Mechanical and Electrical Design and Fabrication of a Bipedal Walking Robot

A PROJECT REPORT

Submitted in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

in

MECHATRONICS ENGINEERING

by

NAMAN GUPTA

(159105037)

MECHATRONICS ENGINEERING

MANIPAL UNIVERSITY JAIPUR JAIPUR-

303007

RAJASTHAN, INDIA

July/2019

DEPARTMENT OF MECHATRONICS

MANIPAL UNIVERSITY JAIPUR,

JAIPUR – 303 007 (RAJASTHAN), INDIA

Date:

CERTIFICATE

This is to certify that the project titled **MECHANICAL AND ELECTRICAL DESIGN AND FABRICATION OF A BIPEDAL WALKING ROBOT** is a record of the bonafide work done by **NAMAN GUPTA** (159105037) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B.Tech) in **Mechatronics** of Manipal University Jaipur, during the academic year 2018-19.

Prof. Ajay Kumar
Project Guide, Dept of Mechatronics
Manipal University Jaipur

Dr. Shahbaz Ahmed Siddiqui
HOD, Dept of Mechatronics
Manipal University Jaipur

ACKNOWLEDGMENT

I would like to thank Prof. Vivek Sangwan, project instructor, who gave me this wonderful opportunity to pursue the agenda for which I came to IIT Bombay, expecting to learn from 8th semester's internship. He mentored me throughout the internship and taught me both, through researcher's perspective of how a research project is carried out and, through a seller's perspective of how product details should be written in order to help the understanding of a customer. It was his motivation towards learning and blending the classroom knowledge with the project, that kept me addicted to the project and the internship. I have gained a lot of knowledge during internship which changed my point of view to make robots. I am happy that this knowledge won't be wasted and will be utilized in future.

A special thanks to INDUS Lab (Intelligent Dynamics and Ubiquitous Systems Lab) and MAC Lab (Mechatronics Lab) in Department of Mechanical Engineering and CENTRAL LIBRARY for providing me the necessary tools to complete my internship with all the help I needed. Thanks to their faculty members and other students working in the labs to help me with all the technical as well as non-technical matters.

My gratitude towards IRCC (Industrial Research and Consultancy Centre), because of which this entire internship was possible. Thank you IRCC committee for organizing this great platform where I get to meet different kind of people and get to know them as well as the upcoming advancements happening in other research fields.

Sincere thanks to Dr. Ajay Kumar Sir, Professor, Department of Mechatronics Engineering for his mentorship and guidance during the internship. Sincere regards to Dr. Shahbaz Ahmed Siddiqui, Head of Department of Mechatronics Engineering for helping to pursue this internship which will shape my career. Finally, thanks to Manipal University Jaipur and IIT Bombay, because of which, my dream towards robotics came into motion and made me one step closer to achieve it.

ABSTRACT

Robotics is in research since long time and researchers every decade keeps moving one step ahead to develop machines that can substitute humans and replicate human actions. Controlling robots to walk like humans and to perform various acrobatic skills just like a gymnast does, is a popular research area in replicating human actions. Robots like Honda’s Asimo, Atlas by Boston Dynamics are two popular robots that are matching up with the human walking and acrobatic abilities. Acrobatic skills are like ballet dancer’s balancing, where a ballerina balances her body on one leg in a certain posture, and front flip handspring, where an athlete flips continuously in forward direction and then stands stable. The objective is to design, develop and simulate a compass type bipedal walking robot which can perform these two gymnastic skills.

The project started with deriving the equation of motion of a 2-link manipulator using Euler-Lagrange’s equation. Applying PID controller, the fully actuated 2-link manipulator is balanced. To control under-actuated double inverted pendulum, linearization method is employed and using pole placement technique, gain matrix is derived. The robot is designed, considering for flipping skill. The base of the robot is made strong, springs and rods are used to provide reaction force/impact force for every step it takes to move forward.

Throughout the internship various methods are learned that needs to follow when making a research project, Euler-Lagrange’s equation and Ode45 is an interesting and major part in robotics that one must know before building a robot. PID control of double inverted pendulum is done as expected and so is the controlling of the brushed DC motor, which teaches about PWM. The robot is designed and the locking mechanism is fabricated using CNC machines.

MATLAB is used to perform simulations of the under-actuated double inverted pendulum, Fusion360 is used to design the compass type bipedal walking robot and Freescale CodeWarrior is used to code Freescale 9S12XEP100 micro-controller to control brushed DC motor RMCS-3017.

LIST OF TABLES

LIST OF FIGURES

CONTENTS

Chapter-1
INTRODUCTION

A robot is a machine which has the ability to carry out complicated actions automatically. A robot can be guided with an external control unit or the controller may be embedded in it. Initially, robots were constructed to look exactly like humans which are called as humanoids. But, as we move ahead in the evolution of robotics, the design of the robots started to change according to the task with no regard how they look.

A humanoid robot is a robot which has its body shaped like the human body. The design varies according to the purpose it has been made for, like for functional purpose, as interacting with humans and environments, for experimental purposes, such as the study of bipedal locomotion, or for other external purposes. Usually, humanoid robots are like humans, though some forms of humanoid robots only have part of a body, for example, from the waist down.

Bipedal locomotion is a terrestrial locomotion where an organism moves by its two legs. A robot that usually moves in a bipedal manner is known as a bipedal robot. Bipedal movement includes walking, running, hopping, flipping. A bipedal mechanical system which is composed of only two links is a Compass type Bipedal Robot.

Robots have mimicked humans in performing various terrestrial locomotion like walking, running, jumping etc.., and has also performed tasks which humans are unable to do and especially which take place in extreme environments such as outer space or the bottom of the sea. Considering in acrobatics, the robot gymnastics is a special research area where most the humanoid organizations are working like Honda’s Asimo, Boston Dynamics’ Atlas, Nao and many other robots.

A handspring is an acrobatic move in which an acrobat executes a complete flip of the body by leaping headfirst from a standing position into an upside-down vertical position and then springing off from the floor with the hands leaping back to a standing position. The direction of rotation of the body can be either in forward or backward direction, and any of it can be performed. A ballet dance balancing is another difficult task which is performed by ballet dancers as shown in the image below. The balancing is done using hip joint as an actuator and controlling only one leg while the other leg is balanced on toes. This is an exact resemblance of an under-actuated double inverted pendulum which has an actuator between the two links.

Fig-1.1: Handspring Skill

Fig-1.2: Ballet Dancer Balancing

This chapter then proceeds to briefly explain the motivation behind the research and the objective of the project.

1. MOTIVATION

Bipedal Walking Robots are usually made to mimic human actions and often used to implement gymnastic and aerobic skills to make them as robust as a human being is. One of the popular gymnastic skill is ballerina's balancing method that is seen in ballet dancing. Using that concept, the robot is designed and balanced using an actuator at the revolute joint between both the links of double inverted pendulum. The basic idea behind this project is to teach the intern regarding the dynamics, control and design of bipedal robot and to tackle the problem through the blend of research experience and classroom knowledge.

2. OBJECTIVE OF THE PROJECT

The main objective of this project is to design, develop, fabricate and simulate a compass type bipedal walking robot which can perform flipping maneuver. The project is divided into two major sections, mechanical design & fabrication, and controlling of the robot.

To perform the flipping maneuver, first goal is to perform ballerina's balancing skill which requires the study of under-actuated double inverted pendulum. So, the project first flows to perform balancing skill and then moves ahead to flipping maneuver.

Chapter-2
BACKGROUND MATERIAL

This chapter discusses about theory and concept about the methodology and the basic science behind the technology and tools used in developing the prototype and controlling the bipedal robot to perform the task

1. LITERATURE

Robots are now used as research tool for scientific interest in various research areas. Researchers study the human body and its behavior to build robust robots. To get the better understanding of it, human's locomotion style is simulated as studied in *Biped Gymnastics*. In a paper, S*wing-up control of an acrobot,* explains the controlling of an under-actuated double inverted pendulum where the actuator is placed in the between the two links as the revolute joint. The acrobot moves from the bottom to upright position with only one actuator which is exactly like an acrobat on a bar where he swings his/her legs using their hip and swings right up in vertically inverted position. This is how a ballerina's balancing is controlled and simulated in the project using the concept of Euler-Lagrange's equation of motion, linearization of a non-linear equation and applying PID controller to control the position. These topics are explained later in this chapter.

Handspring as explained in introduction is a gymnastic skill which looks like a bipedal robot without knees as seen in fig-1. The bipedal robot which does not has knees looks like a compass used in geometry and is also called as Compass based Bipedal Robot as shown in figure below.

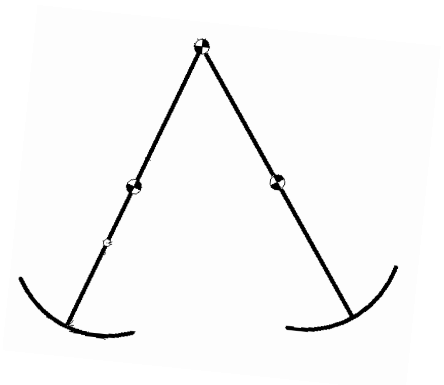


Fig-2.1: Compass Bipedal Robot

The compass bipedal robot has one revolute joint at its hip, on which a motor is connected which is used to control the robot. Before the compass bipeds were used for passive dynamics, where the dynamical behavior of an actuator or a robot is not drawing any energy from a supply. The original model for passive dynamics is based on human's walking motions down the slope which is an under actuated system.

Fully actuated systems, such as the Honda Asimo robot, is not very efficient because each joint has a motor and control assembly which consumes huge amount of power and energy. Human-like gaits are more efficient because their movement is performed by the natural swing of the legs instead of actuating each joint. The human-like gaits are under-actuated systems.

Underactuation means that there is passive dynamics present in the walking motion of the robot, which implies that there are joints that cannot be directly controlled through the system's actuators. It means that there are parts of the systems that cannot be directly actuated and continuously controlled. In such machines, you cannot directly and independently affect both the position and orientation.

2. CONCEPTUAL OVERVIEW

2.1 EQUATION OF MOTION

Equations of motion describe the behavior of a physical system as a function of time in terms of its motion, specifically, as a set of mathematical functions in terms of dynamic variables. There are two main kinds of motion: dynamics and kinematics. Dynamics refers to the differential equations that the system satisfies, for example, Newton's second law or Euler–Lagrange equations. The equation of motion is important to consider in the designing the robots, and plays a major role in simulation and animation, and in the design of control algorithms.

The dynamics equation of motion of the robot is derived using Euler-Lagrange's equation of motion. It describes a general procedure for deriving the dynamics of mechanical systems and the equation of motion of the system, where the system can be described through a set of generalized coordinates. In general, for any system of the type considered, an application of the Euler-Lagrange equations leads to a system of n coupled, second order nonlinear ordinary differential equations of the form

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i; \qquad i = 1,2,\ldots\ldots,n$$

Where the order, n, of the system is determined by the number of generalized coordinates that are required to describe the evolution of the system. **L** denotes the Lagrangian and is represented by

$$L = K - P$$

K is the Kinetic Energy and **P** is the Potential Energy of the system. $\dot{\mathbf{q}}$ and $\mathbf{q}$ are the generalized coordinates for every revolute joint in the robot. $\boldsymbol{\tau}$ is the required torque for every respective '**i**' joint.

2.2 PID CONTROLLER

A PID controller, is a three term controller which stands for Proportional Integral Derivative controller, is a control loop feedback mechanism which is commonly used in control systems and a many of other applications that requires continuously modulated control. A PID controller constantly calculates an error value **e(t)** as the difference between a desired setpoint and a measured point and applies a correction based on PID gain constants.

A daily example is the cruise control of a car, when a car is ascending a hill where its speed lowers while providing a constant engine power. PID plays its role there, it restores the measured speed to desired speed with nominal delay and overshoot in speed by increasing the engine power using PID algorithm.

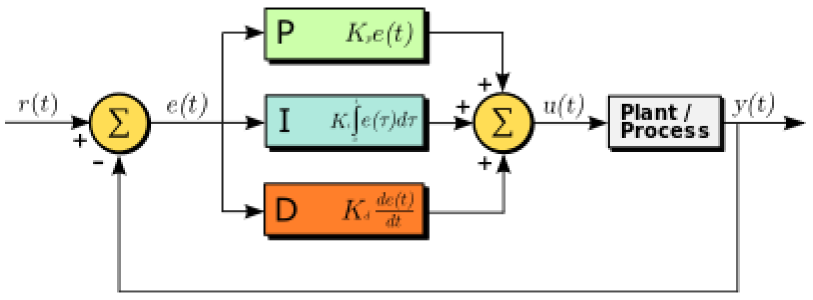


Fig-2.2: Block diagram of PID controller

The above block diagram shows how a PID controller is applied to a Plant. In this, the e(t) is the error value which is the difference between the reference input/desired value and the output/current value. r(t) is reference value, y(t) is the output and u(t) is the control input which is provided to the plant and is of the form

$$u(t) = P + I + D;$$

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$$

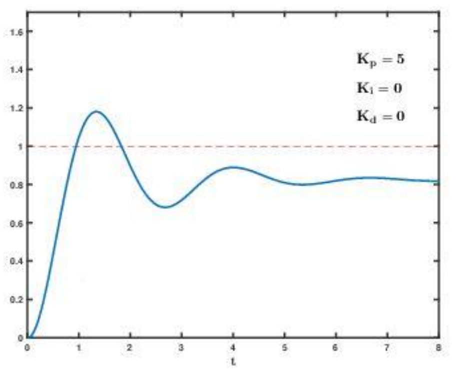
Here **Kp** is proportionality constant, **Kd** is derivative constant and **Ki** is integral constant, **t** is the time variable. The controller attempts to reduce the error over time by adjustment of a control variable **u(t)** to a new value determined by a weighted sum of the control terms.

2.2.1 Understanding PID

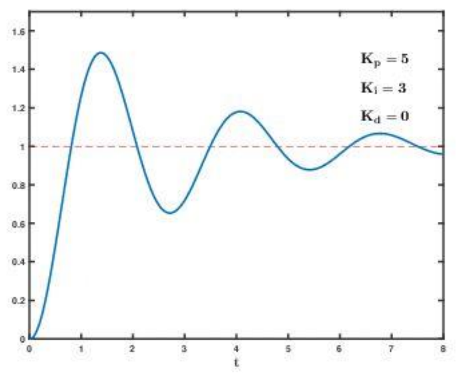
- **P** is proportional to the current value of the error *e(t)* which is setpoint − presentvalue. If the error is positive and large, the control output will be proportionately large and positive, taking into account the proportionallity gain factor ***Kp***. But using only proportional controller will result in an error between the setpoint and the actual value, because it requires an error to generate the proportional response, if there is no error then there is no corrective response as shown in Fig2.3(a).

- **I** is integral term which books for past values of the error *e(t)* and integrates them over time to produce the new I term. When there is a remaining error even after the proportional controller is applied, the integral term seeks to eliminate the remaining error by adding a control effect due to the past cumulative value of the error. When the error is eliminated, the integral term ***Ki*** will cease to grow and results in the diminishing of the proportional effect as the error decreases, but is compensated by the growing integral effect as shown in Fig-2.3(b).

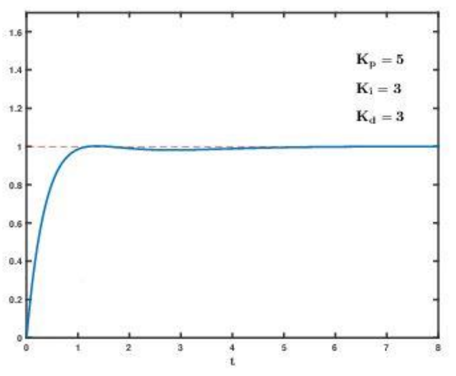
- **D** is derivative term which estimates the future values of the error *e(t)*, based on its current rate of change. It effectively seeks to reduce the effect of the error by using a control parameter ***Kd*** generated by the rate of change of error. The more rapid change, the better is controlling effect as shown in Fig-2.3(c).



(a) P Controller



(b) PI Controller



(c) PID Controller

Fig-2.3 Tuning of PID Controller

2.2.2 PID in Robotics

The PID controller is commonly used in Robotics, due its powerful performance and its simplicity, especially for beginners in line follower robot. Practrical implementation of PID controllers is much more different and elaborated process. In practical, saturation is used right after the PID control input to limit the input, because most of the time the torque as per the controller exceeds the rated torque of the motor. Another major point is its tuning, that is not easy as it seems to be, for the project I have done linearization which will be explained in the next topic and then found the gaining parameters to tune the non-linear model in MATLAB. Therefore, PID tuning is essentially an engineering art that cannot only rely on automated processes but requires the experience of the designer.

2.3 LINEARIZATION

Linearization is finding the linear approximation of a function at its equilibrium point. The linear approximation of a function is found using the first order Taylor series expansion around the equilibrium point. Taylor series expansion for two variables is of the form

$$f(x+\delta x, y+\delta y) \approx f(x,y) + \left.\frac{\partial f}{\partial x}\right|_{x,y} \delta x + \left.\frac{\partial f}{\partial x}\right|_{x,y} \delta y$$

Almost every system is non-linear in nature, the system is approximated by a linear system of equations by taking certain assumptions. One of the main assumptions is that the system's postion and velocity are low and the external perturbations are close enough to the equilibrium points.

2.4 PULSE WIDTH MODULATION

Pulse Width Modulation, or PWM, is a method of getting analog signals using the means of digital signals. A digital signal is switched between on and off to create a square wave which is an analog signal,. This on-off pattern is in between HIGH on (5 Volts) and LOW on (0 Volts), and is varied by changing the time on which the signal is ON and OFF. Pulse width is the duration of "on time", to get varying analog values, pulse width is varied and modulated.

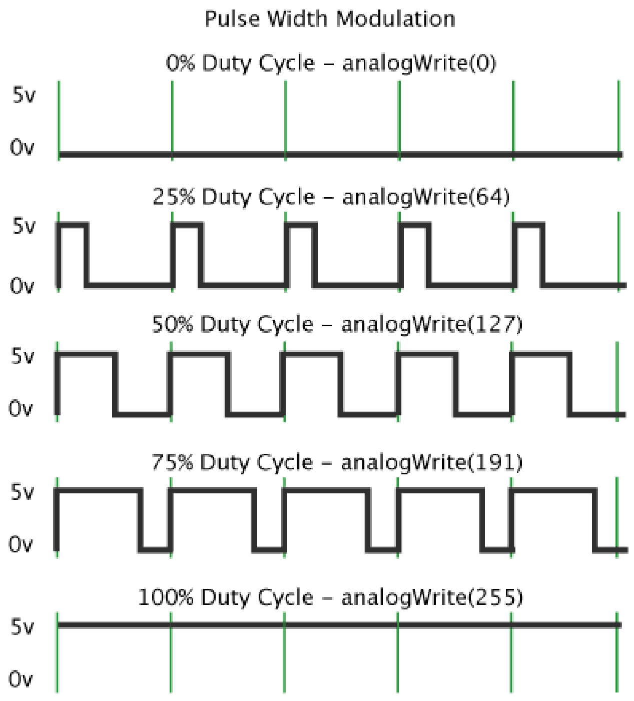

Pulse Width Modulation
0% Duty Cycle – analogWrite(0)
5v
0v
25% Duty Cycle – analogWrite(64)
5v
0v
50% Duty Cycle – analogWrite(127)
5v
0v
75% Duty Cycle – analogWrite(191)
5v
0v
100% Duty Cycle – analogWrite(255)
5v
0v

Fig-2.4: Types of PWM signals

As shown in figure 2.4, the pulse width is modulated by varying the ON time and OFF time. The duty cycle is term used here, which means that the time during which the machine is operating in intermittent manner.

3. TECHNOLOGY USED

In this topic, we discuss about the technologies used to make the prototype and tools that are used in the prototype of the biped as per the objective.

3.1 ODE45

ODE stands for Ordinary Differential Equation, which is a standard solver for ode in MATLAB. Ode45 function is the commonly used solver and is recommended for beginners to start with, when solving ODEs. This function uses Runge-Kutta method with a variable time step for the fast and efficient computation. But, the problem is often not completed till once you have solved the question and obtained the ode's governing the systems motion. It is often recommended to produce a visual representation of what exactly the trajectories are represented by a highly complicated looking ordinary diferential equation looks like. The syntax for ode45 is

$$[t, y] = ode45(odefun, tspan, y0, options)$$

where *odefun* is the function where differential equation is written, *tspan* is the total time starting from t_0 (initial time) to t_f (final time), *y0* is the initial conditions of the system (in this project, position and velocity of each joint of the robot) and *options* is a structured array. Using odeset function, one can create or modify the options structure. Its syntax is

$$options = odeset('RelTol', 1e^{-5}, 'AbsTol', 1e^{-8})$$

RelTol is the relative tolerance and *AbsTol* is the absolute tolerance.

3.2 ENCODER

An encoder is an electronic transducer that converts or encodes information from one cypher to another for ease of enigmas to follow same machine code. It senses a position or orientation and use it as a reference to detect and control position as feedback to the systems.

3.2.1 Rotary Encoder

A rotary encoder, also knows to be a shaft encoder, converts angular position to digital signal. There are two types of rotary encoder, absolute and incremental. Rotary encoders are used in many applications that requires control of mechanical systems, for example, industrial controls, robotics, computer devices, etc..

The absolute encoder provides signal that gives a distinct position within the travel range without requiring any data of previous position. On the other hand, the signal from an incremental encoder is ambiguous and requires cycle counting to maintain absolute position. Both the encoders provide same accuracy but the absolute encoder is more robust to noise, whereas increamental encoder gives change in position in real time.

3.2.2 Quadrature Encoder

The quadrature encoder is a type of increamental encoder used to sense the direction of the movement as well as the position.

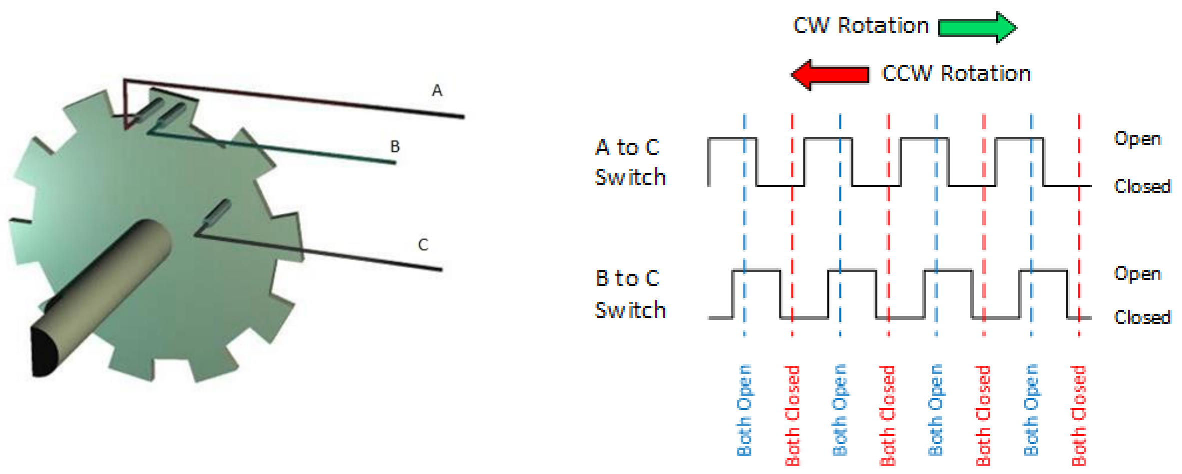


Fig-2.5: Working of Quadrature Encoder

In the above figure, *A* and *B* are encoder channels and *C* is GND. As shown above, when the encoder is moving in clockwise direction, pulses are flowing from channel A to channel C and when encoder is moving in anti-clockwise direction, pulses are flowing from channel B to channel C. The encoder provides 5V of pulse output and the phase difference between the channel A and channel B is of 90°.

3.3 DC MOTOR

A DC motor is a electrical machine that converts electrical energy supplied from DC supply into rotational mechanical energy. DC motors are widely used, because they can be powered from any existing DC power systems according to the required voltage.

3.3.1 Brushed DC Motor

The brushed DC electric motor generates torque directly provided from DC power supply, by using inner commutation, stationary permanent magnets and rotating electromagnets. Brushed DC motors have low initial cost, high reliability, and easy motor speed control but they require high maintenance and have low life-span for high intensity use.

In the below figure-2.5, the brushed DC motor has a two-pole armature and a permanent magnet stator. ***N*** and ***S*** are polarities of the magnets on the inside axis faces, but the outside faces have opposite polarities. The **+** and **-** signs are the commuter's DC current supply which shows where the DC current is applied to the commutator for current supply to the armature coils.

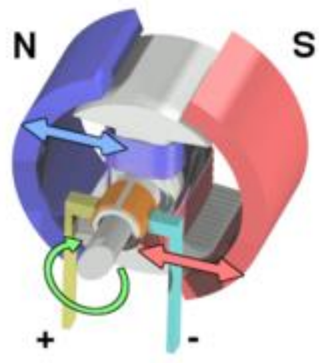


Fig-2.6: Working of Brushed DC Motor

Fig-2.7: Solenoid Lock

3.4 SOLENOID LOCK

An solenoid lock is a locking device that consists of an electromagnet and an armature plate. The solenoid lock is like a latch but for electronic locking and unlocking. It is available in different kinds of modes, in unlocking in the power-on mode type, and locking and keeping in the power-on mode type, which can be used selectively for situations. Fig-2.6 shows the image of a solenoid lock used in the prototype.

3.5 LOAD CELL

A load cell is a transducer that is used to measure weight whose magnitude is directly proportional to the electrical signal created by it. There are various types of load cell, but the one used in the project is a strain gauge type.

Fig-2.8: Compression Load Cell

Chapter-3
METHODOLOGY & IMPLEMENTATION

The project is divided into two functional parts, Control and Mechatronic Design. The mechatronic design is CAD of the prototype.

1. CONTROL

The control is implemented on the double inverted pendulum which goes through the step by step procedure.

1.1 DERIVING EQUATION OF MOTION

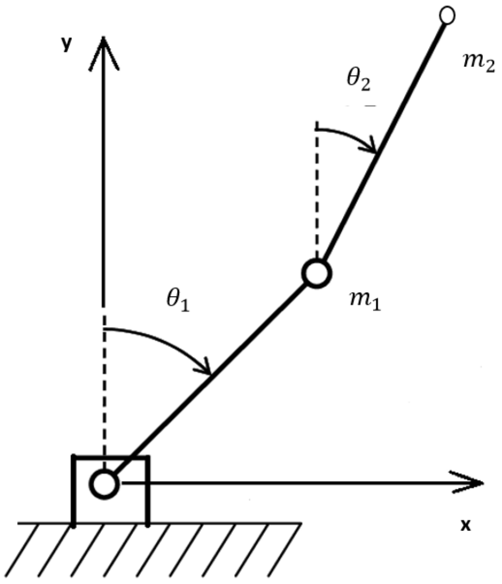


Fig-3.1: Double Inverted Pendulum

In figure 3.1, m_1 and m_2 are centre of mass of both the links respectively. θ_1 and θ_2 are the angles with respect to y axis. L_1 and L_2 are length of both the links from the joint-1 to joint-2 and joint-2 to end effector respectively. Here,

$$p_1 = \begin{bmatrix} L_1 sin\theta_1 \\ L_2 cos\theta_1 \end{bmatrix}; \qquad p_2 = \begin{bmatrix} L_1 sin\theta_1 + L_2 sin\theta_2 \\ L_1 cos\theta_1 + L_2 cos\theta_2 \end{bmatrix};$$

$$v_1 = \begin{bmatrix} L_1\dot{\theta}_1 cos\theta_1 \\ -L_1\dot{\theta}_1 sin\theta_1 \end{bmatrix}; \qquad v_2 = \begin{bmatrix} L_1\dot{\theta}_1 cos\theta_1 + L_2\dot{\theta}_2 cos\theta_2 \\ -L_1\dot{\theta}_1 sin\theta_1 - L_2\dot{\theta}_2 sin\theta_2 \end{bmatrix};$$

$$K.E._1 = \frac{1}{2} m_1 v_1^T v_1; \qquad K.E._2 = \frac{1}{2} m_2 v_2^T v_2;$$

$$P.E._1 = m_1 g L_1 cos\theta_1; \qquad P.E._2 = \frac{1}{2} m_2 g (L_1 cos\theta_1 + L_2 cos\theta_2);$$

$$K = K.E._1 + K.E._2\,; \qquad P = P.E._1 + P.E._2\,;$$

Here p_1 and p_2 are position vectors, v_1 and v_2 are velocity vectors, $K.E._1$ and $K.E._2$ are Kinetic Energy and, $P.E._1$ and $P.E._2$ are Potential Energy of both the links respectively. K is total kinetic energy and P is total potential energy of the double inverted pendulum.

Using Euler Lagrangian's method of solving equation of motion, we get

$$L = K - P; \qquad \frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i;$$

Where *i* represents the number of joints. There are two revolute joints in double inverted pendulum. On solving the above equation, we get

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + G(\theta) = \tau; \qquad \text{-------------} \ (3.1)$$

Where, $$M(\theta) = \begin{bmatrix} (m_1 + m_2)L_1^2 & m_2 L_1 L_2 \cos(\theta_1 - \theta_2) \\ m_2 L_1 L_2 \cos(\theta_1 - \theta_2) & m_2 L_2^2 \end{bmatrix};$$

$$C(\theta, \dot{\theta}) = \begin{bmatrix} m_2 L_1 L_2 \sin(\theta_1 - \theta_2)\,\dot{\theta}_2^2 \\ -m_2 L_1 L_2 \sin(\theta_1 - \theta_2)\,\dot{\theta}_1^2 \end{bmatrix}; \qquad G(\theta) = \begin{bmatrix} -(m_1 + m_2) g L_1 \sin\theta_1 \\ -m_2 g L_2 sin\theta_2 \end{bmatrix};$$

Equation (3.1) can be written as

$$\ddot{\theta} = \frac{-C(\theta, \dot{\theta}) - G(\theta) + \tau}{M(\theta)};$$

The equation has to be verified and it goes through the step by step procedure. First, the total energy is derived which is written as

$$E = K + P;$$

Here, $$K = \frac{1}{2} m_1 L_1^2 \dot{\theta}_1^{\,2} + \frac{1}{2} m_2 \left(L_1^2 \dot{\theta}_1^{\,2} + L_2 \dot{\theta}_2^{\,2} + 2 L_1 L_2 \cos(\theta_1 - \theta_2)\,\dot{\theta}_1 \dot{\theta}_2 \right);$$

$$P = m_1 g L_1 cos\theta_1 + m_2 g (L_1 cos\theta_1 + L_2 cos\theta_2);$$

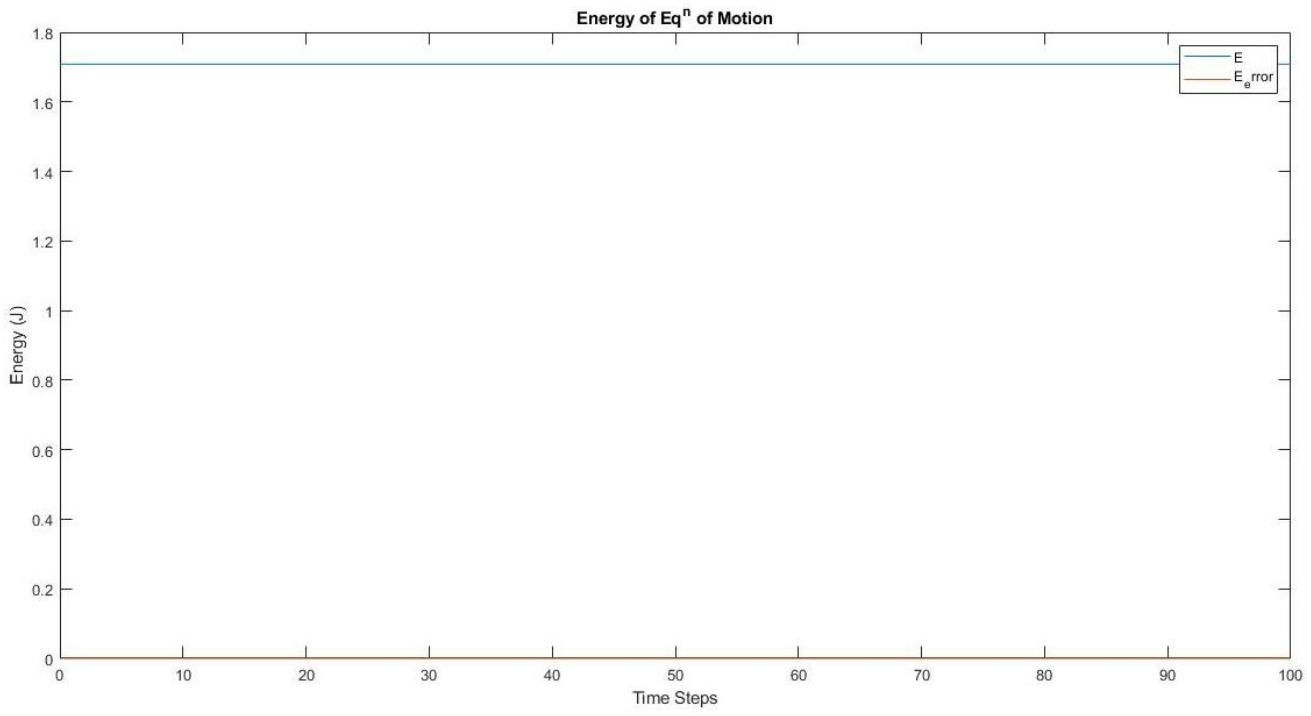


Fig-3.2: Energy at angles (θ_1= 5°,θ_2= -5°)

Figure 4.1 explains that the energy is at 1.7083 J and is constant throughout the simulation, which means that the equation of motion is correct when no torque is applied initially, i.e.,

$$\tau = [0\ 0]^T;$$

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta}) + G(\theta) = 0;$$

1.2 POWER EQUATION

After testing energy equation, power equation is tested by giving torque input. Sinusoidal and cosinusoidal inputs are given at torque of joint-1 and joint-2 respectively.

$$\tau = [\sin(t)\ \ \cos(t)]^T$$

There are two types of power equation, first one is simply differentiating energy equation with respect to time and that is written as

$$Power = \frac{dE}{dt};$$

$$\begin{aligned} Power = {} & m_1 L_1^2 \dot{\theta}_1 \ddot{\theta}_1 + m_2 (L_1^2 \dot{\theta}_1 \ddot{\theta}_1 + L_2^2 \dot{\theta}_2 \ddot{\theta}_2 + L_1 L_2 (\cos(\theta_1 - \theta_2) \left(\dot{\theta}_1 \ddot{\theta}_2 + \ddot{\theta}_1 \dot{\theta}_2\right) \\ & + \dot{\theta}_1 \dot{\theta}_2 \sin(\theta_1 - \theta_2) \left(\dot{\theta}_1 - \dot{\theta}_2\right)) - m_1 g L_1 sin\theta_1 \dot{\theta}_1 - m_2 g \left(L_1 sin\theta_1 \dot{\theta}_1 + L_2 sin\theta_2 \dot{\theta}_2\right); \end{aligned}$$

The other type of power equation is derived by using torque-power relation, which is

$$Power = \tau_1 \dot{\theta}_1 + \tau_2 \dot{\theta}_2;$$

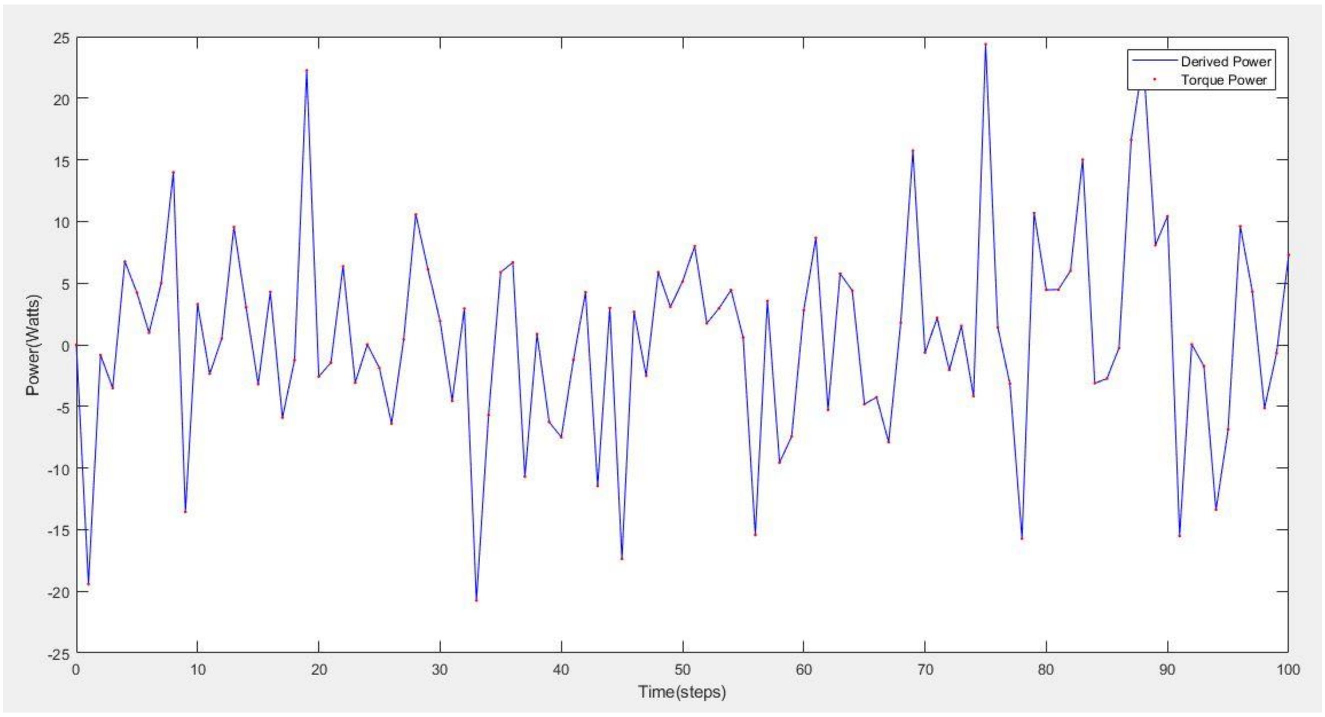


Fig-3.3: Verfication through Power

In figure 3.3, the power derived through differentiating energy and the power provided by torque input are overlapping each other, this confirms that the derived equation of motion is correct. The code for the verification of equation of motion is written in Annexure 1.1.

1.3 PID CONTROL

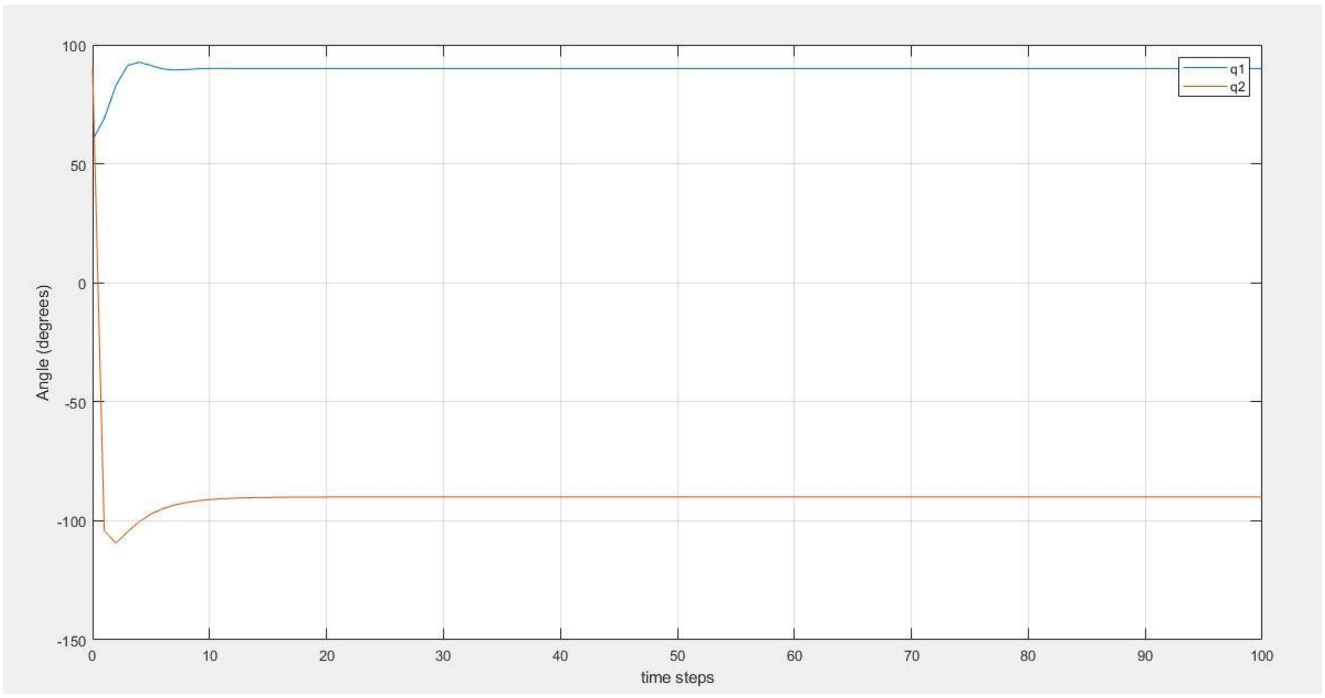


Fig-3.4: PID Control

$$\tau = M(\theta) * U;$$

$$U = K_p(e) + K_d(\dot{e}) + K_i \int e\, dt; \qquad\qquad e = \theta_{des} - \theta;$$

In PID control, using gain parameters the angles are controlled. The gain parameters are set as $K_{p_1} = K_{p_2} = 15, K_{d_1} = K_{d_2} = 10, K_{i_1} = 10$ and $K_{i_2} = 7$. Figure 4.3 displays the position of joint-1 and joint-2 when PID is applied at torque of both joints respectively. Here, *q1* and *q2* are angles of the joint-1 and joint-2 respectively. The code for PID control is written in annexure 1.2.

The next step is to control underactuated double inverted pendulum where the motor/actuator is at joint2 only. For this linearization technique is implemented to determine the gain parameters that are required to control the pendulum.

1.4 CONTROL OF UNDER-ACTUATED DOUBLE INVERTED PENDULUM

To find the tuning parameters of PID for joint-2, instead of hit and try, linearization method is used to determine the gain matrix of the underactuated double inverted pendulum, where the torque is applied at joint-2 only.

We want to hold 2-R manipulator at a certain position q_0, then we assume that the result of any external disturbance is small. Thus, we apply taylor series expansion about the stable point q_0.

$$f(x + \delta x, y + \delta y) \approx f(x, y) + \left.\frac{\partial f}{\partial x}\right|_{x,y} \delta x + \left.\frac{\partial f}{\partial x}\right|_{x,y} \delta y$$

Since, the system is at equilibirum at q_0, $\dot{q} = 0$, $\ddot{q} = 0$. Therefore,

$$\delta \ddot{q} = \left.\frac{\partial \alpha}{\partial q}\right|_{q_0,0} \delta q + \left.\frac{\partial \alpha}{\partial q}\right|_{q_0,0} \delta \dot{q} + \left.\frac{\partial \beta}{\partial q}\right|_{q_0,0} \delta q \tau_0 + \beta(q_0)\delta\tau_0;$$

Here, $\alpha(q, \dot{q}) = M(q)^{-1}(-C(q, \dot{q}) - G(q))$ and $\beta(q) = M(q)^{-1}$.

Now, the above equation is written in the form of $\dot{x} = Ax + Bu$, where

$$A = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \frac{\partial f_1}{\partial x_3} & \frac{\partial f_1}{\partial x_4} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \frac{\partial f_2}{\partial x_3} & \frac{\partial f_2}{\partial x_4} \\ \frac{\partial f_3}{\partial x_1} & \frac{\partial f_3}{\partial x_2} & \frac{\partial f_3}{\partial x_3} & \frac{\partial f_3}{\partial x_4} \\ \frac{\partial f_4}{\partial x_1} & \frac{\partial f_4}{\partial x_2} & \frac{\partial f_4}{\partial x_3} & \frac{\partial f_4}{\partial x_4} \end{bmatrix}; \quad B = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_3}{\partial u_1} & \frac{\partial f_3}{\partial u_2} \\ \frac{\partial f_4}{\partial u_1} & \frac{\partial f_4}{\partial u_2} \end{bmatrix}; \qquad x = [q_1\ q_2\ \dot{q}_1\ \dot{q}_2]; \quad u = [0\ \tau_2];$$

Using Pole Placement technique, the gain matrix is determined which is written as

$$K = [\alpha_4 - a_4\ \alpha_3 - a_3\ \alpha_2 - a_2\ \alpha_1 - a_1]T^{-1};$$

$$T = MW;$$

$$M = [B \vdots AB \vdots AB^2 \vdots AB^3]; \qquad W = \begin{bmatrix} a_{n-1} & a_{n-2} & \cdots & a_1 & 1 \\ a_{n-2} & a_{n-3} & \cdots & 1 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ a_1 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{bmatrix};$$

where M is the controllability matrix and a_is are coeficients of the characteristic polynomial

$$|sI - A| = s^n + a_1 s^{n-1} + \cdots + a_{n-1}s + a_n;$$

Substituting *m1=m2=1*, *L1=L2=1* and *g=9.8*, the non-linear equation of motion is linearized using taylor's series which generates the equation of the form

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

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 19.6 & -9.8 & 0 & 0 \\ -19.6 & 19.6 & 0 & 0 \end{bmatrix}; \qquad B = [0 \quad 0 \quad -1 \quad 2]^T;$$

$$x = [q_1 \quad q_2 \quad \dot{q}_1 \quad \dot{q}_2]^T; \qquad u = \tau_2;$$

Using pole placement method, the desired poles are set and the gain matrix is

$$K = [-1640 \quad -448 \quad -500 \quad -204];$$

Here K_{p_1}= -1640, K_{p_2}= -448, K_{d_1}= -500 and K_{d_2}= -204.

Substituting $u_2 = -(K_{p_1} e_1(t) + K_{d_1} \dot{e_1}(t))$, we can control the first link of the pendulum. This concept is also called as Reaction wheel pendulum, where a rotating wheel is at the end of a link and actuated by DC motor. The pendulum is unactuated and is controlled by the reaction wheel only.

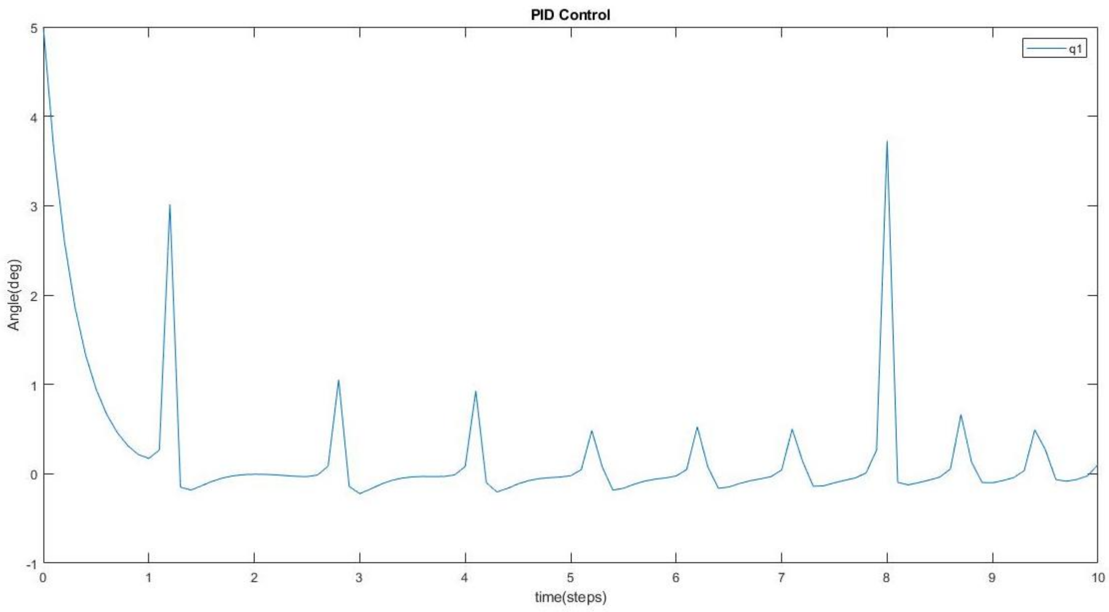


Fig-3.5: Reaction Wheel Pendulum Control

In figure 4.4, the reaction wheel which is our 2nd link, it balances the 1st link. The spikes means a slight deviation for very instant of time to balance itself. The code is written in appendix 1.3.

2. MECHATRONIC DESIGN

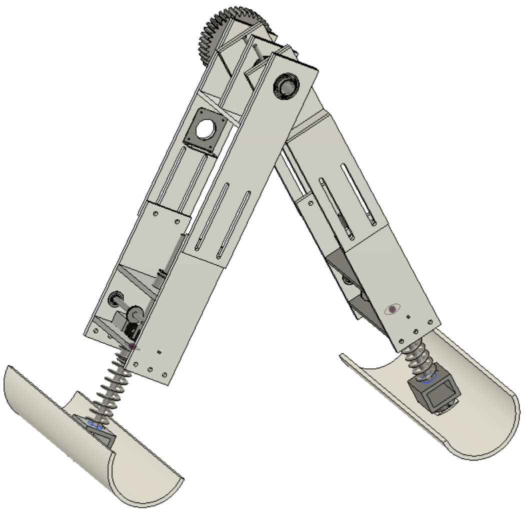


Fig-3.6: Design of Bipedal Walking Robot

2.1 FOOT

The material used for semi-circular cylinder is PVC pipe. The reason to use PVC pipe is because it is denser as compared to other plastics, very rigid, very good tensile strength and readily available in cheap rates. These features support the foot, because, when the robot flips the impact of the whole body will be affected on the foot and PVC is effective to use for that purpose, as it won't break easily.

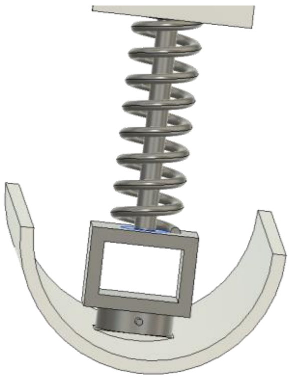


Fig-3.7: Foot Design

As shown in the above figure 3.3, compression load cell is fixed on the PVC base. A compression load cell is used to measure the weight applied normally to it and converts to an electrical signal.

Right above, the load cell is a 3D printed hollow cuboid of thickness 5mm which is used to connect the delrin rod and the compression load cell. Compression spring is used for the reaction force when the robot takes its next flip, it stores the energy as the robot takes a step and gets locked by the locking mechanism, then when the other leg is flipped and as soon as it touches the ground, the spring is unlocked and using the stored energy the other leg is flipped.

2.2 LOCKING MECHANISM

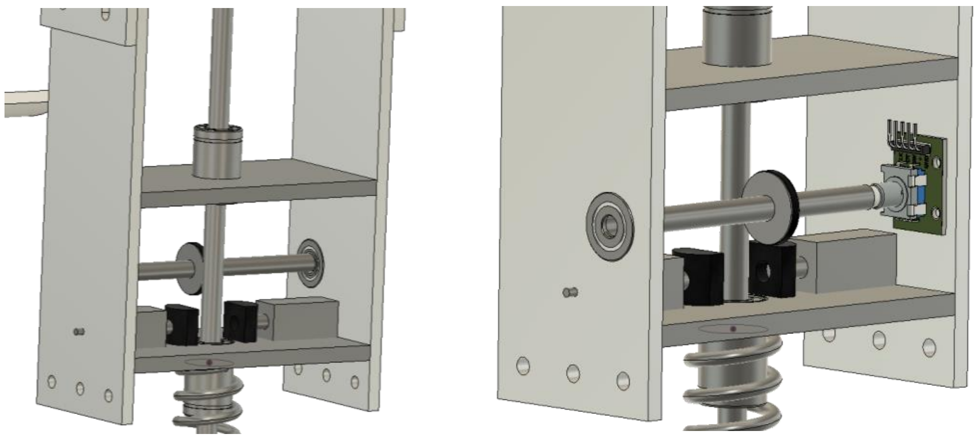


Fig-3.8 Locking Mechanism

In locking mechanism, the concept of cycle braking is used, as the brake’s handle is pulled, the rubber brakes come closer to each other and the rotating wheel is braked against friction.The rod (of 10mm dia), frames and the base (of 5mm thickness) on which solenoid lock is resting are of delrin material. Delrin is a crystalline plastic bridges the gap between metals and plastics by offering an excellent balance of properties. It possesses high tensile strength, creep resistance and toughness.

As the robot takes its first step, the foot hits the ground and due to its impact the rod slides through the linear bearing. The rod moves in the upward direction and the wheel (mounted on a 7mm dia rod) attached to it moves in counter clockwise direction. Due to this the encoder also moves in the same direction as the wheel does. When the delrin rod moves in the downward direction, at that instant, the solenoid lock enables and applies force against the moving rod to stop it, thus, locking the rod and the overall foot of the robot.

2.3 HIP

The hip of the robot is where both the legs are joining together using a revolute joint. The revolute joint is controlled by a brushed DC motor, which is mounted on a leg and drives the revolute joint using a rubber belt.

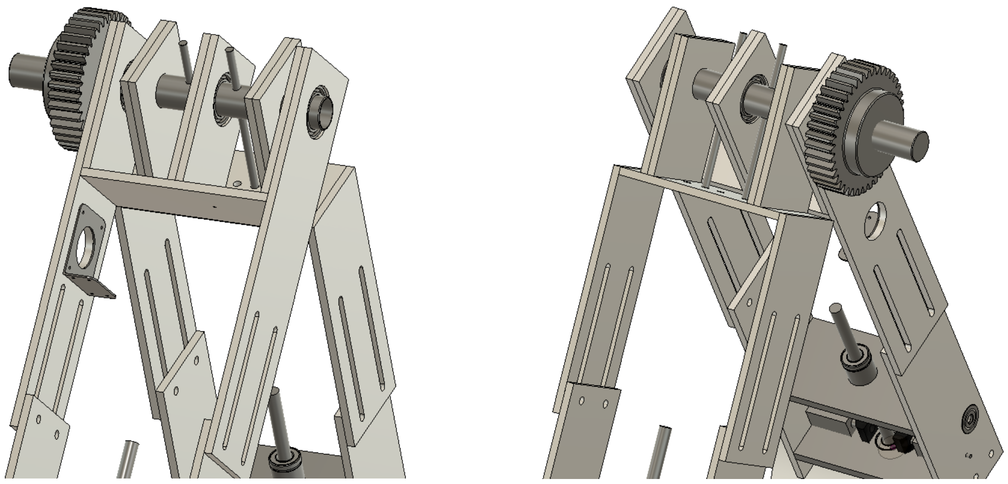


Fig-3.9 Design of Hip

The motor is mounted on one of the leg as shown in the above image at L-clamp. The hip features height adjustment, in which the robot’s height can be adjusted according the required needs upto 8cm. The maximum height of the robot is 50cm. The height is adjusted using a nut bolt coupling which is screwed as the required height using a ruler which is pasted on the leg.

The locking rods are used to lock the shaft with the other leg on which the motor is not mounted. So that when the motor actuates the shaft, the other leg comes into motion. The entire frame is of delrin material, the shaft is of steel rod.

3. DC Motor Control

The position of the DC servo motor is controlled by a STEP/PULSE and DIRECTION digital interface similar to stepper motors. By regulating the amount of voltage supplied to DC motor, the speed of the motor is controlled and this is the reason PWM is used for DC motor speed control.

Fig-4.0: RMCS-2301 Motor Controller

Fig-4.1: RMCS-3017 Brushed DC Motor

Below shown is the circuit diagram of the RMCS motor which is controlled by RMCS controller.

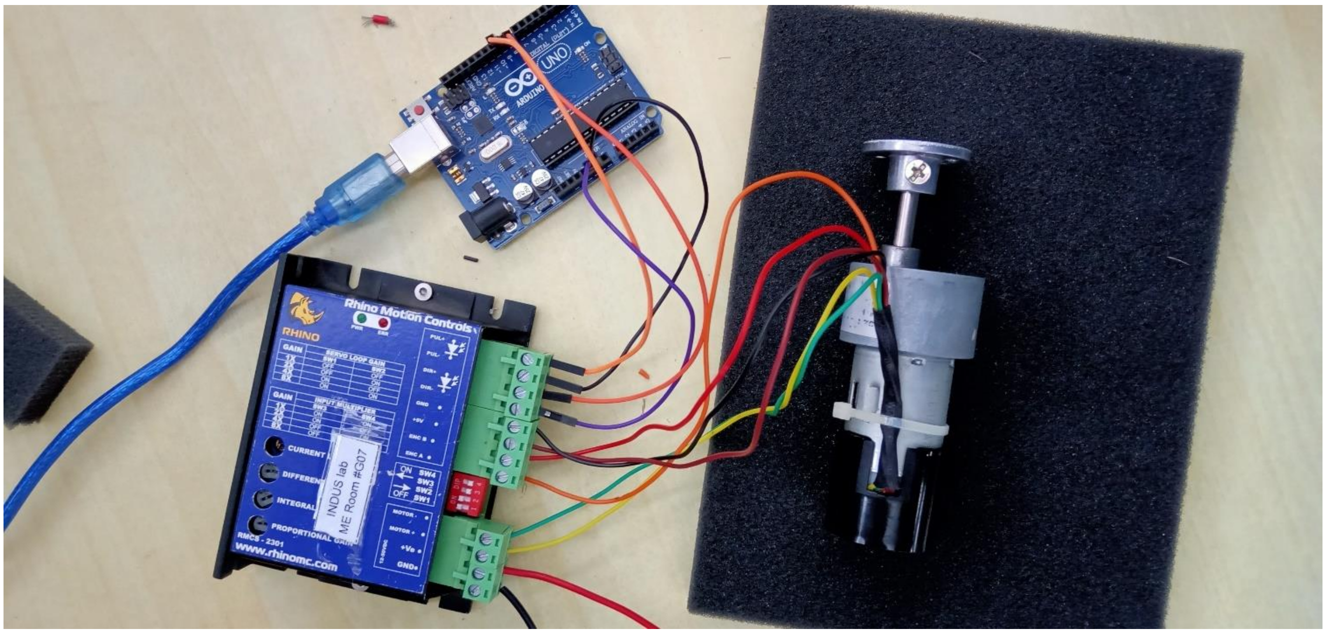


Fig-4.2 Circuit Diagram for Motor Control

The issue faced while controlling the motor is getting encoder feedback using Arduino board. The motor to encoder gear ratio is 1:20, so to read encoder pulses on 3280 CPR the arduino’s sampling frequency is incapable of reading data. To overcome this issue, the microcontroller Freescale 9S12XEP100 is used.

Freescale 9S12XEP100 is an academic microcontroller. In this controller, the sampling frequency needs to be set to get encoder feedback, using this, the motor is controlled for setpoint tracking.

The program is written in Annexure 2.1 for the control of DC Motor.

<u>Chapter-4</u>
RESULT

The robot is designed successfully by considering all the environmental parameters, and has additional features in it which makes its testing simple for practical implementation. The design is robust; its height is 50cm when standing on both the legs and 90cm when standing upright on a single leg. The robot has the height adjustment feature of up to 8cm which make the robot's height vary from 42cm to 50cm. It has portability option, which means it can be easily dismantled and again assembled by nuts and bolts.

Controller for double inverted pendulum is made and simulated on MATLAB. The simulation is done in step by step procedure, which was learned from the basics and implemented. The PID controller is learned and simulated for the double inverted pendulum.

To control the under-actuated double inverted pendulum for balancing ballerina's skill, linearization method is applied to linearize the equation of motion of the 2-link manipulator. Using this technique, the gain parameters were derived and substituted in PID control of under-actuated double inverted pendulum where the controlling of first link is done. This resembles the controlling of reaction wheel pendulum which is another step that needs to be done while balancing the robot.

Chapter-5
OUTCOME & FUTURE SCOPE

The outcome of the project after 6 months of research is that the design has been successfully created and the locking mechanism is fabricated and assembled. Dynamics and Controlling of the robot is understood and the step by step procedure that needs to be taken when developing a robot is known. Implemented the dynamics of the robot on MATLAB and simulated a reaction wheel pendulum. The design of the compass bipedal robot is used further in research to perform handspring maneuver.

The robot is made for scientific interest in robot gymnasium but has the future scope to perform flipping skill, which can be used in military purpose as well as in farming purpose. In military purpose, it can be used as a decoy to send explosives to enemy’s land during war. In farming, it can be a robot farmer who can monitor the farm by flipping within the farm. The flipping mechanism has advantage in walking over soil as compared to normal bipedal walking robots.

REFERENCES

Journal / Conference Papers

[1] Víctor De-León-Gómez, Víctor Santibañez, Javier Moreno-Valenzuela. A procedure to find equivalences among dynamic models of planar biped robots. Simulation Modelling Practice and Theory, Elsevier, 2017. <hal-01728357>

[2] Torleif Anstensrud, “2-D Passive Compass Biped Walker”, Norwegian University of Science and Technology, Country, August, 2013.

Reference / Hand Books

[1] M. Spong and Vidhyasagar, “Robot Dynamics and Control”, Wiley Co., 3rd Edition

[2] K. Ogata, “Modern Control Engineering”, Prentice Hall, 3rd Edition

Web

[1] Ode45, OnRamps, MATLAB

[2] Robokits

ANNEXURES

1. MATLAB CODES

1.1 VERIFICATION OF EQUATION OF MOTION

1.1.1 Main Program

```
tspan=0:1:100; %total time

M1=1; M2=1; a=1; L=1; g=9.8;

q1=deg2rad(5); %Joint-angle 1
dq1=deg2rad(0);
q2=deg2rad(-5); %Joint-angle 2
dq2=deg2rad(0);

Ei=(0.5*(M1+M2).*(a.^2).*(dq1.^2))+(0.5*M2*((L^2)+(2*a*L*cos(q2))).*(dq1.^2))+(M2*
((a^2)+(a*L*cos(q2))).*dq1.*dq2)+(0.5*M2*(a^2).*(dq2.^2))+(M1*g*a.*sin(q1))+(M2*g*
((L.*sin(q1))+(a.*sin(q1+q2))));

y0=[q1 q2 dq1 dq2]; %Initial states

opts=odeset('RelTol',10e-9,'AbsTol',10e-10);
[t,y]=ode45(@Energy,tspan,y0,opts);

q1=y(:,1);
q2=y(:,2);
dq1=y(:,3);
dq2=y(:,4);

K=(0.5*(M1+M2).*(a.^2).*(dq1.^2))+(0.5*M2*((L^2)+(2*a*L*cos(q2))).*(dq1.^2))+(M2*(
(a^2)+(a*L*cos(q2))).*dq1.*dq2)+(0.5*M2*(a^2).*(dq2.^2)); %Kinetic Energy
P=(M1*g*a.*sin(q1))+(M2*g*((L.*sin(q1))+(a.*sin(q1+q2)))); %Potential Energy

E=K+P; %Total Energy
E_error=zeros(length(t),1);

for i=1:length(t)
    E_error(i)=abs(E(i)-Ei);
end

Power=(m1*(a^2)*dq1*ddq1)+(m2*((a^2)*dq1*ddq1)+((L^2)*dq2*ddq2)+(a*L(cos(q1-
q2)*((dq1*ddq2)+(ddq1*dq2))+(dq1*dq2*sin(q1-q2)*(dq1-dq2)))))-
(m1*g*a*sin(q1)*dq1)-m2*g*(a*sin(q1)+L*sin(q2)*dq2); %Power

figure(1)
plot(t,E)
hold on
plot(t,E_error)
hold on
legend('E','E_error')
title('Energy of Eq^n of Motion');
xlabel('Time Steps');
ylabel('Energy (J)');
```

```
figure(2)
plot(t,Power)
hold on
```

1.1.2 Ode Function Energy

```
function dY=Energy(t,y)

M1=1; M2=1; a=1; L=1; g=9.8;

q1=y(1);
q2=y(2);
dq1=y(3);
dq2=y(4);

dq=[dq1; dq2];

m11=((M1+M2)*(a^2))+(M2*(L^2))+(2*M2*a*L*cos(q2));
m12=(M2*(a^2))+(M2*a*L*cos(q2));
m21=(M2*(a^2))+(M2*a*L*cos(q2));
m22=M2*(a^2);

c11=-M2*a*L*sin(q2)*((2*dq1*dq2)+(dq2^2));
c21=M2*a*L*sin(q2)*dq1*dq1;

g11=(((M1*a)+(M2*L))*g*cos(q1))+(M2*g*a*cos(q1+q2));
g21=M2*g*a*cos(q1+q2);

ddq1=(((-(c11+g11)).*m22)-((-(c21+g21)).*m12))./((m11.*m22)-(m12.*m21));
ddq2=(((-(c11+g11)).*m21)-((-(c21+g21)).*m11))./((m12.*m21)-(m11.*m22));

ddq=[ddq1; ddq2];

dY=[dq; ddq];
```

1.1.3 Acceleration Function

```
function ddq=ddq_solver(q1,q2,dq1,dq2,t1,t2)

M1=1; M2=1; a=0.5; L=1; g=9.81;

m11=((M1+M2)*(a^2))+(M2*(L^2))+(2*M2*a*L.*cos(q2));
m12=(M2*(a^2))+(M2*a*L.*cos(q2));
m21=(M2*(a^2))+(M2*a*L.*cos(q2));
m22=M2*(a^2);

c11=-M2*a*L.*sin(q2).*((2.*dq1.*dq2)+(dq2.^2));
c21=M2*a*L.*sin(q2).*dq1.*dq1;

g11=(((M1*a)+(M2*L))*g.*cos(q1))+(M2*g*a.*cos(q1+q2));
g21=M2*g*a.*cos(q1+q2);

ddq1=(((t1-(c11+g11)).*m22)-((t2-(c21+g21)).*m12))./((m11.*m22)-(m12.*m21));
ddq2=(((t1-(c11+g11)).*m21)-((t2-(c21+g21)).*m11))./((m12.*m21)-(m11.*m22));
ddq=[ddq1 ddq2];
end
```

1.2 PID CONTROL

1.2.1 Main Program

```
clear all;

tspan=0:0.1:10;

M1=1; M2=1; L1=1; L2=1; g=9.8;

Kp1=20; Kp2=20;
Kd1=15; Kd2=17;

PID=[Kp1 Kd1 Kp2 Kd2];

q1=deg2rad(5); %Joint-angle 1
dq1=deg2rad(0);
q2=deg2rad(-5); %Joint-angle 2
dq2=deg2rad(0);

y0=[q1 q2 dq1 dq2]; %Init states

opts=odeset('RelTol',10e-9,'AbsTol',10e-10);
[t,y]=ode45(@(t,y) New_PID(t,y,PID),tspan,y0,opts);

q1=y(:,1);
q2=y(:,2);
dq1=y(:,3);
dq2=y(:,4);

figure(1) %Plot%
plot(t,rad2deg(q1))
hold on
legend('q1');
title('PID Control');
xlabel('time(steps)');
ylabel('Angle(deg)');

Robot_Movie(q1,q2)
```

1.2.2 Robot Movie(Animation)

```
function Anmt=Robot_Movie(q1,q2)

tsol=0:0.1:100;
L1=1; L2=1;

j1=q1;
j2=q2;

x1=L1.*sin(j1);
y1=L1.*cos(j1);
x2=(L2.*cos(deg2rad(90)-j1+j2))+(L1.*sin(j1));
y2=(L2.*sin(deg2rad(90)-j1+j2))+(L1.*cos(j1));

i=1; j=1:i:length(tsol);    %% generating images in 2D
```

```
figure(10)
 for i=1:length(j)-1
    hold off
    plot([x1(j(i)) x2(j(i))],[y1(j(i)) y2(j(i))],'o',[0 x1(j(i))],[0 
y1(j(i))],'k',[x1(j(i)) x2(j(i))],[y1(j(i)) y2(j(i))],'k')
    xlabel('x')
    ylabel('y')
    axis([-2 2 -2 2]);
    grid
    hold on
    MM(i)=getframe(gcf);
 end
drawnow; %% exporting to
Anmt=mpgwrite(MM,'RGB','trajectory.mpg');
```

1.2.3 Ode Function

```
function dY=New_PID(t,y,PID)

M1=1; M2=1; L1=1; L2=1; g=9.8;

Kp1=PID(1); Kp2=PID(3);
Kd1=PID(2); Kd2=PID(4);

q1=y(1);
q2=y(2);
dq1=y(3);
dq2=y(4);
q1_des=deg2rad(0);
q2_des=deg2rad(0);

u1=((Kp1*(q1_des-q1))+(Kd1*(-dq1)));
u2=((Kp2*(q2_des-q2))+(Kd2*(-dq2)));
U=[u1; u2];

dq=[dq1; dq2];

m11=((M1+M2)*(L1^2))+(M2*((L2^2)+(2*L1*L2*cos(q2))));
m12=-M2*((L1^2)+(L1*L2*cos(q2)));
m21=m12;
m22=M2*(L1^2);

M=[m11 m12; m21 m22];

c11=M2*L1*L2*sin(q2)*(-2*dq1+dq2)*dq2;
c21=M2*L1*L2*sin(q2)*(dq1^2);

C=[c11; c21];

g11=-(((M1*L1)+(M2*L2))*g*sin(q1))-(M2*g*L1*cos(deg2rad(90)-q1+q2));
g21=M2*g*L1*cos(deg2rad(90)-q1+q2);
G=[g11; g21];

T=M*U;
t1=T(1); t2=T(2);

ddq1=(((t1-(c11+g11))*m22)-((t2-(c21+g21))*m12))/((m11*m22)-(m12*m21));
ddq2=(-((t1-(c11+g11))*m21)+((t2-(c21+g21))*m11))/(-(m12*m21)+(m11*m22));
ddq=[ddq1; ddq2];
dY=[dq; ddq];
```

1.3 LINEARIZATION

```
function K=controller(A,B)
M=[B A*B (A^2)*B (A^3)*B];
J=[-10 0 0 0; 0 -10 0 0; 0 0 -10 0; 0 0 0 -10];
Phi=polyvalm(poly(J),A);
K=[0 0 0 1]*(inv(M))*Phi;
```

The K matrix is substituted in the PID program in appendix 1.2.1.

2. MOTOR (RMCS-3017)

2.1 SPECIFICATIONS

RMCS-3017 is a high torque quad encoder geared DC motor.

Table-1: Technical specification of Motor

Operating Voltage	12V DC
Motor Speed at Output Shaft (RPM)	900RPM
Stall Torque (Kg-cm)	3.02 Kg-cm
Rated Torque (Kg-cm)	1.5 Kg-cm
Gear Ratio	1:20
CPR at Output Shaft of Motor	3280
Weight	180gm

Table-2: Encoder Pinouts

Encoder Type	Quad Encoder
Line Encoder	41 Line Encoder
Base Motor CPR	41*4=164
Orange	EncA
Red	EncB
Yellow	Motor+
Green	Motor-
Brown	+5V DC
Black	Gnd

2.2 CONTROL PROGRAM

The motor is controlled using 9S12XEP100 microcontroller and its code is as

```
#include <hidef.h>      /* common defines and macros */
#include "derivative.h"      /* derivative-specific definitions */

float kp = 0.01;
float ki = 0.001;
float p = 0;
long int sum = 0;
int     i=1,j=1;
long    count=0;
int     En_Read=0x0000;
int     des = 0;
int     L_DATA=0x00,U_DATA=0x00;
void    PWM_init(void);
void    PWM_for_Motor(void);

void interrupt 66 intrr_func(void){
 PITTF_PTF0 = 1;
 PORTB_PB2 = 0;
 PORTB_PB3 = 0;  //Enable=0,SelectPin=0       {SEL=PB2,OE=PB3}
 U_DATA=PORTA;
 PORTB_PB2 = 1;         //Enable=0,SelectPin=1       {SEL=PB2,OE=PB3}
 L_DATA=PORTA;
 En_Read=(U_DATA<<8)|L_DATA; //Combining lower byte and Upper Byte into a 16-bit
binary number
 PORTB_PB3 = 1; //Inhibit logic reset Enable Pin=1, Select Pin = X
        count++;                //Reference value update
        if(kp*(-En_Read+des) + ki*sum>=0) {
          if(kp*(-En_Read+des) + ki*sum<250){
            sum += (-En_Read+des);
            p = kp*(-En_Read+des) + ki*sum;
            PWMDTY5=p;
          }
          else{
            PWMDTY5=250;
          }
          PORTB_PB6 = 0;
          PORTB_PB7 = 1;
        }
        else{
            if((kp*(-En_Read+des) + ki*sum)*(-1)<250){
              sum += (-En_Read+des);
              p = kp*(-En_Read+des) + ki*sum;
              PWMDTY5=(-1)*p;
            }
            else{
              PWMDTY5=250;
            }
            PORTB_PB6 = 1;
            PORTB_PB7 = 0;
        }
}

void main(void) {

        DDRA=0x00;
        DDRB=0xFF;
        PWM_for_Motor();
        PWM_init();
        PITCFLMT_PITE = 1;
        PITCE_PCE0 = 1;
        PITINTE_PINTE0 = 1;
        PITMUX_PMUX0 = 0;
        PITMTLD0 = 199;
        PITLD0 = 999;

EnableInterrupts;


for(;;) {
    _FEED_COP(); /* feeds the dog */
  } /* loop forever */
  /* please make sure that you never leave main */
}


void    PWM_for_Motor(){
     PWME_PWME5=1;
     PWMCLK_PCLK5 = 0;
     PWMPRCLK_PCKA0 = 1;
     PWMPRCLK_PCKA1 = 1;
     PWMPRCLK_PCKA2 = 0;
     PWMPOL_PPOL5 = 1;
     PWMDTY5 = 5;
     PWMPER5 = 250;
     PORTB_PB6 = 0;
     PORTB_PB7 = 1;
}

void    PWM_init()     {       //sampling rate for encoder
        PWMCLK_PCLK2=0;
        PWME_PWME2=1;
        PWMPOL_PPOL2=1;
        PWMPRCLK=0x00;
        PWMDTY2=40;
        PWMPER2=50;
}
```