Overhead Ball Throwing for Humanoid Robot Soccer

by

QAZI AAFAQUE MASOOD

00000118053

Supervisor

DR. YASAR AYAZ

Robotics and Intelligent Machines

SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

September, 2019

Overhead Ball Throwing for Humanoid Robot Soccer

by

QAZI AAFAQUE MASOOD

00000118053

A thesis submitted in partial fulfillment of the requirements for the degree of

MS ROBOTIC AND INTELLIGENT MACHINES ENGINEERING

Thesis Supervisor:

DR. YASAR AYAZ

Thesis Supervisor's Signature: _____

Robotics and Intelligent Machines Engineering

SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY,

ISLAMABAD

SEPTEMBER, 2019

Declaration

I certify that this research work titled "*Overhead Ball Throwing for Humanoid Robot Soccer"* is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources it has been properly acknowledged / referred.

QAZI AAFAQUE MASOOD
2015-NUST-MS-RIME-00000118053

Plagiarism Certificate

It is certified that MS Thesis Titled **"Overhead Ball Throwing for Humanoid Robot Soccer"** by **"Qazi Aafaque Masood"** has been examined by us. We undertake the follows:

a. Thesis has significant new work/knowledge as compared already published or are under consideration to be published elsewhere. No sentence, equation, diagram, table, paragraph or section has been copied verbatim from previous work unless it is placed under quotation marks and duly referenced.
b. The work presented is original and own work of the author (i.e. there is no plagiarism). No ideas, processes, results or words of others have been presented as Author own work.
c. There is no fabrication of data or results which have been compiled/analyzed.
d. There is no falsification by manipulating research materials, equipment or processes, or changing or omitting data or results such that the research is not accurately represented in the research record.
e. The thesis has been checked using TURNITIN (copy of originality report attached) and found within limits as per HEC plagiarism Policy and instructions issued from time to time.

Name & Signature of Supervisor
Dr. Yasar Ayaz

Signature **:** _____

Copyright Statement

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS thesis written by Mr. Qazi Aafaque Masood Registration No. 118053 of **__SMME___** has been vetted by undersigned, found complete in all aspects as per NUST Statutes/Regulations Policy, is free of plagiarism, errors, and mistakes and is accepted as partial fulfillment for award of MS degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in the said thesis.

Signature with stamp: _____

Name of Supervisor: Dr. Yasar Ayaz

Date: _____

Signature of HoD with stamp: _____

Date: _____

Countersign by

Signature (Dean/Principal): _____

Date: _____

FORM TH-4

MASTER THESIS WORK

We hereby recommend that the dissertation prepared under our supervision by: (Student Name & Regn No.) QAZI AAFAQUE MASOOD, 118053

Titled: Overhead Ball Throwing for Humanoid Robot Soccer be accepted in partial fulfillment of the requirements for the award of MS degree.

Examination Committee Members

1. Name: AP Sara Baber Signature:_____

2. Name: Dr. Hassan Sajid Signature:_____

3. Name: Muhammad Hamza Asif Signature:_____

   (Co-supervisor)

Supervisor's name: Dr. Yasir Ayaz Signature:_____

Date:_____

_____ _____

Head of Department Date

COUNTERSINGED

_____

Dean/Principal

Date:_____

Acknowledgements

I would like to thank many individuals who helped me with finishing my thesis. Top of all. I want to Thank Allah for provision of knowledge, health and the strength to finish the research.

I would like to express my gratitude to my friends, especially Mr. Saifullah for helping with this project. I would like to thank Mr. Hamza who helped me during the course of my masters. I also would like to thank my company, EZETEK, for providing me with all the resources I needed to work on my thesis.

Most of all, I would like to Thank Dr. Yasar Ayaz for encouraging me and keeping faith in me even in the darkest of times. I would like to thank him for all the supervision he provided throughout my time here.

My thanks and appreciation go to all people who helped me one way or another with my work.

Dedicated to my exceptional parents and Mr. Akhtar Nawaz. He may be away now but his spirit stays with me forever

Abstract

This research is a handout to the RoboCup, a football competition for robots that aims to create more intelligent machines. Our task is the design and implementation of legal throw-in from sidelines. Objective is to formulate and test a technique that can be used for the above mentioned task. In football, legal throw-ins are carried out by using both hands, while keeping both the feet on the ground throughout the process. We have designed and implemented this technique on NAO. NAO has three fingers in each hand and to grasp the ball properly throughout the process is a daunting task. Grasp force was important since it is important to keep the ball in hands while lifting the ball behind the head and following the generated trajectory. For this purpose, we initially designed Static Grasping Model and Dynamic Grasping Model. After the calculation of sufficient force, the trajectory generation was planned and executed. We generated circular and elliptical trajectories and tested both separately. After the calculation of the maximum velocity on both the above mentioned trajectories, we planned the release points. At the time of release, we varied the throw-in force equip our robot to choose the destination player accordingly. We designed different methods to maximize the stability. Final results were closer to the expectations. Future works include the use of computer vision algorithms to variate the destination points.

Index Terms—*RoboCup, NAO, Static Grasping Model, Dynamic Grasping model, Trajectory Generation*

Table of Contents

Declaration ..... **i**
Plagiarism Certificate ..... **iii**
Copyright Statement ..... **iii**
Acceptance Certificate ..... **iv**
TH-4 ..... **v**
Acknowledgements ..... **vi**
Dedication ..... **vi**
Abstract ..... **viii**
Table of Contents ..... **ix**
List of Figures ..... **x**
List of Tables ..... **xi**
CHAPTER 1: INTRODUCTION AND LITERTURE REVIEW **1**
CHAPTER 2: NUMERIC SPECIFICATIONS OF EQUIPMENT **5**
2.1 Dimensions of ball **5**
2.2 Relevant dimensions of NAO and joint Specifications 6
CHAPTER 3: PROBLEM FORMULATION **7**
CHAPTER 4: GRASPING MODELS **10**
CHAPTER 5: EXPERIMENTAL RESULTS **17**
REFERENCES **28**

List of Figures

Figure 1: Simulation of our legal throw-in from sidelines 4
Figure 2: LEFT- Red ball that was previously used, dia. 6.5cm. RED - ball used in the project, dia. 10.0 cm.. **Error! Bookmark not defined.**
Figure 3: Front and Top View of NAO and Dimensions in millimeters 6
Figure 4: Joints of Left hand: LShoulderPitch, LShoulderRoll, LElbowYaw, LElbowRoll and LWristYaw.....7
Figure 5: Joints of Right hand: RShoulderPitch, RShoulderRoll, RElbowYaw, RElbowRoll and RWristYaw 8
Figure 6: Force Resistor Circuit – Patched, with one FSR 10
Figure 7: Force Resistor Circuit – Schematic. With PWM and without PWM signal.....11
Figure 8: Static Grasping Model – Posture.....12
Figure 9: Dynamic Grasping Model – Posture 13
Figure 10: Static Grasping Model – Posture - Graph 14
Figure 11: Static Grasping Model – Posture - Graph 14
Figure 12: Release Point.....18
Figure 13: Initial Point – Co-ordinates of joints 23
Figure 14: Initial Point – Euler Angles, Orientation.....24
Figure 15: Initial Point – Transformation Matrix 24
Figure 16: Final Point – Co-ordinates of joints 25
Figure 17: Final Point – Euler Angles, Orientation 25
Figure 18: Final Point – Transformation Matrix 26

List of Tables

Table 1: Left Arm Joint nomenclature, joint axis revolution and ranges.....8
Table 2: Right Arm Joint nomenclature, joint axis revolution and ranges.....9

CHAPTER 1: INTRODUCTION AND LITERTURE REVIEW

The people from generation Y and before that, can recall the memories of their childhood when they had to wait for their friends to play any sport. The injection of RC cars was nothing less than a pleasant shock. We finally could play something without a friend. With time, even this started to get boring, because the cars were not learning and the only thing they could do, was move and turn on our commands. We definitely wanted it to do more, may be do things on its own. Less we know, Artificial Intelligence was on the rise at that time, we just didn't get our hands on it then.

1.1. Background

The real initial success of artificial intelligence is considered to be the time when a Robot (IBM Deep Blue) defeated Garry Kasparov, the world champion of chess [1] at that time. Its success can be dedicated to its vast database of moves, yet the effective use its database was all that mattered really. It had complex estimation functions and it emphasis and focus was majorly on search extensions.

After that, a series of headlines boosted the advancements of AI, whether it was the deployment NASA pathfinder, with its primary objective to create a single dataset from a big AVHRR global area coverage dataset [2] or other successful missions, AI succeeded.

Further advancements in AI were inevitable and the results can be seen today. However, it was decided that the best possible way to get the rapid results and elevate the outcomes of this field is to make it competitive enough in the field of sports against human opposition. It all started in 1950 when Turing published "Computing Machinery and Intelligence"[3]. Its end or final accomplishment cannot be determined, may be Artificial Intelligence has no end. It will continue to grow.

All the football fans (or should we say soccer for US readers) love to play the game as well. Even though, football is a tough game to play, but being human beings, we learn it easily, at least the basic parts of it. The initiative of RoboCup is one link to that long chain of making robots intelligent enough to compete against human opposition. Standard Platform League (SPL) has NAO as players where environment is different from chess, it is dynamic, real time and the control

.

is distributed. The first Robocup competition took place in Nagoya and at that time, the target was to create agents that can collaborate well in a multi-agent environment with various strategy derivations [4].

The challenge we took on and try to accomplish was of throwing the ball using both the hands. In football, when the ball goes out of play by an opposition player's touch, one of the team player from other team has to start the play by throwing the ball back in field. In order for the throw to be legal, there are certain rules. Player needs to have both the feet on the ground at the time of release. Throw has to be over the head, cannot be side arm and finally throw has to be carried out from outside the rectangular boundaries of the playing area.

Over the period of time, many researchers have attempted the throw and used it in different applications. There hasn't been one similar to our application throw-in attempted in the past and we came up with different ways to attempt this using various techniques. Han Lin [5] devised a method of picking up the ball and throwing it in to a box. In his research, machine vision was used to find the ball at first. After detecting the ball, one hand ball pick-up was used. We cannot use this method as it requires one hand pick up and the throw is not actually a throw but it is a drop instead.

Mittal, Bhavishya and Pratibha [6] came up with used NAO to identify the target in the cluster and throw the ball at that target. Identification was done by the built-in modules and after detecting if the target is in the defined radius, robot performed the throw, otherwise it communicated its inability to do so. The throw was under-arm and one handed and can have application in military. The techniques the used cannot be used by us since we require a two handed throw and it has to be from top of the head.

Teppei Tsujita, Atsushi Konno and Yasar Ayaz [7] generated motion for HRP-2 for the nail driving application. They devised a control system to exert impulsive force multiple times to drive the nail successfully in the wall. Feed Control system kept the robot stable even when large force was exerted. Since, we need to throw the ball using range of force from less to excessive depending on the distance of player we are aiming at, we require similar techniques to keep our robot stable.

This project replicates this technique to carry out a legal throw-in. It was indeed a challenging task as NAO has 3 fingers on each hand and its fingers are not sophisticated in control. Hand can either open up or close down. There is not independent control of each finger, instead

.

all three fingers are opened and closed with a single command. Due to this, it was important for us to devise a strategy to calculate an optimal force at which the ball is gripped properly. To test an optimal force, we carried out gripping tests using Force Sensor Resistors (FSR). We derived static gripping model and dynamic gripping model.

After the completion of Static and Dynamic Gripping Models, and the calculation of an optimal gripping force, next task was to generate the trajectory for throw-in. Trajectory generation was divided in 2 steps. Generation for "Lifting" and generation for "Throwing". The initial point "*T_sl*" of trajectory generation for "Lifting" was the point where we handed the ball to NAO and the final point "*T_fl*" is above the head. In the trajectory generation for "Throwing", the final point "*T_fl*" becomes the initial point "*T_st*" and the final point "*T_ft*" is the release point which is varied constantly depending upon the target of the throw. Changing the angle and force changes the impact point and increases the flexibility of our contribution to the target application.

Erkorkmaz, Kaan and Yusuf [8] explained the idea of quintic spline that limits the jerk in the motion. The applied this trajectory generation method on CNC machine manipulator and produced continuous positions, velocities and accelerations. Parameterization errors were removed by their algorithm, hence feed rate swings were eliminated.

Mihai Dupac [9] used cubic polynomials to solve the trajectory generation problems of the end effector of a rotating manipulator to generate smooth trajectories. Geometric paths were calculated using the polar curves and the results were verified using the two different configurations to generate the velocity and trajectory profiles.

Kucuk [10] derived an algorithm to create optimal trajectories. His algorithm "Optimal Trajectory Generation Algorithm" (OTGA) is implemented for series and parallel manipulators. 7th order cubic splines were used to generate smooth motions by removing the ripples in velocities and acceleration curves. Particle Swarm optimization was used to optimize the algorithm because of it easy use. It consumes less power and produces 2.5% less jerks.

We used cubic splines interpolation to generate the trajectories. We added multiple via points while generating trajectory for lifting. It kept the hands in a smooth path. Same method was used by Teppei Tsujita and Yasar Ayaz [7] in their nail driven task for HRP-2. However, they generated the trajectories for one arm. We designed the trajectory for one arm and mirrored it onto the next one. The reason for that is the identical but mirror motion for both the arms. The advantage of using quintic or cubic spline interpolation is the reason that it reduces some of the factors to

.

zero and makes the computation a little easier. We generated the trajectory *q_sl*, *q_fl*, *q_st* and *q_ft* in time *T_l* and *T_t* respectively.

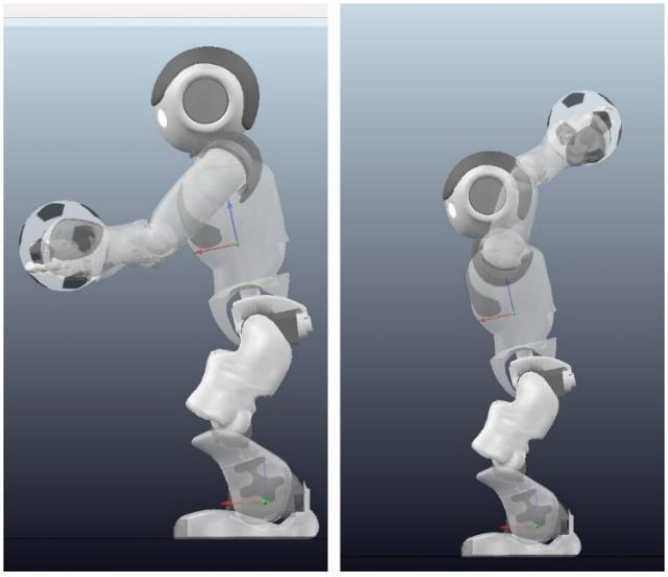

Figure 1 Simulation of our legal throw-in from sidelines

After implementing the trajectories, the next challenge was to determine the release point of the ball. The release angle determines the point of impact on the ground just like the force. We decided to use circular and elliptical trajectories for "Throwing Part". We kept a constant angle and moved the ball on the circumference of the circle with radius "r". The radius in fact is the third side of the triangle that is formed by the upper arm and the lower arm. The change in the angle of elbow defines the radius of the circle. We attempted throw-ins using variable force. We used different trajectories to change the point of impact. Release angle also changes the point of impact and creates that all needed versatility that this specific application demands. For future, we can use machine vision techniques to detect our team mates and using different decision making strategies to choose the receiving player among different available options.

.

CHAPTER 2: NUMERIC SPECIFICATIONS OF EQUIPMENT

2.1 Dimensions of ball

The specifications of ball being used is of great importance. Its size and color alters the whole grasping and playing strategies. Before 2017, Red ball was used in RoboCup. It was relatively easy because using detection techniques of machine vision, getting the information of the ball was not a daunting task. That ball was smaller in size than the one used now. On one end grasping the red ball was easier because of the smaller hands of NAO and on the other end, the ball to robot ratio in size was not realistic to what we have in human football. Its diameter was 6.5cm. One way of detecting the ball was done by using color based surface detection.

However, now the ball that is being used is more realistic in features. It has black and white patches. Detection has become difficult because more sophisticated techniques are required, but then again, this ball is more practical and challenges the involved researchers. One way of perceiving the ball is by using classifier based detection techniques. However, this ball is more suitable to our project because we are using two hands for grasping. Its bigger size makes this task fairly practical. The illustration of comparison is shown in figure 2.1.

Figure 2 - On the left, red ball that was previously used, dia. 6.5cm. On the right, ball used in the project, dia. 10.0 cm

.

2.2 Relevant dimensions of NAO and joint Specifications

Joint is the connection between two links. The NAO we used has 24 Degrees of freedom. It has open and close motion in both the hands as well. We utilized the motion in hip and knee joint as well as the motion in the arms.

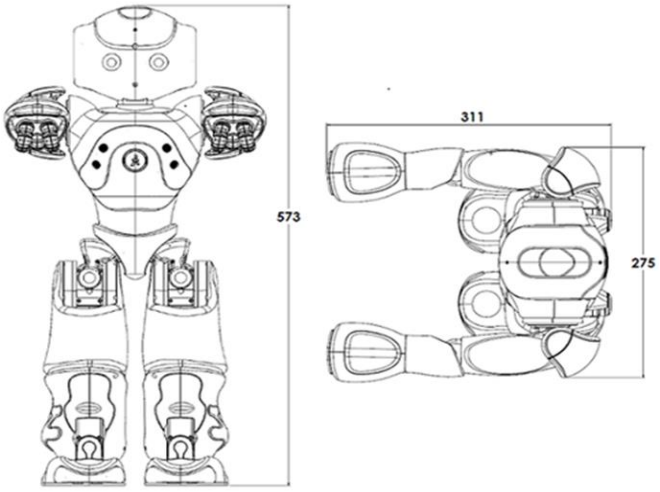

Figure 3 Front and Top View of NAO and Dimensions in millimeters

Height of NAO robot is 573 mm, depth is 311 mm and width is 275 mm.

.

CHAPTER 3: PROBLEM FORMULATION

As mentioned before, NAO robot has 5 joints in its each arm. Problem we are tackling here is of Throw-in from sidelines. While discussing throws in football, it is important to remember that outfield players are only allowed to touch the ball with their hands is when ball goes out of play by opposition's last touch. Goal keeper however can throw the ball in any way he wants, that is whole other research area.

Our problem is defined within the regularities of FIFA for legal throw-ins. Humans, or in our case robot needs to have both feet on ground, however, not necessarily aligned. Throw cannot be under arm and has to be from above the head. Release should take place from both the hands at the same time.

We have certain constraints in our case. Fingers of NAO are not sophisticated and cannot be controlled independently. We are using all the 10 DOF from both the arms. The ranges and joint specifications of both hands are as follows:

Left Hand: Left Shoulder Pitch, Left Shoulder Roll, Left Elbow Yaw, Left Elbow Roll, Left Wrist Yaw.

Right Hand: Right Shoulder Pitch, Right Shoulder Roll, Right Elbow Yaw, Right Elbow Roll, Right Wrist Yaw.

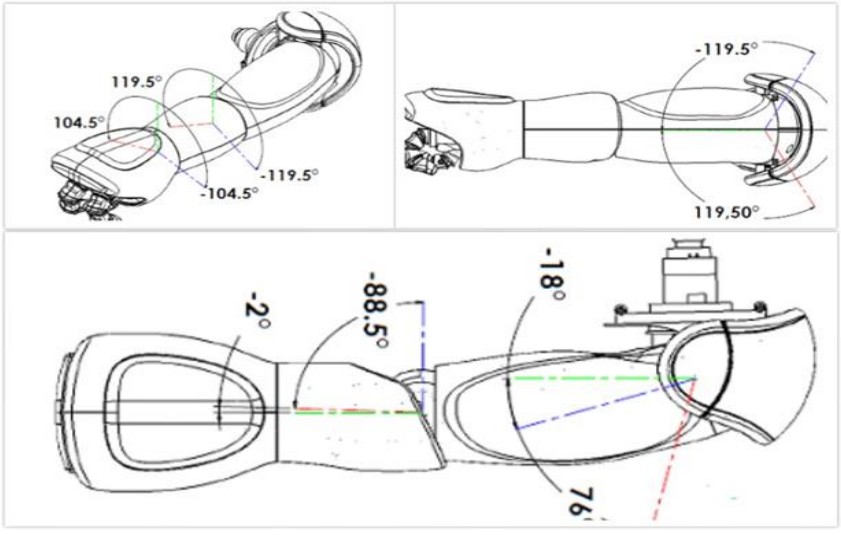

Figure 4 – Joints specifications of Left hand: LShoulderPitch, LShoulderRoll, LElbowYaw, LElbowRoll and LWristYaw.

.

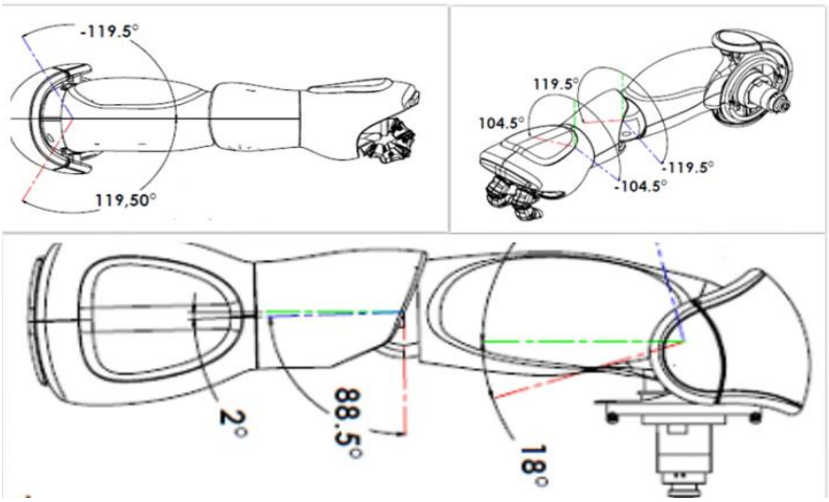

Figure 5 – Joints specifications of Right hand: RShoulderPitch, RShoulderRoll, RElbowYaw, RElbowRoll and RWristYaw.

We are starting our robot from standby positions where its joint values and frame positions are at default positions. Default joint values at standby position for Left Shoulder Pitch, Left Shoulder Roll, Left Shoulder Yaw, Left Shoulder Roll and Left Wrist Yaw are 90.0, 11.5.0, -90.0, 0.0 and -90.0 respectively. Similarly joint values for Right Shoulder Pitch, Right Shoulder Roll, Right Elbow Yaw, Right Elbow Roll and Right Wrist Yaw at the standby posture are 90.0, -11.5, 90.0, 0.0 are 90.0. All the angle values are in degrees.

We are moving the robot initially to a point where we hand over the ball to the robot. It picks it up from a specified position in x, y and z. Constraints in our case are joint ranges. We have to work within a workspace defined by joint maximum values. Joint ranges are presented in the table 3.1 and table 3.2 below.

Table 1 – Left Arm Joint nomenclature, joint axis revolution and ranges

Joint Name	**Motion Axis**	**Range "Degree"**
LShoulderPitch	Left shoulder joint front and back (Y)	-119.5 to 119.5
LShoulderRoll	Left shoulder joint right and left (Z)	-18 to 76
LElbowYaw	Left shoulder joint twist (X)	-119.5 to 119.5
LElbowRoll	Left elbow joint (Z)	-88.5 to -2
LWristYaw	Left wrist joint (X)	-104.5 to 104.5
LHand	Left hand	Open and Close

.

Table 2 – Right Arm Joint nomenclature, joint axis revolution and ranges

Joint Name	**Motion Axis**	**Range "Degree"**
RShoulderPitch	Right shoulder joint front and back (Y)	-119.5 to 119.5
RShoulderRoll	Right shoulder joint right and left (Z)	-76 to 18
RElbowYaw	Right shoulder joint twist (X)	-119.5 to 119.5
RElbowRoll	Right elbow joint (Z)	2 to 88.5
RWristYaw	Right wrist joint (X)	-104.5 to 104.5
RHand	Right hand	Open and Close

We have carried out our experiments on simulation initially and after gathering the results, we projected the same experiments on real robot. We have compiled all the results and we have presented those results and comparisons in this paper.

.

CHAPTER 4: GRASPING MODELS

As stated, fingers of NAO are not very sophisticated and cannot be controlled independently and efficiently. So, it was important to determine the suitable force for gripping the ball before lifting it to the back of head to avoid the spilling. For this purpose, we created two grasping models; Static and Dynamic.

4.1 Force Sensor Resistor (FSR)

Force Sensor Resistor (FSR) was used to determine the optimal grasping force. Arduino was used to process the data and was connected as shown in the figure 4.1. It was directly attached to multiple fingers on both hands separately and on the basis of distance between both hands at the time of gripping, data was collected. Process was repeated several times to get an average of all the readings. Optimal force was determined. We stored the reading in terms of joint angles and applied those readings to the task of grasping.

Components used for this circuit are;

- Arduino UNO
- 10K Resistor
- Force Sensor Resistor

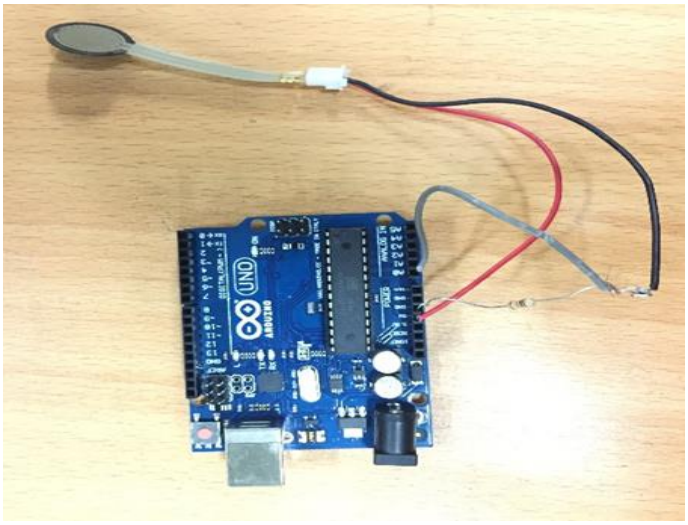

Figure 6 – Force Resistor Circuit – Patched, with one FSR

.

We have used the following voltage equation;

$$V_o = V_{cc}\left({}^{R}/_{R+FSR}\right) \qquad (1)$$

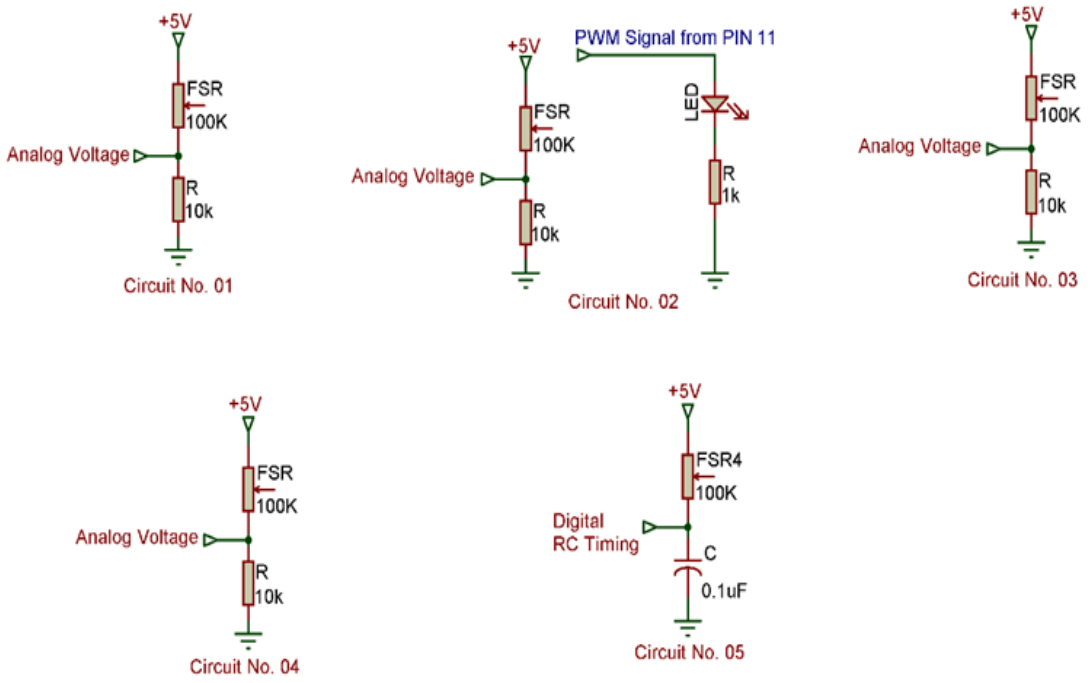

Figure 7 – Force Resistor Circuit – Schematic. With PWM and without PWM signal

4.2 Static Grasping Model

The main reason to obtain the different values of static grasping model and static grasping model was to make sure that our robot doesn't drop the ball throughout its trajectory execution.

The outline of Static grasping model starts with robot being at rest (Standby posture) initially. We place the ball at the pick-up location. Robot grasp the ball at that point and distance and force values are varied.

We carried out the same steps using different force values. One reference point in our case is to test the gripping by variation in distance between both the hands. The closer hands are, more force is being exerted on the ball.

.

$$F \infty S \qquad (2)$$

Where “F” is force exerted and “S” is the distance between both hands at particular exerted force.

4.3 Dynamic Grasping Model

Difference between Static and Dynamic Grasping models as indicated by the nomenclature is the motion. In the static grasping model, our robot grasps the ball from the pick-up location, follows via points and reach at the back of the head. Robot then slowly follows the designed trajectory of the throw-in, however, it does not release the ball because our main objective is to find out the optimal force at which robot grips the ball properly without spilling it in between.

We tested the force values multiple times to gather the optimal force and distance value. We compiled the results of both the models and utilized one optimal value for the rest of experiments.

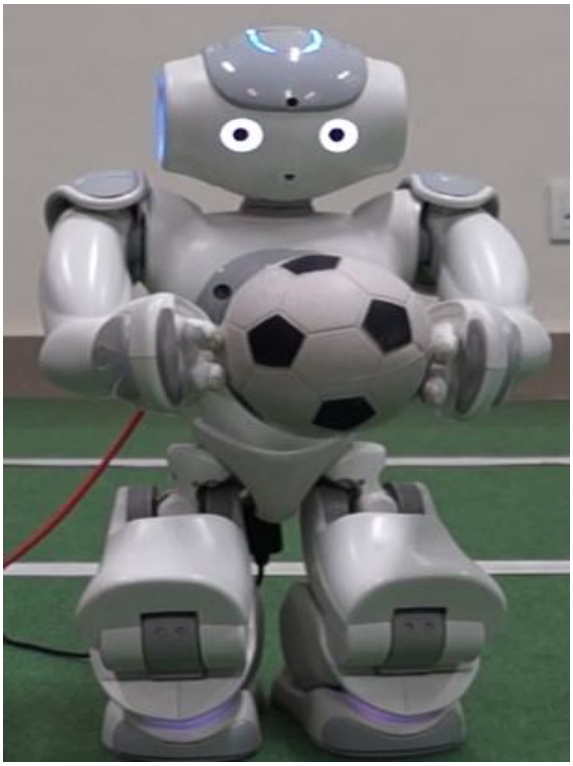

Figure 8 – Static Grasping Model – Posture

.

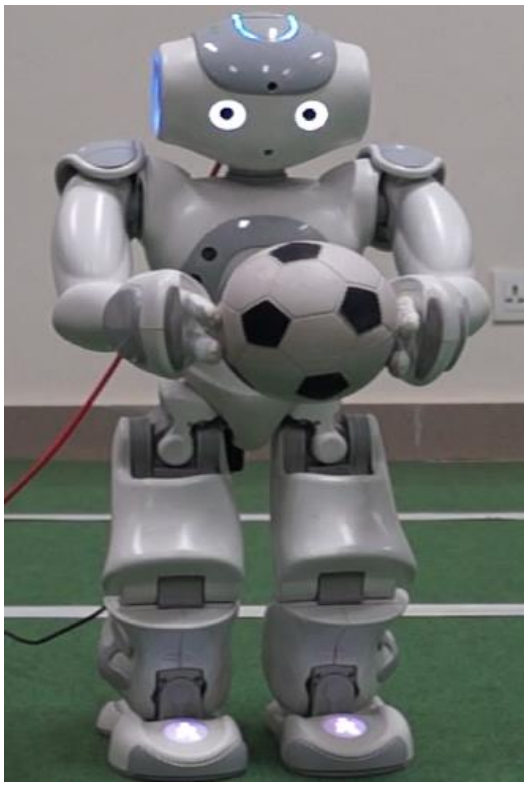

Figure 9 – Dynamic Grasping Model – Posture

Difference between Static and Dynamic Grasping models in terms of postures can be seen in the figure 8 and figure 9.

4.4 Graphical Results

We have accumulated the graphs between Distance and Force and Resistance and force. The graphs below shows the different experiment results.

.

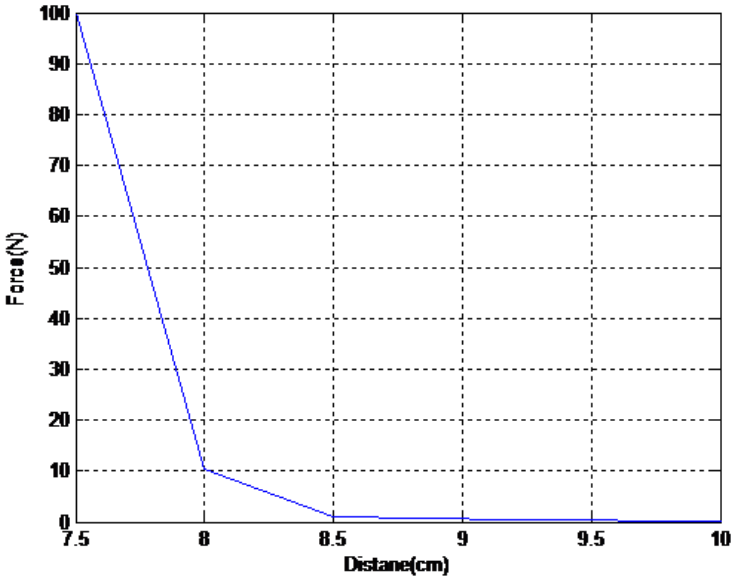

Figure 10 – Static Grasping Model – Posture

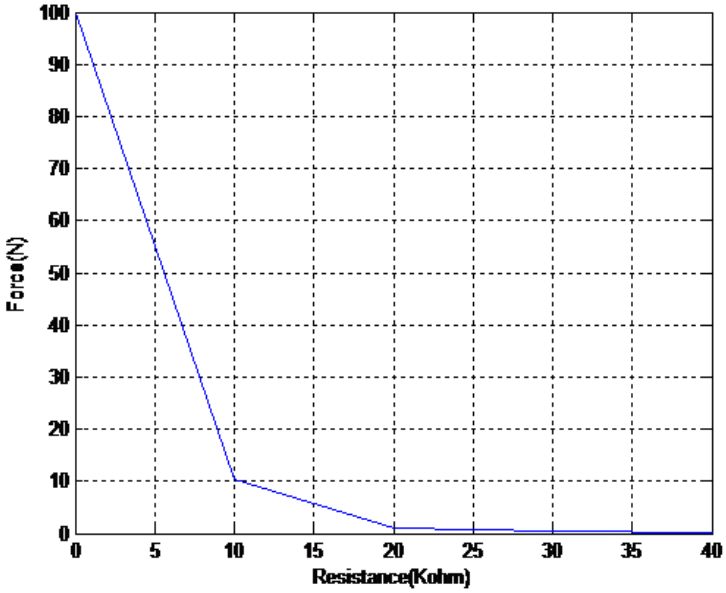

Figure 11 – Static Grasping Model – Posture

.

4.5 Initial Point Route

The results of Static and Dynamic Grasping models provided us with the optimal force. When the right force is applied, it ensures the perfect gripping and creating trajectories from then and there is next step but the factor of ball slippage is not a concern anymore.

In order to move the ball to the initial point, that in our case, we considered the back of the head, where the ball will be initially grasped by both hands before the throwing part, we had to ensure that while moving to that point hands do not touch the head, which is an obstacle. To avoid obstacles, we had to make sure that our hands follow certain via points to that initial point.

It wasn't just solving for the Cartesian co-ordinates but orientation matters a lot in this case, as the hands have to be in the right orientation at the initial points.

For orientation, we embedded Euler Angles in our code and determine our end effector's frame orientation of each instant. The orientation at the initial point is calculated using following equations;

$$r = atan2(rot(2,1), rot(2,2)) \qquad (3)$$

$$p = \text{asin}\big(-rot(2,0)\big) \qquad (4)$$

$$y = atn2(rot\ (1,0), rot(0,0)) \qquad (5)$$

Where r, p and y shows roll, pitch and yaw respectively and "rot" is the rotation matrix.

With the help of inverse kinematics, we can determine the joint angles required to reach the desired points in the Cartesian plane and in order to find the desired joint angles, we need orientation as well, as transformation matrix is;

$$T = \begin{bmatrix} Rot_{11} & Rot_{12} & Rot_{13} & T_x \\ Rot_{21} & Rot_{22} & Rot_{23} & T_y \\ Rot_{31} & Rot_{32} & Rot_{33} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (6)$$

In this 4 x 4 transformation matrix, upper 3x3 matrix is the orientation and 3 x 1 matrix at the end is translation matrix.

But since we are considering trajectory generation through via points, we need to use cubic splines to ensure smooth and jerk free motion. We have considered some limitation or constraints, such as initial velocity is V_o is zero and initial acceleration a_o is also zero when time "t" is zero. We have considered the final time to be 4 seconds as time to move the ball to the initial point is

.

not of concern.

General form of third degree cubic polynomial equation is;

$$\theta_t = a_o t + a_1 t^2 + a_2 t^3 \qquad (7)$$

This polynomial resulted in a jerk-free motion and we inducted 5 via points up to final position.

.

CHAPTER 5: EXPERIMENTAL RESULTS

We have used velocity and time as parameters for our experiments. We have determined the initial position, via points and final positions. Our model can be divided into 2 phases. Each portion is also divided into multiple sub portions.

Phase 1:

Our phase one starts from grasping to carrying the ball behind the head. This portion is tricky because the ball and hands cannot touch the head and it is important to determine the via points with close details.

- Initial position: Infront of the torso.
- Number of via points: 4
- Final position: Back of the head
- Time: 3 seconds

Phase 2:

Our second phase is starts from behind the head, i.e. the final position of the phase one and ends when the ball is released and the velocity is zero.

- Initial position: Back of the head
- Number of via points: 1
- Final position: Infront of the face
- Time: Variable. Depends on the velocity.

We considered 1-dimensional spline for a set of $n + 1$ points $(y_0, y_1, \ldots, y_n)$, which in phase 1 are 7. So the equation starts at y0 and ends at y6. 6th piece of the spline becomes y6 and replaces yi(t) in the following equation 1.

$$Y_i(t) = a_i + b_i t + c_i t^2 + d_i t^3, \qquad (1)$$

.

Figure 12 – Release Point

In the above equation, t0 = 0 and t1 = 3, it is because the total time to complete the interpolation (Phase 1) is 3 seconds.

where t is a parameter when t= [0, 3], and i[0, 6];

In order to find the time to reach each interpolation becomes 0.42 seconds.

General equation for the interpolation isl

$$Y_i(0) = y_i = a_i \quad (2)$$

$$Y_i(1) = y_{i+1} = a_i + b_i + c_i + d_i . \quad (3)$$

Derivative gives us;

.

$$Y_i'(0) = D_i = b_i \tag{4}$$

$$Y_i'(1) = D_{i+1} = b_i + 2c_i + 3d_i. \tag{5}$$

Solving equations for the instances will yield,

$$a_i = y_i \tag{6}$$

$$b_i = D_i \tag{7}$$

$$c_i = 3(y_{i+1} - y_i) - 2D_i - D_{i+1} \tag{8}$$

$$d_i = 2(y_i - y_{i+1}) + D_i + D_{i+1}. \tag{9}$$

Getting the second derivative, we get

$$Y_{i-1}(1) = y_i \tag{10}$$

$$Y_{i-1}'(1) = Y_i'(0) \tag{11}$$

$$Y_i(0) = y_i \tag{12}$$

$$Y_{i-1}''(1) = Y_i''(0), \tag{13}$$

Inferior and end points become

$$Y_0(0) = y_0 \tag{14}$$

$$Y_{n-1}(1) = y_n \tag{15}$$

Then,

$$Y_0''(0) = 0 \tag{16}$$

$$Y_{n-1}''(1) = 0. \tag{17}$$

Rearrangement will give us

$$\begin{bmatrix} 2 & 1 & & & & & & \\ 1 & 4 & 1 & & & & & \\ & 1 & 4 & 1 & & & & \\ & & 1 & 4 & 1 & & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & & 1 & 4 & 1 & \\ & & & & & 1 & 2 & \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_{n-1} \\ D_n \end{bmatrix} = \begin{bmatrix} 3(y_1 - y_0) \\ 3(y_2 - y_0) \\ 3(y_3 - y_1) \\ \vdots \\ 3(y_{n-1} - y_{n-3}) \\ 3(y_n - y_{n-2}) \\ 3(y_n - y_{n-1}) \end{bmatrix}. \tag{18}$$

System becomes;

$$\begin{bmatrix} 4 & 1 & & & & & 1 \\ 1 & 4 & 1 & & & & \\ & 1 & 4 & 1 & & & \\ & & 1 & 4 & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & & & 1 & 4 & 1 \\ 1 & & & & & 1 & 4 \end{bmatrix} \begin{bmatrix} D_0 \\ D_1 \\ D_2 \\ D_3 \\ \vdots \\ D_{n-1} \\ D_n \end{bmatrix} = \begin{bmatrix} 3(y_1 - y_n) \\ 3(y_2 - y_0) \\ 3(y_3 - y_1) \\ \vdots \\ 3(y_{n-1} - y_{n-3}) \\ 3(y_n - y_{n-2}) \\ 3(y_0 - y_{n-1}) \end{bmatrix}.$$

Real numerical data in the world is usually hard to analyze. It would be hard to acquire any feature that would efficiently correlate the information and would be extremely unfavourable. The concept of the cubic spline was created for this purpose. A series of distinctive cubic polynomials are attached between each of the information points using this method, stipulating that the acquired curve will be constant and appear smooth. These cubic splines can then be used over an interval to determine change rates and cumulative change. In this short introduction, we will only address splines that interpolate information points that are similarly spaced, although a more solid form could include points that are unequally spaced.

The basic idea behind cubic spline interpolation is based on the tool used by the engineer to draw smooth curves through a number of points. This spline is made up of weights attached at the points to be connected to a flat surface. Then a flexible strip bends over each of these weights, resulting in a pleasantly smooth curve.

The mathematical spline is in principle comparable. The points are numerical data in this situation. The weights are the coefficients used to interpolate the information on the cubic polynomials. These coefficients ' bend' the line to pass through each of the information points without any erroneous conduct or continuity breaks. Another option is spline interpolation, which includes a variety of interpolation methods that decrease overfitting impacts. The technique described here for cubic spline interpolation is commonly used in finance. It applies only in one dimension, but is helpful for modeling curves of yield, forward curves and other constructions of terms.

.

A cubic spline is a feature f: x[k], x[k+1], built by combining cubic polynomials pk(x) at various intervals. It has the shape

$$f(x) = \begin{cases} p_1(x) & x^{[1]} \le x < x^{[2]} \\ p_2(x) & x^{[2]} \le x < x^{[3]} \\ \vdots & \vdots \\ p_{m-1}(x) & x^{[m-1]} \le x \le x^{[m]} \end{cases}$$

Consider points (x[1], y[1]), (x[2], y[2]), ... (x[m], y[m]), x[1] < x[2] < ... < x[m]. By interpolating a cubic polynomial pk between each pair of successive points (x[k], y[k]) and (x[k+1], y[k+1]), we build a cubic spline according to the following limitations.:

1. Every polynomial goes through its designated end points:

$$p_k\left(x^{[k]}\right) = y^{[k]} \quad \text{and} \quad p_k\left(x^{[k+1]}\right) = y^{[k+1]}$$

2. First derivatives links at internal points:

$$\frac{d}{dx} p_k\left(x^{[k+1]}\right) = \frac{d}{dx} p_{k+1}\left(x^{[k+1]}\right)$$

3. Second derivatives links at internal points:

$$\frac{d^2}{dx^2} p_k\left(x^{[k+1]}\right) = \frac{d^2}{dx^2} p_{k+1}\left(x^{[k+1]}\right)$$

4. Second derivatives disappears at the end points

$$\frac{d^2}{dx^2} p_1\left(x^{[1]}\right) = 0 \quad \text{and} \quad \frac{d^2}{dx^2} p_{m-1}\left(x^{[m]}\right) = 0$$

The above circumstances indicate a linear equation system for the cubic spline that can be solved. In practice, fitting a cubic spline to less than five points makes little sense. However, let's interpolate a cubic spline between just three points for illustration purposes.

Take the points ($x^{[k]}$, $y^{[k]}$) = (1, 1), (2, 5), (3, 4). We find to fit a cubic polynomial on the range [1, 2] and another cubic polynomial on the range [2, 3]. These take the forms

$$p_1(x) = \delta_1 x^3 + \gamma_1 x^2 + \beta_1 x + \alpha_1$$

$$p_2(x) = \delta_2 x^3 + \gamma_2 x^2 + \beta_2 x + \alpha_2$$

Our starting condition requires

.

$$p_1(1) = 1$$

$$p_1(2) = 5$$

$$p_2(2) = 5$$

$$p_2(3) = 4$$

The second rule requires

$$p_1'(2) = p_2'(2)$$

The third rule requires

$$p_1''(2) = p_2''(2)$$

Finally, the last rule requires

$$p_1''(1) = 0$$

$$p_2''(3) = 0$$

We get eight equations in eight unknowns. Which can be presented as;

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 8 & 4 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8 & 4 & 2 & 1 \\ 0 & 0 & 0 & 0 & 27 & 9 & 3 & 1 \\ 12 & 4 & 1 & 0 & -12 & -4 & -1 & 0 \\ 12 & 2 & 0 & 0 & -12 & -2 & 0 & 0 \\ 6 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 18 & 2 & 0 & 0 \end{pmatrix} \begin{pmatrix} \delta_1 \\ \gamma_1 \\ \beta_1 \\ \alpha_1 \\ \delta_2 \\ \gamma_2 \\ \beta_2 \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 5 \\ 5 \\ 4 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Which we be solved to get,

.

$$\begin{pmatrix} \delta_1 \\ \gamma_1 \\ \beta_1 \\ \alpha_1 \\ \delta_2 \\ \gamma_2 \\ \beta_2 \\ \alpha_2 \end{pmatrix} = \begin{pmatrix} -1.25 \\ 3.75 \\ 1.50 \\ -3.00 \\ 1.25 \\ -11.25 \\ 31.50 \\ -23.00 \end{pmatrix}$$

Our final two polynomials are

$$p_1(x) = -1.25x^3 + 3.75x^2 + 1.50x - 3.00$$

$$p_2(x) = 1.25x^3 - 11.25x^2 + 31.50x - 23.00$$

5.1. Initial Point Details

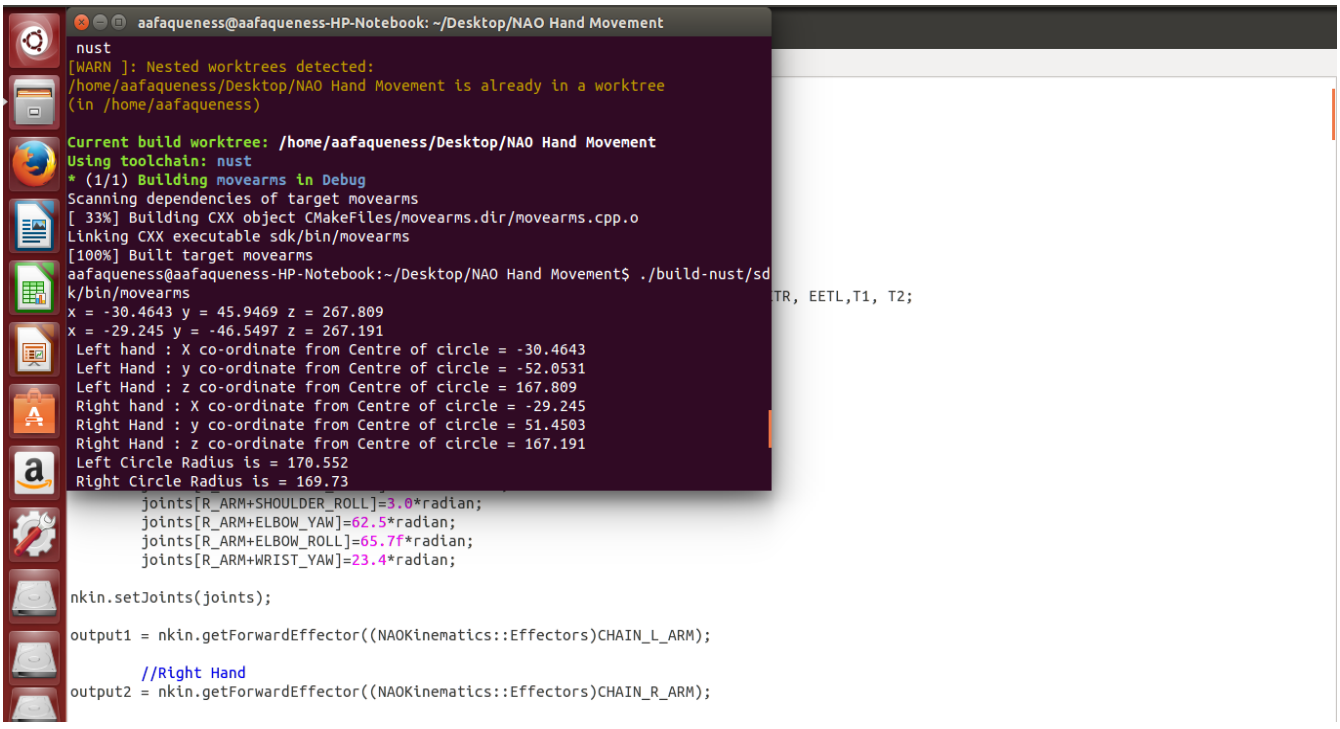

Figure 13 – Initial Point – Co-ordinates of joints

.

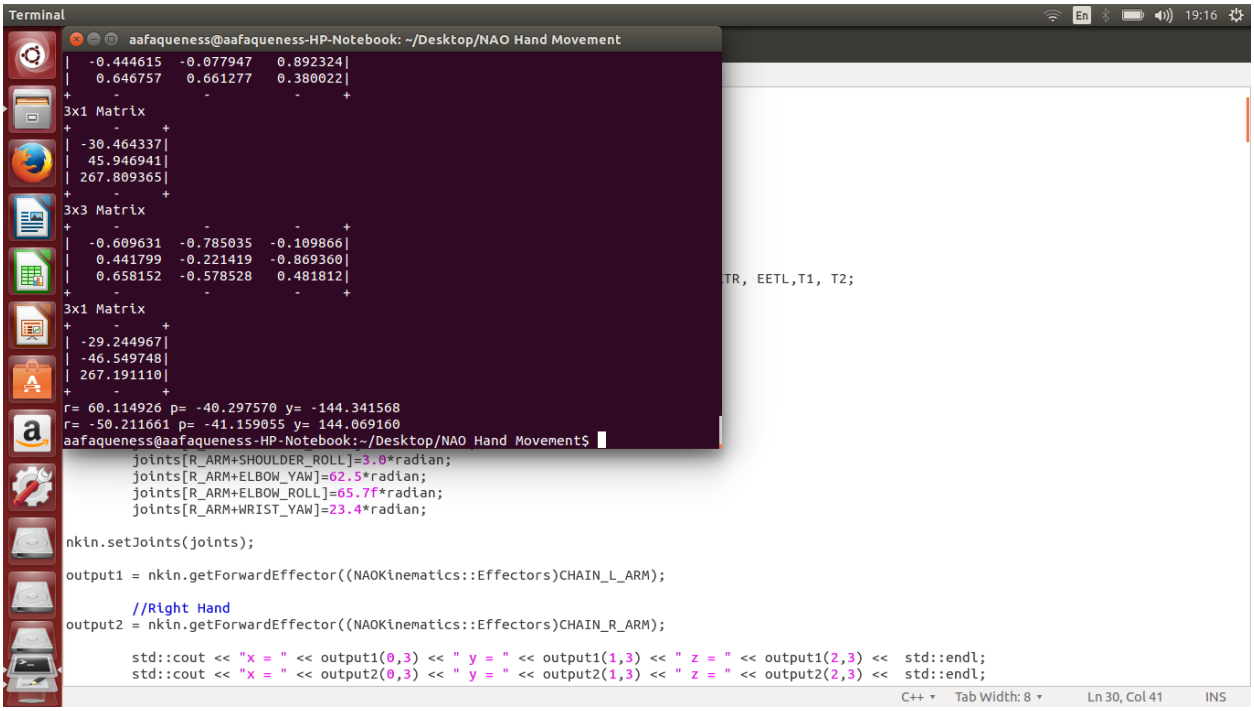

Figure 14 – Initial Point – Euler Angles, Orientation

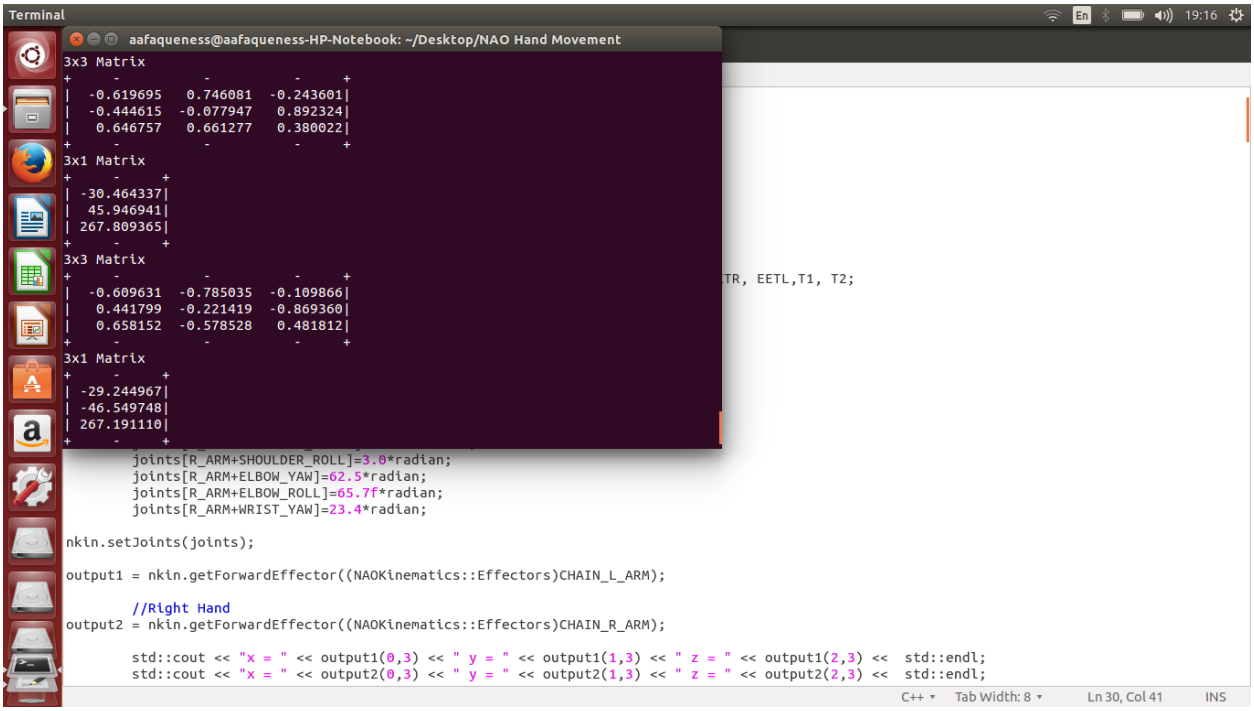

Figure 15 – Initial Point – Transformation Matrix

.

5.2. Final Point Details

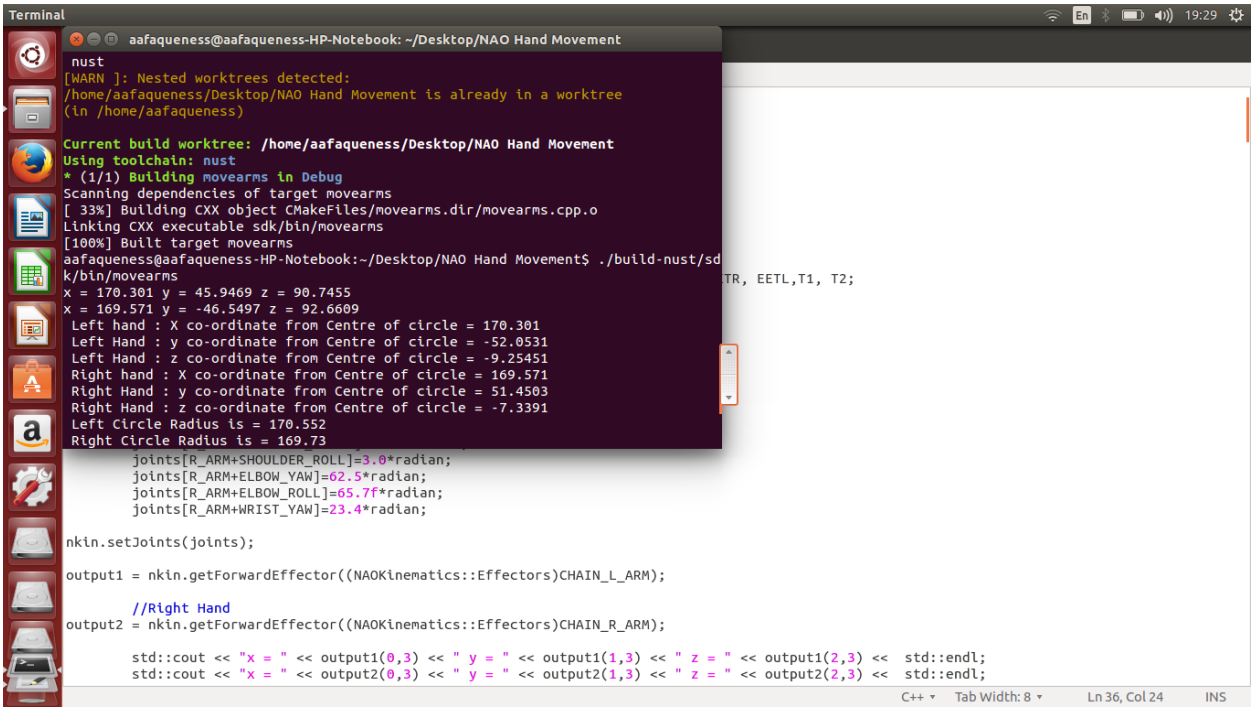

Figure 16 – Final Point – Co-ordinates of joints

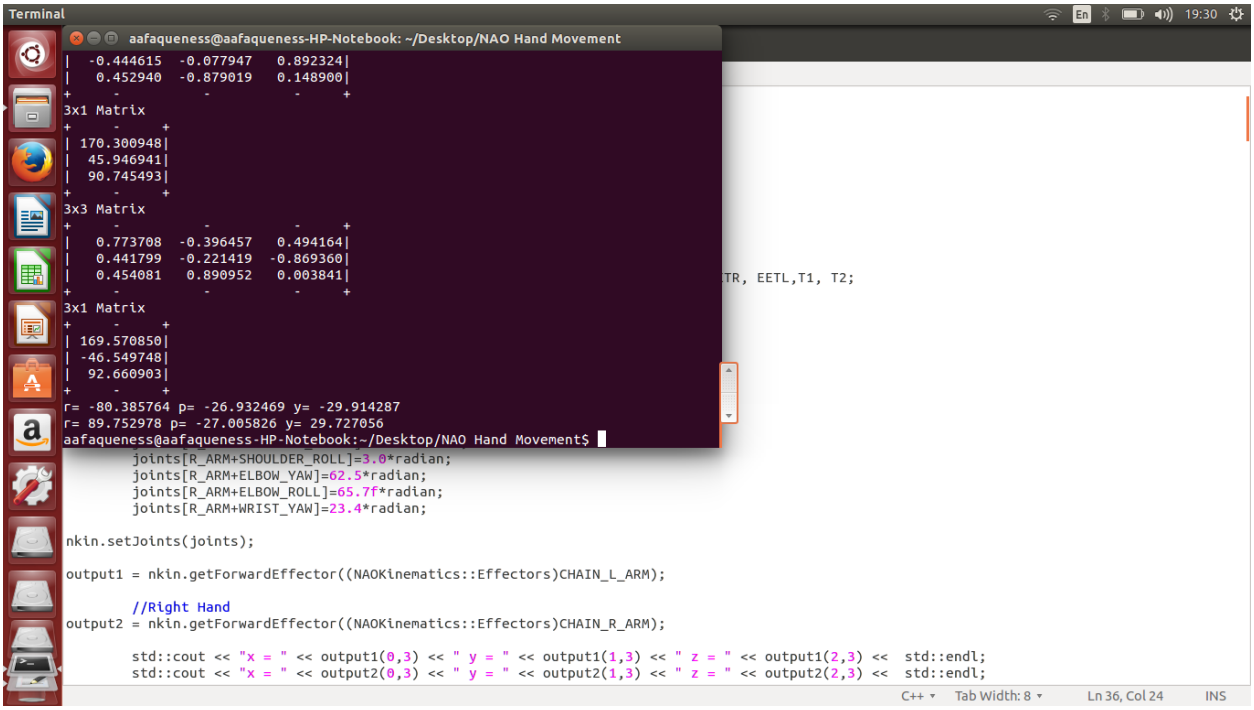

Figure 17 – Final Point – Euler Angles, Orientation

.

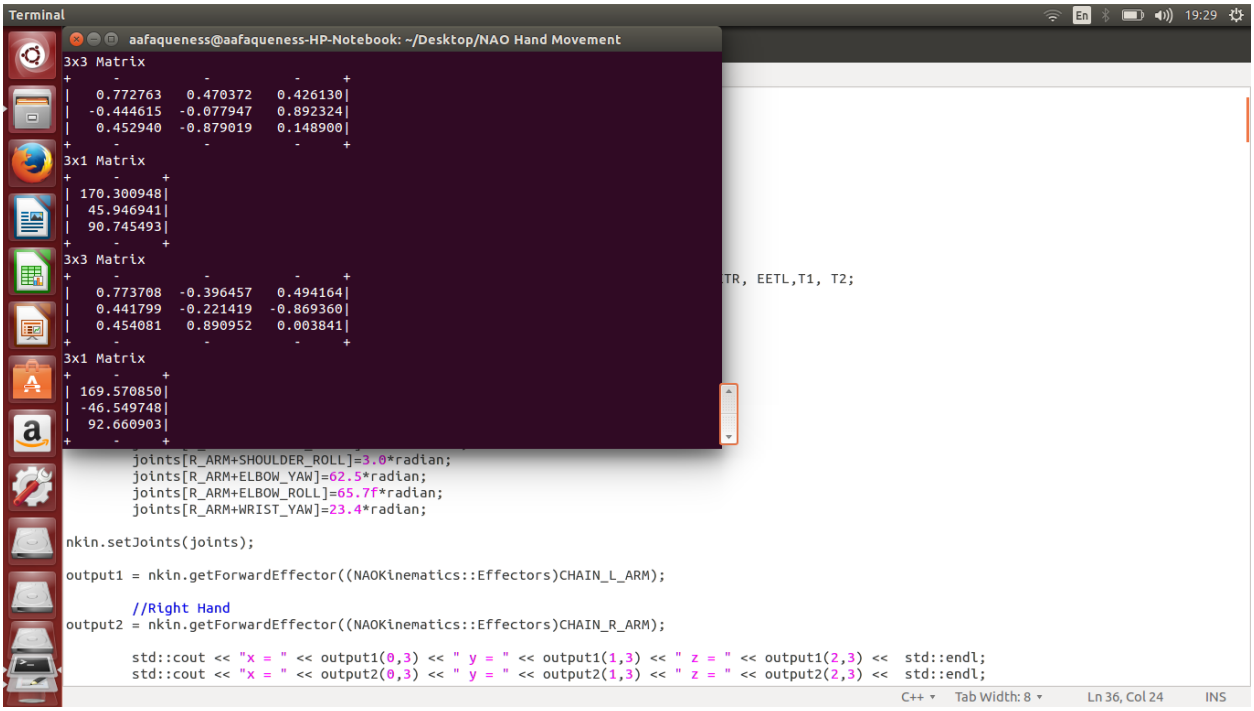

Figure 18 – Final Point – Transformation Matrix

Future Work

Future work includes the use of;

- Image processing
    - To differentiate between the team mate and opposite team player
- Artificial Intelligence
    - To analyze which player to choose while throwing the ball – Decision making

.

REFERENCES

[1] Campbell, Murray, A. Joseph Hoane Jr, and Feng-hsiung Hsu. "Deep blue." Artificial intelligence 134.1-2 (2002): 57-83.

[2] Prince, S. D., and S. N. Goward. "Evaluation of the NOAA/NASA Pathfinder AVHRR Land Data Set for global primary production modelling." International Journal of Remote Sensing 17.1 (1996): 217-221.

[3] Turing, Alan M. "Computing machinery and intelligence." Parsing the Turing Test. Springer, Dordrecht, 2009. 23-65.

[4] Kitano, Hiroaki, et al. "Robocup: The robot world cup initiative." Proceedings of the first international conference on Autonomous agents. ACM, 1997.

[5] Lin, Han. "Behavior Design of Nao Humanoid Robot: a Case of Picking up the Ball and Throwing into the Box." (2015).

[6] Mittal, Bhavishya, and Pratibha Prajapati. "Humanoid Robot: Throw Ball At A Target." (2014).

[7] Tsujita, Teppei, et al. "Humanoid robot motion generation for nailing task." Advanced Intelligent Mechatronics, 2008. AIM 2008. IEEE/ASME International Conference on. IEEE, 2008.

[8] Erkorkmaz, Kaan, and Yusuf Altintas. "High speed CNC system design. Part I: jerk limited trajectory generation and quintic spline interpolation." International Journal of machine tools and manufacture 41.9 (2001): 1323-1345.

[9] Gallant, André, and Clément Gosselin. "Extending the capabilities of robotic manipulators using trajectory optimization." Mechanism and Machine Theory 121 (2018): 502-514.

[10] Kucuk, Serdar. "Optimal trajectory generation algorithm for serial and parallel manipulators." Robotics and Computer-Integrated Manufacturing 48 (2017): 219-232.

[11] Moeslund, Thomas B., Adrian Hilton, and Volker Krüger. "A survey of advances in vision-based human motion capture and analysis." Computer vision and image understanding 104.2-3 (2006): 90-126.

.

[12] Anguelov, Dragomir, et al. "SCAPE: shape completion and animation of people." ACM transactions on graphics (TOG). Vol. 24. No. 3. ACM, 2005.

[13] Billard, Aude, et al. "Robot programming by demonstration." Springer handbook of robotics (2008): 1371-1394.

[14] Poppe, Ronald. "Vision-based human motion analysis: An overview." Computer vision and image understanding 108.1-2 (2007): 4-18.

[15] Cao, Y. Uny, Alex S. Fukunaga, and Andrew Kahng. "Cooperative mobile robotics: Antecedents and directions." Autonomous robots 4.1 (1997): 7-27.

[16] Dudek, Gregory, and Michael Jenkin. Computational principles of mobile robotics. Cambridge university press, 2010.

[17] Cao, Y. Uny, et al. "Cooperative mobile robotics: Antecedents and directions." Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots. Vol. 1. IEEE, 1995.

[18] Parker, Lynne E. "Current state of the art in distributed autonomous mobile robotics." Distributed Autonomous Robotic Systems 4. Springer, Tokyo, 2000. 3-12.

[19] Støy, Kasper. "Using Situated Communication in Distributed Autonomous Mobile Robotics." SCAI. Vol. 1. 2001.

[20] Kanayama, Yutaka, et al. "A stable tracking control method for an autonomous mobile robot." Proceedings., IEEE International Conference on Robotics and Automation. IEEE, 1990.

[21] Kanayama, Yutaka, et al. "A stable tracking control method for an autonomous mobile robot." Proceedings., IEEE International Conference on Robotics and Automation. IEEE, 1990.

.