedback, i.e. that we can measure all of our states.

There will be only one axle connecting the two wheels and a platform will be mounted on that to make two-wheel self-balancing robot which can balance itself. To obtain this physical structure, two wheels are mounted with two dc motors in this work. The platform will not remain

[1] PhD Candidate, Assistant Lecturer, Department of Physics, Taungoo University

[2.] Dr, Professor and Head, Universities' Research Centre, University of Yangon

stable itself because the system is not stable. The control objective is to keep the two wheels robot in the upright position by using inertial measurement unit (IMU) sensors and optical encoder. At first we have decided to just balance the robot on its two wheels.

Mathematical Modeling

Mathematical model is required to design the control law. Therefore, Newton law based model of the IP has been derived. The IP consists of a moveable cart rail system and a swing-able pole connected to the cart. A free body diagram of an inverted pendulum mounted on a motor-driven cart is shown figure (1). Cart position is controlled with DC motor. The non-linear mathematical model of the IP is derived using the Newton law approach. Vertical force does not affect the cart position and the horizontal movement is controlled by the forces applied through DC motors [Katsuhiko Ogata, (2010)]. The obtained non-linear mathematical model of system is given by equations (1) and (2). The system state and parameters are explained in table (1) and system specifications are provided in table (2).

$$(M_c + m)\ddot{x} + ml\ddot{\theta}\cos\theta = ml\dot{\theta}^2\sin\theta + u \tag{1}$$

$$(I + ml^2)\ddot{\theta} + ml\cos\theta\,\ddot{x} = mgl\sin\theta \tag{2}$$

Since the inverted pendulum must be kept vertical position, it can be assumed that θ and $\dot{\theta}$ are small quantities such that sin $\theta \cong \theta$, cos $\theta = 1$ and $\dot{\theta}^2\theta = 0$. Then the non-linear system equations (1) and (2) can be linearized to equation (3) and (4) [Liu, Jinkan, (2017)].

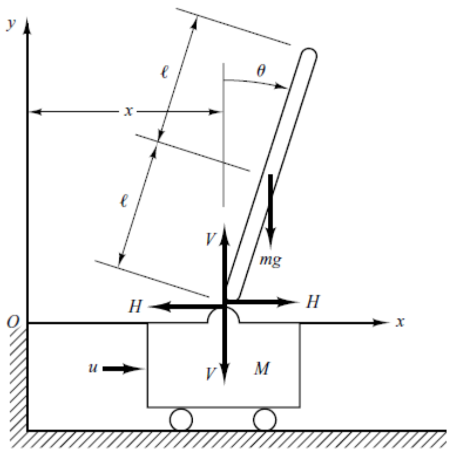


Figure 1 Free body diagram of inverted pendulum system

Table 1. The system states and parameters

Symbol	Description
M_c	mass of cart
m	mass of pendulum
l	length of pendulum
g	gravity
$\theta, \dot{\theta}, \ddot{\theta}$	angular position, angular velocity and angular acceleration of the pendulum rod from the vertical line
$x, \dot{x}, \ddot{x}$	cart position, velocity and acceleration respectively
u	control force applied to the cart
I	moment of inertia of the rod

Table 2 Parameter measurement values for the system

Symbol	Meaning	Value	Unit
M_c	Cart mass	0.46	kg
m	Pendulum mass	0.14	kg
L	Cart length	0.23	m
l	Length of pendulum	0.17	m
I	Inertia	$\frac{1}{12}mL^2$	kg/m^2
g	Gravity	9.8	m/s^2

$$\ddot{\theta} = \frac{m(m+M_c)gl}{(M_c+m)I+M_cml^2}\,\theta - \frac{ml}{(M_c+m)I+M_cml^2}\,u \tag{3}$$

$$\ddot{x} = -\frac{m^2gl^2}{(M_c+m)I+M_cml^2}\,\theta + \frac{I+ml^2}{(M_c+m)I+M_cml^2}\,u \tag{4}$$

The system equations can be reduced as follow.

$$\ddot{\theta} = t_1\theta + t_3u \tag{5}$$

$$\ddot{x} = t_2\theta + t_4u \tag{6}$$

where $t_1 = \frac{m(m+M_c)gl}{(M_c+m)I+M_cml^2}$, $t_2 = -\frac{m^2gl^2}{(M_c+m)I+M_cml^2}$,

$t_3 = -\frac{ml}{(M_c+m)I+M_cml^2}$ and $t_4 = \frac{I+ml^2}{(M_c+m)I+M_cml^2}$.

Let define as $x_1 = x$, $x_2 = \dot{x}$, $x_3 = \theta$ and $x_4 = \dot{\theta}$. Therefore,

$$\left.\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= t_2x_3 + t_4u \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= t_1x_3 + t_3u \end{aligned}\right\} \tag{7}$$

Converting equation (7) to the equivalent state space form given as,

$$\dot{X} = A\mathbf{x} + Bu \tag{8}$$

where $\dot{X} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix}$, $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$, $A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & t_2 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & t_1 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 \\ t_4 \\ 0 \\ t_3 \end{bmatrix}$

x is the *n* dimensional state vector, **u** is the *m* dimensional input vector, **A** is the $n \times n$ system matrix and **B** is the $n \times m$ control matrix.

By using parameter measurement values from table (2), the system equation becomes

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -2.4877 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 62.7141 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ 2.0897 \\ 0 \\ -10.6657 \end{bmatrix} u(t) \tag{9}$$

Stability

One of the first things we want to do is analyze the stability of the open-loop system (without any control). The poles or eigenvalues of the system matrix, *A*, determine the stability of the system. For this system based on the state space equation (9), the poles can be found simply by using Matlab command (poles = eig(A)). The poles or eigenvalues of opened loop system matrix are 0, 0, 7.9192 and -7.9192. The system has four poles with one in the right half plane which makes the system unstable. Therefore, a linear controller needs to be designed to force the poles in the left half plane.

Control Algorithm

Most single input, single output (SISO) systems can be adequately analyzed and controlled using basic transfer function techniques and PID type controllers. For systems that have multiple inputs and multiple outputs (MIMO) a more suitable type of control framework is called full state feedback. A full state feedback controller measures or estimates all of the system "states" and uses this "state" multiplied by a gain matrix to control the system.

The inverted pendulum based two-wheel self-balancing robot is an unstable and nonlinear system. In order to make the pendulum in upright position where the system will be stable, the suitable control algorithm has to be implemented.

Controllability and observability

The concept of controllability and observability were introduced by Kalman (1960) and play an important role in the control of multivariable systems. A system is said to be controllable if a control vector u(t) exists that will transfer the system from any initial state $x(t_0)$ to some final state x(t) in a finite time interval. A system is said to be observable if at time t_0, the system state $x(t_0)$ can be exactly determined from observation of the output y(t) over a finite time interval [Roland S. Burns, (2001)].

If a system is described by the following equations

$$\dot{X} = A\mathbf{x} + Bu$$

$$y = C\mathbf{x} + Du \qquad (10)$$

then a sufficient condition for complete state controllability is that the $\boldsymbol{n}\times\boldsymbol{n}$ matrix

$$M = [B: AB: \ldots : A^{n-1}B] \qquad (11)$$

contains $\boldsymbol{n}$ linearly independent row or column vectors, i.e. is of rank $\boldsymbol{n}$ (that is, the matrix is non-singular, i.e. the determinant is non-zero). Equation (11) is called the controllability matrix.

The system described by equations (10) is completely observable if the $\boldsymbol{n}\times\boldsymbol{n}$ matrix

$$N = [C^T: A^TC^T: \ldots : (A^T)^{n-1}C^T] \qquad (12)$$

is of rank n, i.e. is non-singular having a non-zero determinant. Equation (12) is called the observality matrix. The rank of controllability and observability matrix of a system model can be determined in Matlab using command rank(ctrb(A,B)) or rank(ctrb(sys)) and rank(obsv(A,C)) or rank(obsv(sys)) respectively.

Linear Quadratic Regular (LQR) Control

LQR is a linearized and optimal control technique which provides optimal gains for the systems. It is more suitable for the linear systems having no uncertainties or disturbances. The major benefit of this technique is that it gives the gains to minimize the cost function [Saqib Irfan, (2108)]. For an n^{th} order system the general cost function of LQR is given as,

$$J = \int_0^{\infty} [x^T Q x + u^T R u]\, dt \qquad (13)$$

where, Q $\in$ $R^{n\times n}$ is positive definite or positive semi definite Hermitian matrix or real symmetric matrix, R $\in$ $R^{r\times r}$ is a positive definite Hermitian matrix or real constant number and J is always scalar quantity. The Q matrix is a weighting function for the states and R is weighting function for the inputs. LQR gain is computed as,

$$K = R^{-1}B^{T}P \tag{14}$$

The Riccati equation is given in equation (15) to solve the optimal controller.

$$PA + A^{T}P + Q - PBR^{-1}B^{T}P = 0 \tag{15}$$

The control law for the linear system is given as,

$$u = -K\mathbf{x} \tag{16}$$

$$\dot{X} = A\mathbf{x} - BK\mathbf{x}$$

$$\dot{X} = (A - BK)\mathbf{x} \tag{17}$$

Using Matlab to solve the equations, this controller had the following pole locations and gains. The controller parameters R = 0.3 and the diagonal values of Q matrix = [1, 1, 10, 1] are chosen to achieve good stability performance of the system. The poles and LQR gain matrix for the closed loop system are [-22.3818 -3.6206 -1.7162 -1.7162] and [-2.5820 -3.4896 -22.6641 -3.4435] respectively. Since all poles are in the left half plane, the system will be stable at pendulum upright position. Assuming it is possible to directly measure the entire state (i.e. y = **x**) implementing a state space controller is really simple. The state is simply multiplied by a control gain matrix KLQR and the result is fed back into the plant. As a block diagram for full state feedback control system is shown in figure (2).

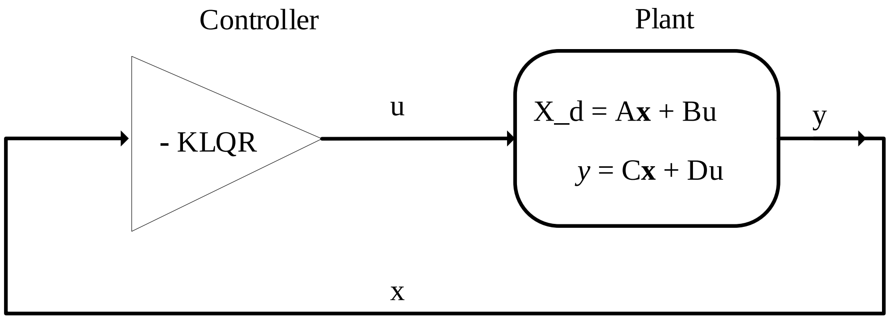


Figure 2 The block diagram of full state feedback control system

Results and Discussion

Implementation results

Simulink support package for Arduino hardware and Rensselaer Arduino support package had been installed to implement this research as prerequisite. The dynamics of the plant was sampled at fixed f_s = 200 Hz, which equals to the sampling frequency of the IMU sensors. Therefore, the microcontroller will execute the Simulink code every 50 milliseconds. The real time simulation involved in the proposed work was performed on Simulink, with simulation time of 30 s. But the following figures for the system response graphs are zoomed out to view clearly comparative study.

The controller from figure (2) was implemented as a Simulink model, shown in figures (3) and (4). The based part of two-wheel robot position, system state x can be obtained from the encoders on the motor and its velocity, system state *x_dot* is obtained from derivative block of *x* data. The Simulink diagram associated with robot position and velocity is shown in figure 3(a). Micro-Electro-Mechanical Systems (MEMS) package (MPU6050) is used to provide the rate of pitch angle $\dot{\theta}$, and pitch angle θ, as shown in figure 3(b). Measuring this pitch angle using the MPU6050 provided to be a challenging part of designing the controller. The accelerometers were relatively slow to respond and could not be used by themselves to measure pitch angle. The gyro signals have noise and a DC offset (bias) and could not be used alone to measure the angle.

A complimentary filter was used to weigh the accelerometer and gyro signals together to determine the pitch angle. Figure 4(b) shows the complimentary filter design used in the two-wheel robot. All four of the state variables are passed into a vector and multiplied by the LQR gain vector KLQR. The output of the gain is then passed onto the motor controller as shown in figure 4(a). The controller also includes logic to shut the motor off if the robot pitch angle is greater than 20° in either direction.

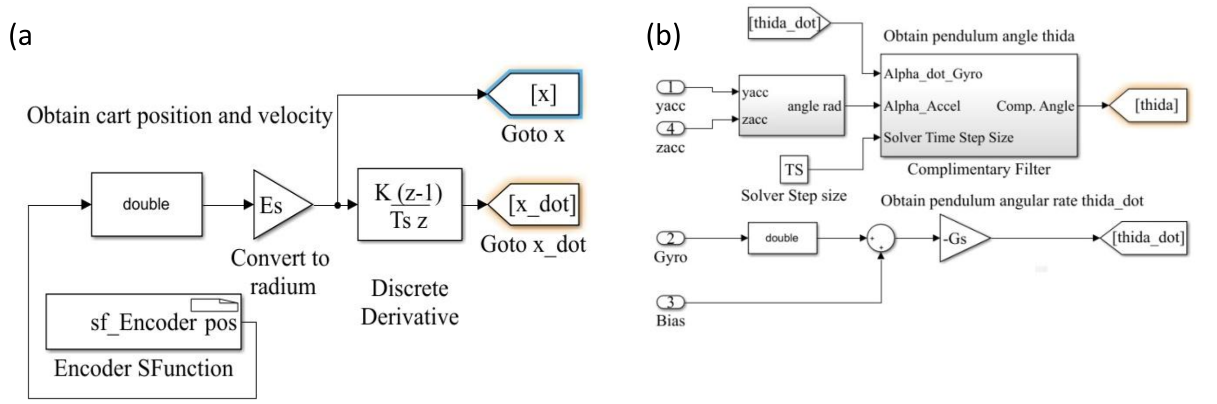


Figure 3 Simulink diagram for obtaining system state variables [$x \ \dot{x} \ \theta \ \dot{\theta}$]

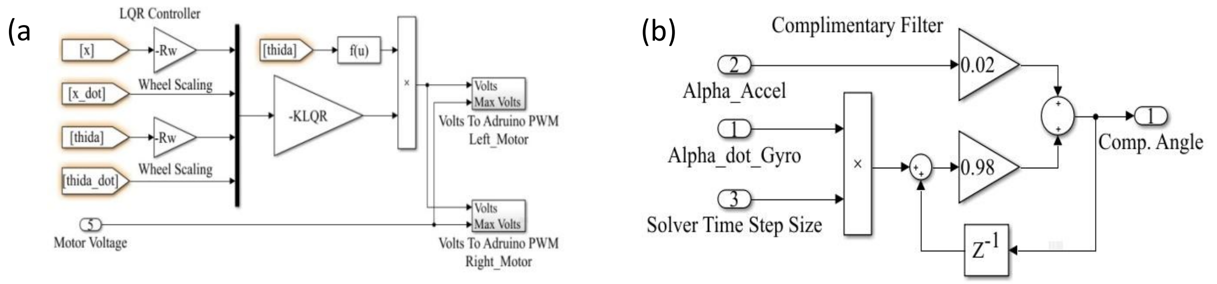


Figure 4 Simulink diagram for LQR controller and Complimentary filter

The desired dynamics system response can be set by adjusting the parameters Q and R values. The values of KLQR depend on both R and Q values. These LQR gain vector will affect the control input signal level, the system response and the robustness of the system.

The control input signal conditioning can be set by adjusting R value. By choosing a large value of R, the system will be operated with less energy i.e. low power consumption. Therefore, it is called expensive control strategy. If the very large value of R is used, the system cannot be responded fast enough to prevent the robot from falling. On the other hand, the system will be responded faster with the choosing of low value R. It is usually called cheap control strategy. The control signal is also become large and chattering. By using very low value of R, the system will become oscillate and unstable. Three different values of R, its corresponding KLQR gain values and the system response are compared in table (3). In this research, the optimum R value can be tuned manually with 0.3 for the constructed two-wheel robot which has the system parameter values given in table (2). Figure 5 (a), (b) and (c) are shown for the real time Simulink graphs of control signal, the pitch angles and the robot position values respectively.

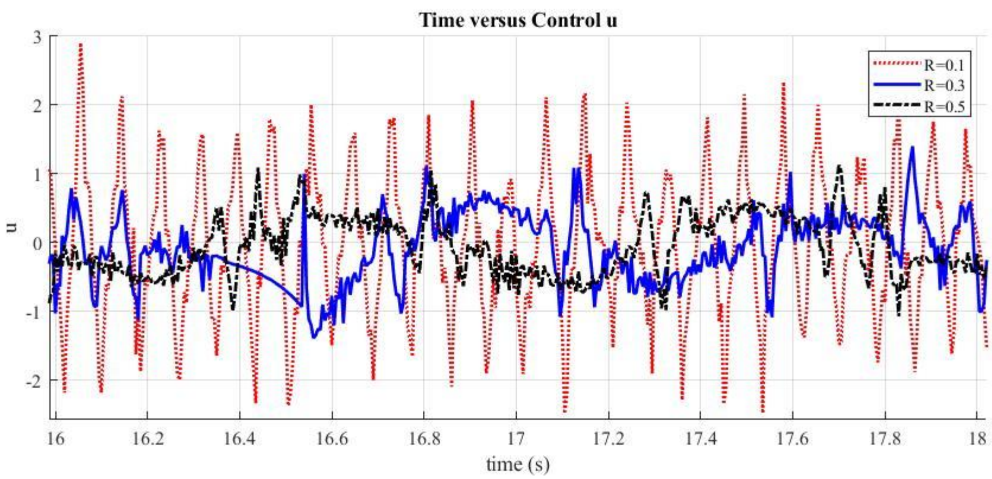
Similarly, the Q values can hardly influence on the system response because it is a weighting function for the system state. In this two-wheel robot, there are four system variables, the position and speed of the based part of robot and the pitch angle and angular rate of the balanced robot. The specific system state response, [$x\ \dot{x}\ \theta\ \dot{\theta}$] can be tuned manually with the corresponding diagonal values of Q matrix, Q(1,1), Q(2,2), Q(3,3) and Q(4,4) respectively. By using larger value of Q, the system will try to stabilize with the least possible changes in the states. Alternatively, using the smaller value of Q will imply less influence about the changes in the states. The choosing for the best values of Q and R is a challenging part in LQR algorithm.

Table 3 The system response for different values of LQR gains

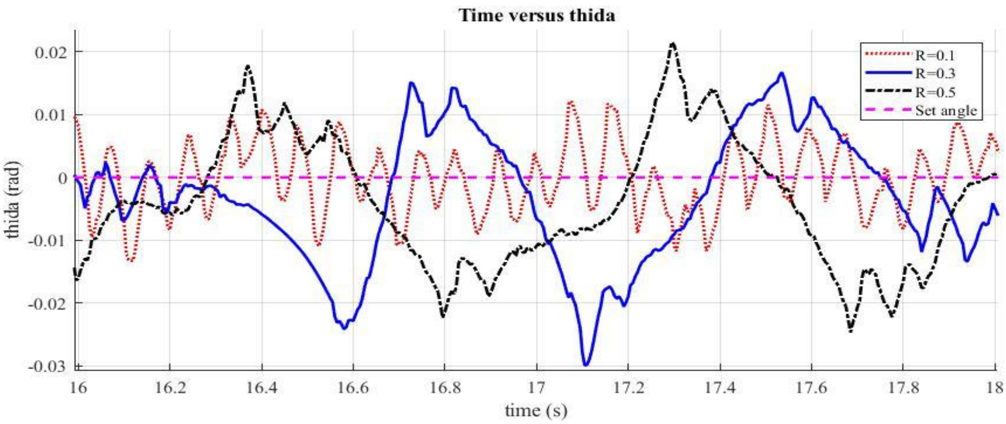
R	KLQR				Control Input	Chattering frequency	System Response
0.1	-3.1623	-4.9191	-27.8789	-4.9002	large	high	aggressive
0.2	-2.2361	-3.5604	-22.7021	-3.7581	↕	↕	↑
0.3	-1.8257	-2.9641	-20.5141	-3.2717			optimum
0.4	-1.5811	-2.6109	-19.2573	-2.9913			↓
0.5	-1.4142	-2.3708	-18.4254	-2.8054	small	low	delay

The real time pitch angle responses and the control signal u for different three diagonal values of Q matrix are compared as shown in figure (6). The small and large Q values are tuned manually to achieve the desired system performance. While the low values of Q (Q = [1, 1, 10, 1]) are used, the control signal is small and the system cannot robust. To obtain the robustness character, the larger Q values (Q = [6, 1, 150 3]) are tuned to compute the controller gain matrix but the system becomes jitter. The figure 6(a) and 6(b) are shown for the control signal and pitch angle response respectively. Since the high frequency switching is occurred at control signal, the robot will be jittered while balancing. This jitter effect may produce noise signal to the microcontroller. The experimental setup for real time simulation with Matlab Simulink and two-wheel balancing robot photos are shown in figure 7(a) and 7(b) respectively.

(a)



(b)



(c)

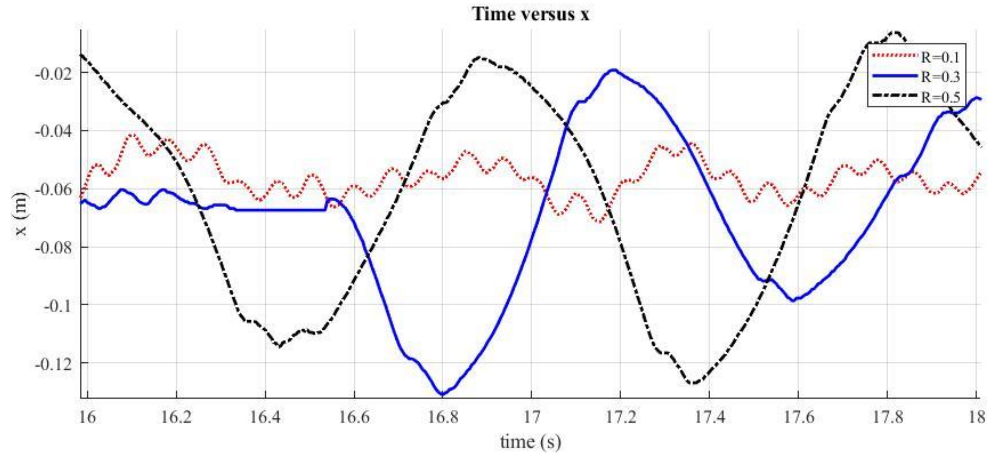
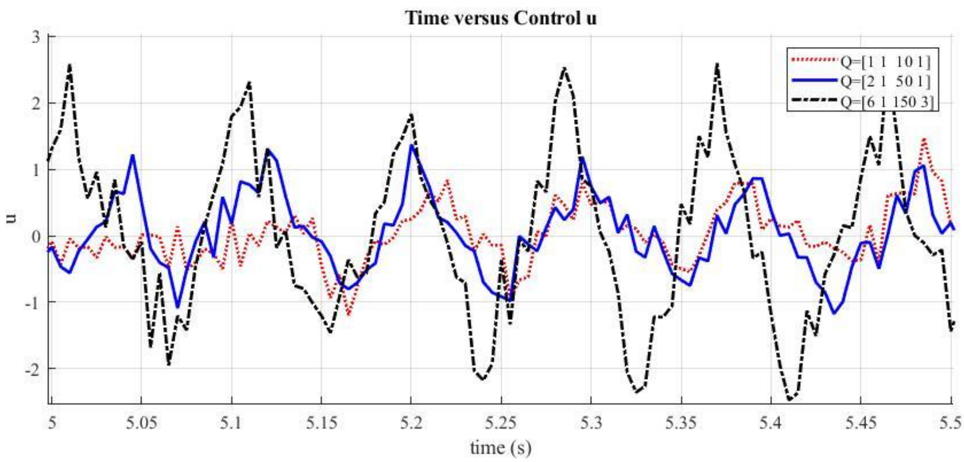


Figure 5 The control signal and system response compare results for three different R values

(a)



(b)

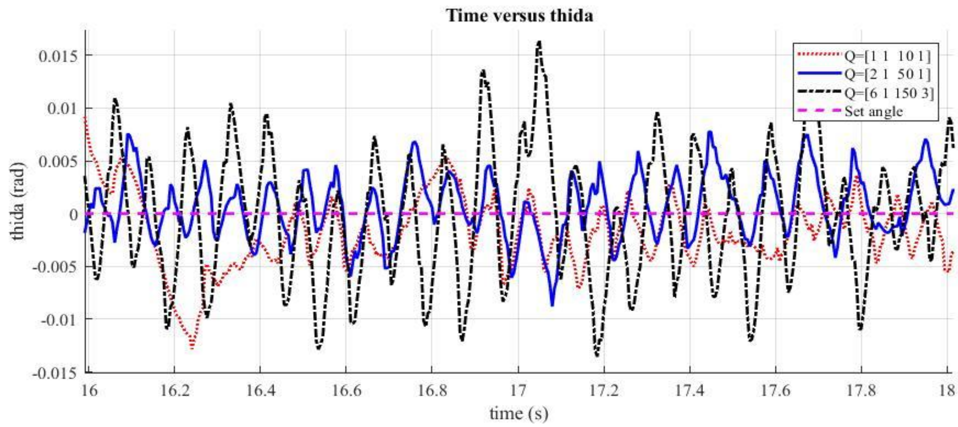


Figure 6 The control signal and system response compare results for three different Q values

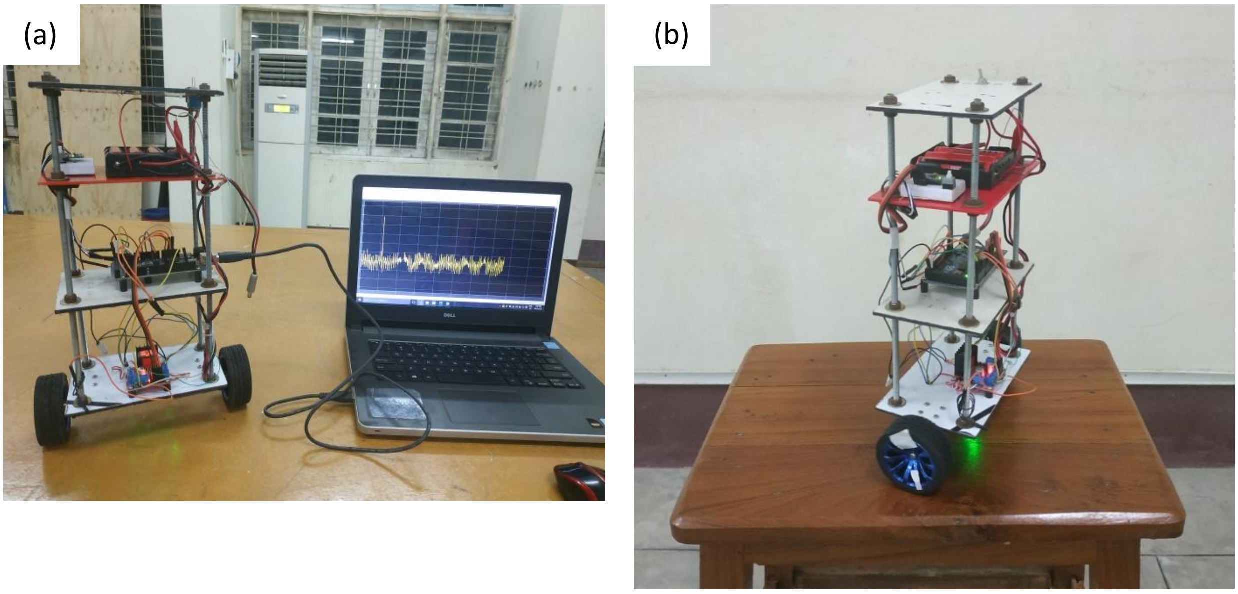


Figure 7 Two-wheel self-balancing balancing robot

Conclusion

In this research, two-wheel self-balancing robot was implemented by using state space control, LQR algorithm. The Rensselaer Arduino Support Package and Simulink support package for Arduino hardware were used not only to program for Arduino mega but also to analyze the performance of controller with real time simulation. Experimental results show that self-balancing can be achieved with LQR control in the vicinity of the upright position. The parameters Q and R values can be used as design parameters to penalize the state variables and the control signals. Since Q and R values are very sensitive variables, they are tuned carefully to provide desired system response. From this research, the MEMS gyro behavior suggests that Kalman filtering would be helpful in attenuating the noise from the gyro data. It is obviously that both sensor selection and signal conditioning are important to the performance of the control systems.

Acknowledgements

The authors would like to acknowledge Professor Dr Khin Khin Win, Head of Department of Physics, University of Yangon, for her valuable suggestion and comments for this work. I would like to thank Professor Dr Nge Nge Khaing, Head of Department of Physics, Taungoo University, for her kind permission to carry out this work.

References

Brian Howard & Linda Bushnell. (2015). Enhancing Linear System Theory Curriculum with an Inverted Pendulum Robot. *American Control Conference(ACC), IEE.*

Katsuhiko Ogata. (2010). *Modern Control Engineering*, Pearson Education, Inc., publishing as Prentice Hall, One Lake Street, Upper Saddle River, New Jersey 07458.

Keerthi Prakash & Koshy Thomas. (2016). A Comparative Study of Controllers for a TwoWheeled Self-balancing Robot. *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering.*

Liu, Jinkan. (2017). *Sliding Mode Control Using MATLAB*, Beihang University, Beijing, China.

Roland S. Burns. (2001). *Advanced Control Engineering*. Butterworth-Heinemann, Linacre House, Jordan Hill, Oxford OX2 8DP, 225 Wildwood Avenue, Woburn, MA 01801-2041, Department of Mechanical and Marine Engineering, University of Plymouth, UK.

Yizhu Li. (2018). *A Comparative Study on Several Control Strategies for Inverted Pendulum Systems*. Master thesis, The Department of Mechanical and Industrial Engineering, Northeastern University, Boston, Massachusetts.