People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

University M'Hamed BOUGARA – Boumerdes

Institute of Electrical and Electronic Engineering

Department of Electronics

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

MASTER

In **Electrical and Electronic Engineering**

Option: **Computer Engineering**

Title:

Implementation and Control of Biped Humanoid Robot Using Static Walk.

Presented by:

- **Youssouf Ismail Cherifi**
- **Zakaria Aroui**

Supervisor:

Mr. A.Mohammed Sahnoun

Registration Number:……/2015

Acknowledgements

First of all we would like to praise almighty ALLAH for giving us the power to realize this project.

We would also thank our supervisor Mr. A.Mohammed SAHNOUN who supported us, and helped us during the different phases of the project.

Many thanks are given to all the professors and workers at the IGEE. Especially, Dr. D.CHERIFI for guiding us during our project selection, and providing us with all the necessary needs required to realize the project.

At last, and not least, we would like to thank all our families' members and friends, especially Khaled, Abd El Ghani and Abdesselam without forgetting the aluminum shop and the smith who helped us with the building of the parts.

To our parents, families, friends, our beloved Dr. D.Cherifi and everyone who still has faith in this country – wonderful human beings, without whose patience, tolerance, understanding, compassion, support, and love, all our projects would not have been possible.

Abstract

Robotics is seen to be one of the most dominant aspects in several areas of interest by the human world, and one special type of them are humanoids, which are of an important use nowadays to ease the way of our life.
This project is about building a biped humanoid robot, and applying a control system to it, so that it can walk forward. This biped robot has a height of thirty three centimeters (33 cm), with 8 degrees of freedom.

Two main parts were required to implement this project, the construction part, and the control part.
First, the construction part, which is all about finding a suitable design for the biped robot, and building the right model of it in the real world. This part was handled as follows:

- Designing the mechanical structure of the biped robot and all its parts, and set the layout of its parts.
- Modeling the parts using one millimeter (1mm) aluminum sheets.
- Based on the overall design, we assembled the biped robot by adding the eight necessary actuators and the needed controller.

Second, the control part, which is about making the biped robot walk properly. We made use of the static walk approach for this task; that is maintaining the robot statically stable during both phases of its walking sequence, the double support phase, and the single support phase. This is achieved by keeping the robot's center of gravity within the support area.

To conclude, this project is about using static walking mechanism to make the biped humanoid robot walk on a flat surface, in hope of further improvements in the future.

List of Figures

List of Tables

Acronyms

- CoG Center of Gravity
- CoP Center of Pressure
- DoF Degree of Freedom
- PWM Pulse Width Modulation
- SSP Single Support Phase
- DSP Double Support Phase
- ZMP Zero Moment Point
- IDE Integrated Development Environment
- SDK Software Development Kit
- CAD Computer-Aided Design
- DH Denavit-Hartenberg
- USB Universal Serial Bus

Table of Contents

Introduction

The human life is so precious, which is the one reason that has allowed humanity to move forward during its early stages and even now. In order to preserve this life, researches and nights were spent to invent tools that can ensure proper development to our life; such tools allowed us to achieve things that seemed impossible to achieve by using our own naked efforts, other tools helped us overcome natural casualties, or predict them before they even happen, and even, there are tools used to protect us from wild animals and enemies that may threaten our lives.

Even so, some tools require a human contribution to operate, which makes those human beings in the presence of a possible danger, while other tools are themselves dangerous, and may cause real damage to their operators. For these particular reasons, remote and autonomous tools were built, and robots make a great part of those tools.

Nowadays we find different types of robots; specific task robots that are designed based on their specific task, which will allows them to perform their task impeccably, but because there are so many tasks that are required, either in industrial or domestic fields, the numerous designs implemented allow humans to do different tasks despite of having the same anatomy. This same anatomy is well designed that if applied on robots perfectly, then robots may relieve humans from ever risking their lives.

This type of robots is called Humanoid Robots, which are used as a research tool in several scientific areas.

Researchers need to understand the human body structure and behavior (biomechanics) to build and study humanoid robots. On the other side, attempting to simulate the human body will lead to a better understanding of its anatomy.

Human cognition is a field of study which is focused on how humans learn from their sensory information, in order to acquire perceptual and motor skills. This knowledge is used to develop computational models of human behavior, and it has been improving over time.

Once, it has been suggested that some very advanced robotics will facilitate the enhancement of ordinary humans.

Although the initial aim of humanoid research was to build a better orthoses and prostheses for human beings, knowledge has been transferred between both disciplines. A few examples are: powered leg prosthesis for neuromuscular impaired, ankle-foot orthosis, biological realistic leg prosthesis, and forearm prosthesis.

Besides research, humanoid robots are being developed to perform human tasks, like personal assistance, where they should be able to assist the sick and elderly, and do dirty or dangerous jobs. Regular jobs, like being a receptionist or a worker of an automotive manufacturing line, are also suitable for humanoids. In essence, since humanoids can use tools, operate equipment and vehicles that are designed for humans, then, humanoids could, theoretically, perform any task a human being can do, as long as they have the proper software and hardware. However, the complexity of doing so is deceptively great.

Humanoid robots, especially the ones with artificial intelligence algorithms, could be useful for future dangerous and/or distant space exploration missions, without any need to turn around and return back to Earth once the mission is completed.

The objective of our work is to design, model, implement and control a biped humanoid robot, and add some features to it to interact with the world around; and this work will be a starting point for us to develop more complex humanoids and integrate them in the medical use and our daily life

In the following chapters, we will uncover some of the subjects that are needed to implement one of the important parts in any humanoid robot; this part is the legs, they are so important that they are given a name on their own, which is Biped Robot.

In chapter 1, a simple analysis of the human anatomy is shown, along with the mechanisms that are required for a human being to walk. The importance of biped robots is also shown, including what makes those robots so special in comparison with other type of robots.

Chapter 2 is about the design of the implemented biped robot. Each part is shown on its own, along with the necessary geometrical parameters, and how all the parts fall along to make the final product. Also, a small description about the components that were used is introduced, which includes the actuators (mainly the servomotors), and the controller (the Arduino mega).

Chapter 3 is the most important of all, because it is about analyzing the necessary kinematics, which will be needed later in the control of the robot. This includes forward kinematics, DH parameters and calculation of the robot's center of mass.

Finally, in chapter 4, the used control method is explained, which is the Static walking, along with the analytic tools needed to ensure its success (inverse kinematics). Additionally, other means of controlling the biped robot are explained as well.

Chapter I

Human Anatomy and Biped Researches

1.1 Introduction

The human anatomy has a big influence in the development of robotics, because by analyzing and integrating certain organs and body parts, robots were able to do what they weren't able to do in a decade or so, and one of the most useful parts in a human body is the leg; hence, biped robots were developed.

In this chapter, the first chapter of our report, we are going to analyze the walking mechanism of the human race, this includes the anatomy of the human leg, the gait phases and how are we keeping our balance during our walking motion. After that, we will review the different research projects that were carried out to enhance the field of legged robot locomotion; but before that, here are some reasons for why going through all the trouble of building a biped robot.

1.2 History of biped robots

The history of robots goes back to the early ages of humanity, but at that time, the idea was only myths mentioned in ancient religions, such as the mythical creature Talos, which is considered to be made from bronze, with a body of half human and half bull. However the first attempt to create a self-automated machine was done by an ancient Greek engineer named Ctesibius, in 270 BC. The machine that he invented was a water clock, used to indicate hours based on the water level. Archimedes (278-212 BC) invented several machineries that were used later in the development of robots. Some of these machineries are still used nowadays in robotics.

In the medieval times, several human-like figures were used by the church to impress the worshippers. Those figures were run by a hidden mechanism, and the clock jack was one of those mechanical figures. It was a small robot that could strike time on a bell with its axe that was behind the operation of those figures.
In 1495 Leonardo da Vinci designed what may be the first humanoid robot. The design was made so the robot could sit up, wave its arms and move its head through a flexible neck, while opening and closing its jaw. In the 17^{th} and 18^{th} centuries, there were some major developments that caused the field of robotics to grow faster, such as the invention of the Pascaline calculating machine by Blaise Pascal in 1645, and its portable version in 1666, by Samual Morland. In addition to the invention of some powerful machines, such as the Steam Man in 1865, a human like machine that can pull wheeled carts. Also, ELEKTRO in 1937, a humanoid robot that can walk, talk and smoke.

These inventions caused uproar within the general public, especially after the release of several literatures and movies, where robots rebel against killing them or enslaving them.
It was then in 1942, that Isaac Asimov wrote the "Three laws of Robotics", which are as follows:
Law One: A robot may not injure a human (or humanity), or, through inaction, allow a human (or humanity) to come to harm.

Law Two: A robot must obey orders given it by human beings, except where such orders would conflict with a higher order law.
Law Three: A robot must protect its own existence, as long as such protection does not conflict with a higher order law.
A zeroth law was added later stating that a robot may not injure a human being, or, through inaction, allow a human being to come to harm, unless this would violate a higher order law.

After the Second World War the field has undergone a rapid growth, especially with U.S.A and U.S.S.R racing to discover the space. At that time, several reconnaissance robots and satellites were made, and by the time CPUs arrived, robots started to be used everywhere; there were no longer bounded to a single use.

Humanoids are becoming increasingly popular in providing entertainments too; for example, Ursula, which is a female robot that sings, plays music, dances, and speaks to her audience at Universal Studios. Several Disney attractions, in some of their theme park shows, use animatrons, robots that look, move, and speak much like human beings. These animatrons look so realistic, that it would be hard to distinguish, from a distance, whether they are real humans or not; and despite of their realistic look, they have no cognition or physical autonomy. Various humanoid robots, with their possible applications in daily life, are featured in an independent documentary film called Plug & Pray, which was released in 2010.

1.3 Why biped robots?

The human beings and almost all on land living animals use legs for locomotion. However not many machines were built using legs for movement. The reasons therefore are the complex design and control. Nevertheless, the main advantage of waking machines is that in contrast to wheeled robots, they do not need a customized environment. They could be able to move in an environment that is only accessible by human beings. In theory not only walking but also running, jumping, climbing or even swimming could be implemented. In contrast, wheeled machines need a relative planar terrain and enough space to avoid obstacles. Bipeds use different support areas for carrying their weight and getting grip and are in the ideal case as fast and flexible as a human. Using this flexible support on the ground, a large adaptability is achieved. The legs can also be considered as an individual suspension system whereby the upper part of the body moves forward on another trajectory as the feet. Decoupling the legs from the rest of the body allows carrying payload smooth through a rough terrain. Both types of robots are designed for a specified environment: The wheeled robots are more efficient on a planar surface whereas walking machines have an advantage on all other terrains.

The operational area of robots, especially with two legs, is the natural setting of humans. The human body has, because of his anatomy, an exceptionally maneuverability which is perfect exploited for his locomotion. Thus, he can adapt to a new environment with minimal effort.

1.4 Basics of two legged walking

To understand the topic of the biped walking an overview of a human model will be shown. For the reason that most of the humanoid robots use the human body as paragon, it is suggestive to use the same terminology as for the human anatomy. As shown in Fig 1.1, there are three basic planes referred to as frontal (or coronal), sagittal and transversal [1].

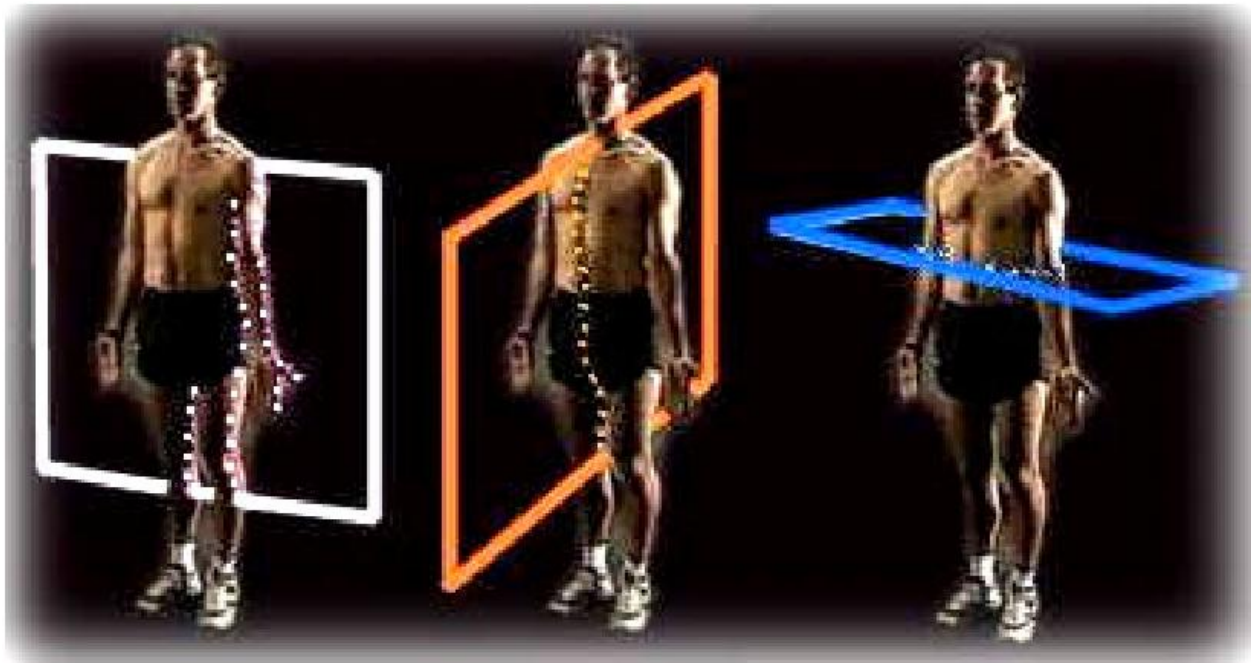


Fig 1.1: *The three anatomic planes: frontal, sagittal and transversal.*

1.4.1 Gait phases

Walking is a cyclic movement consisting of two main phases, which alternates on both legs (see Fig 1.2 and 1.3).
During the double support phase (I), both feet are in contact with the ground. In this phase, the body has a stable position because of the wide support area on the ground. The system enters this state with the heel strike (IV) and exits it with the toe off (II) movement.

During the single support phase, only one foot is in contact with the floor. In this state, the center of mass (COM) of the system rotates like an inverted pendulum above the contact point. Meanwhile the swing leg moves forward (III) to touch the ground again and enter the other phase.

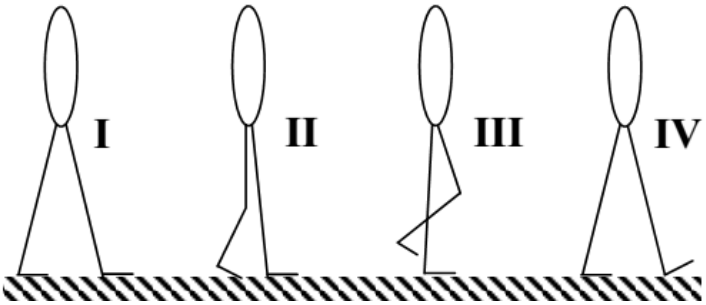


Fig 1.2: *Leg position during one-half cycle.*

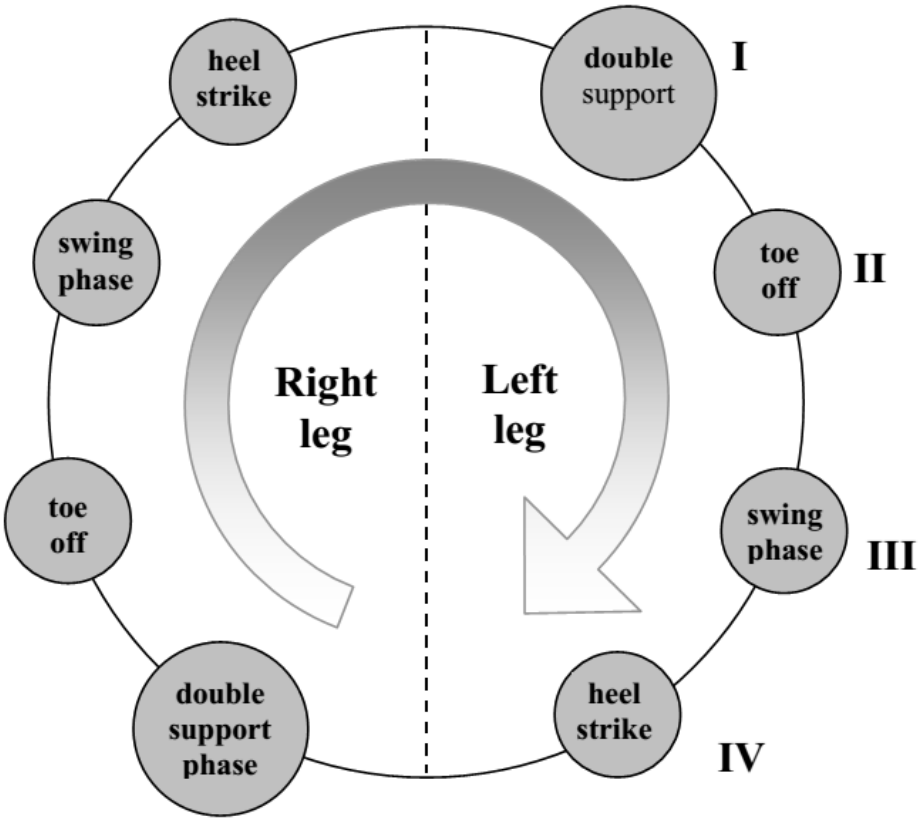


Fig 1.3: *The cyclic phase rotation of biped walking.*

The COM of a walking human oscillates continuously four centimeters up and down and simultaneously left and right (we will talk about this in chapter IV). The point is located in the hip (see Fig 1.4). For the development of a walking gait, there are three important parameters (see Fig 1.4): the step length (d), the step height (f) and the step period [2]. The controller can use these parameters to stabilize the gait. They also affect directly the speed and the position of the entire system.

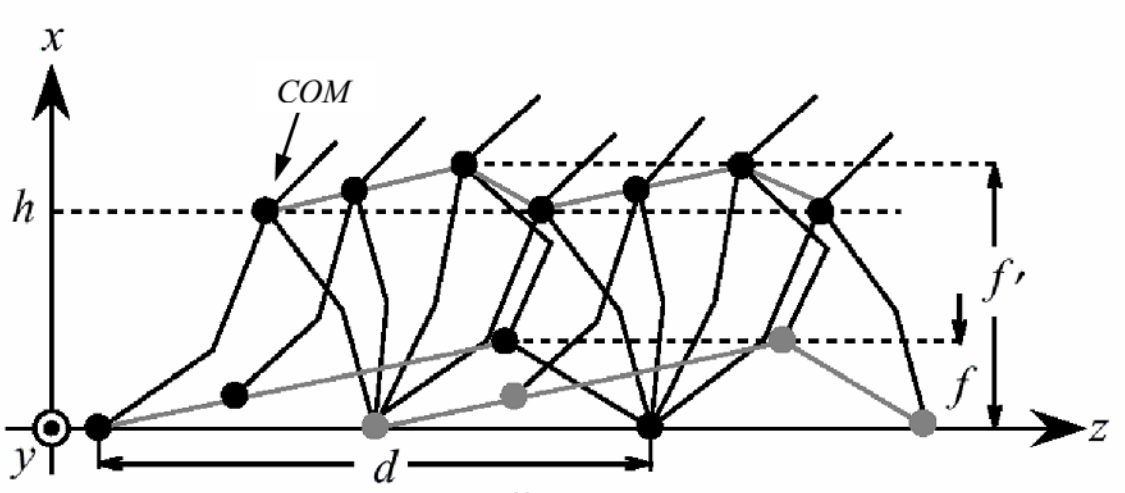


Fig 1.4: *Side view of the movement of the legs during a walking gait.*

1.4.2 Static and dynamic balance

Walking can be divided into two main groups: walking with static balance and walking with dynamic balance.

During a static walk the normal projection of the center of mass (NPCM) always stays in between the boundaries defined by the feet. If both feet are on the ground, the NPCM has to be within the polygon determined by the outer corners of the biped feet. If only one foot is in contact with the ground, the NPCM has to be within the area of this foot. While the movement is slow enough, the system dynamics can be ignored. Static walking assumes that if the system's motion is stopped at any time, it will stay in a stable position indefinitely. However, the speed achieved using static walk is not that high and the efficiency is far away from the human walking speed.

To introduce the semi-dynamic and the dynamic walk, another terminus has to be explained. The Zero Moment Point (ZMP) is the point on the ground where the sum of all moments on the robot is equal to zero. During a dynamic movement, normally this point has to be within the boundaries of the feet exactly as the NPCM during a static movement [3].
The semi-dynamic walk is similar to the static walk except short periods of time, in which the whole system tends to be unstable and the ZMP can exceed the stability region provided by the feet. The body may be falling during this part of the gait, and unless the feet are positioned correctly, it could fall to the ground.
A dynamically stable biped, by comparison is one that moves through unstable positions in its walking gait, and needs to intelligently adjust and plan its movements to remain stable at any given time. In this walking gait, the ZMP can also move outside the supported region for a finite amount of time, but is generally in constant movement, thus the feet are in continuous motion.

Human being and animals rarely use static walk. To achieve better results with respect to speed and efficiency some kind of controlled instability must be introduced to become more similar to the paragon of nature.

1.4.3 Walking and running

The locomotion for bipeds can be divided into two main groups, walking and running. The actual research on this field concentrates principally on walking. The two forms have very different characteristics, according to the movement. However, the main difference is the contact of the feet with the ground. During walking at least, one foot is on the ground at all times, and during the double support phase even two. Whereas while running only one foot touches the ground simultaneously. Furthermore there are only short period in which the foot has ground contact followed by a long time span with ballistic movement.
Running has a few advantages. The most important advantages are the higher velocity and the high efficiency. Running allows an elastic energy recovery during the jump phase. Therefore, a running robot must have an elastic mechanism to absorb the kinetic energy and give it back in the right moment.

1.5 Review and research

“Only if a robot is able to move free in our environment, he will one day be able to be a real help for human being and carry the name humanoid robot”. With this sentence Honda’s robot Asimo (Advanced Step in Innovative Mobility) was presented in 2001 to the world (see Figure 2.5). He was the improved successor of Honda’s P2 and P3 which were built in the lately 90’s. With a weight of 43kg, he is much lighter than his predecessors are and that is one of the most important advantages. This robot is the result of about 14 years of research and approximately 10 prototypes. ASIMO can walk continuously while changing directions, taking every step smooth and natural. He also has the ability to climb up and down a flight of stairs [4].

Fig 1.5: *The humanoid robot ASIMO.*

The research of legged machines begun in the early 80's and the Massachusetts Institute of Technology (MIT) Leg Laboratory was one of the first in developing a variety of walking, running or jumping machines.

The first Leg Lab robot was the Planar One-Leg Hopper (see Fig 1.6) [5]. It had just one leg with a small foot. It was designed to explore active balance and dynamic stability in legged locomotion. It was controlled with a simple three-part algorithm.

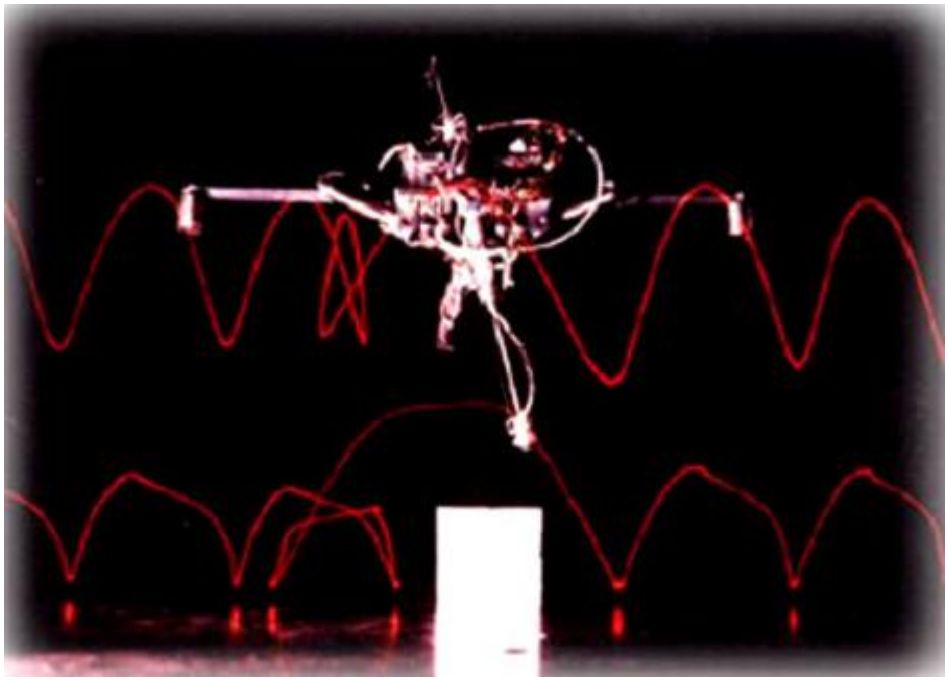


Fig 1.6: *Planar One-Leg Hopper from the MIT (Raibert, M. H. 1980-82).*

Following machines were built to show that actively balanced dynamic locomotion could be accomplished with simple control algorithms. The 3D One-Leg Hopper (see Fig 1.7), for example hopped in place, travelled at a specified rate, followed simple paths, and maintained balance when disturbed.

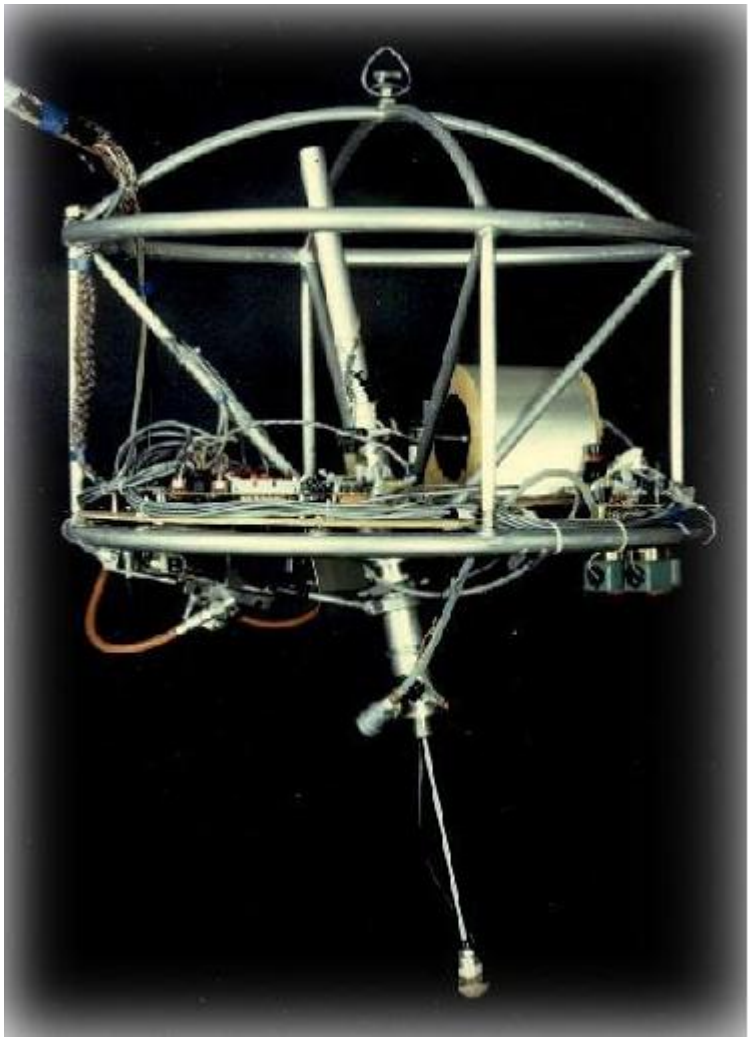


Fig 1.7: *3D One-Leg Hopper from the MIT (Raibert, M. H. 1983-84).*

The research group around M. Raibert at the MIT showed that a one legged robot could stay upright with active balance, using separate control algorithm for hopping and forward movement. Furthermore, he showed that the jumping-technology could easily be upgraded to a biped running.

At the University of Western Australia (UWA) at the department of Electrical, Electronic and Computer Engineering the research group around T. Bräunl had a breakthrough in 1998 with the design of robot called "Johnny Walker" (see Fig 1.8, left). The robot used nine servos and an onboard 32-bit Controller. Each leg had 4 degrees of freedom (DOF) and with this robot; an investigation of different dynamic walking gaits was possible.

Fig 1.8: *Walking bipeds Johnny walker, Jack Daniels and Andy.*

The main problem encountered on this first prototype was the heavy frame. The consequence was a heavy robot, which required high torques to move the legs. This problem was eliminated on another prototype called Andy (see Fig 1.8, right). The whole design was revised and for better liberty of action, the robot was provided with five DOF per leg. The result was a much lighter and flexible robot. Actual research is now taking place on this walking machine with promising results.

1.6 Summary

This chapter of the report served as motivational entry to the implemented project, by specifying the different needs for such robots. Additionally, it got through all the required analysis of the walking mechanism of the human body, this included the human gait phases, how does the body stabilize itself during these phases, and what differentiate a walking from a running.

The chapter also pointed to the different research projects that were carried out in order to improve the field of legged robot locomotion, some good example of that were, the HONDA's famous robot ASSIMO, and several other projects implemented by the famous engineering university MIT.

Chapter II

Robot Structure and Peripherals

2.1 Introduction

The previous chapter explained how the biped robots (humanoid robots in general) are based on the human anatomy. This key feature will allow robots of the future to possess human capabilities, allowing them to replace us on most of the jobs that are deemed dangerous or life threatening.

Considering the advantages of biped robots, we are going to implement our own biped robot. Throughout this chapter we will go through the conception and design of the entire robot. First, we are going to describe the robot's frame, which will cover the parts that make up the frame, the material of which these parts are built with and how they are designed. Then, the geometrical and physical properties, once the entire frame is assembled.
Next, we will explore the chosen controller and actuators, as well as why these components are selected, based on the robot's properties (both physical and geometrical).

2.2 Robot structure

The implemented biped robot is a multi-link object, where 17 parts are used to form this multi-link object, and because of that, the material of construction need to be carefully chosen. The matter that was chosen for building these parts was aluminum, considering that it has small density (evaluated by 0.0027 g/mm3), and a solid structure. By choosing aluminum the strain on the joint is reduced, while getting a defined structure.
Out of the 17 parts we extinguish only 7 different types of construction. Fig 2.1 shows a plan of the different parts, and Fig 2.2 displays a CAD implementation of each part using PTC Creo modeling and simulation software.

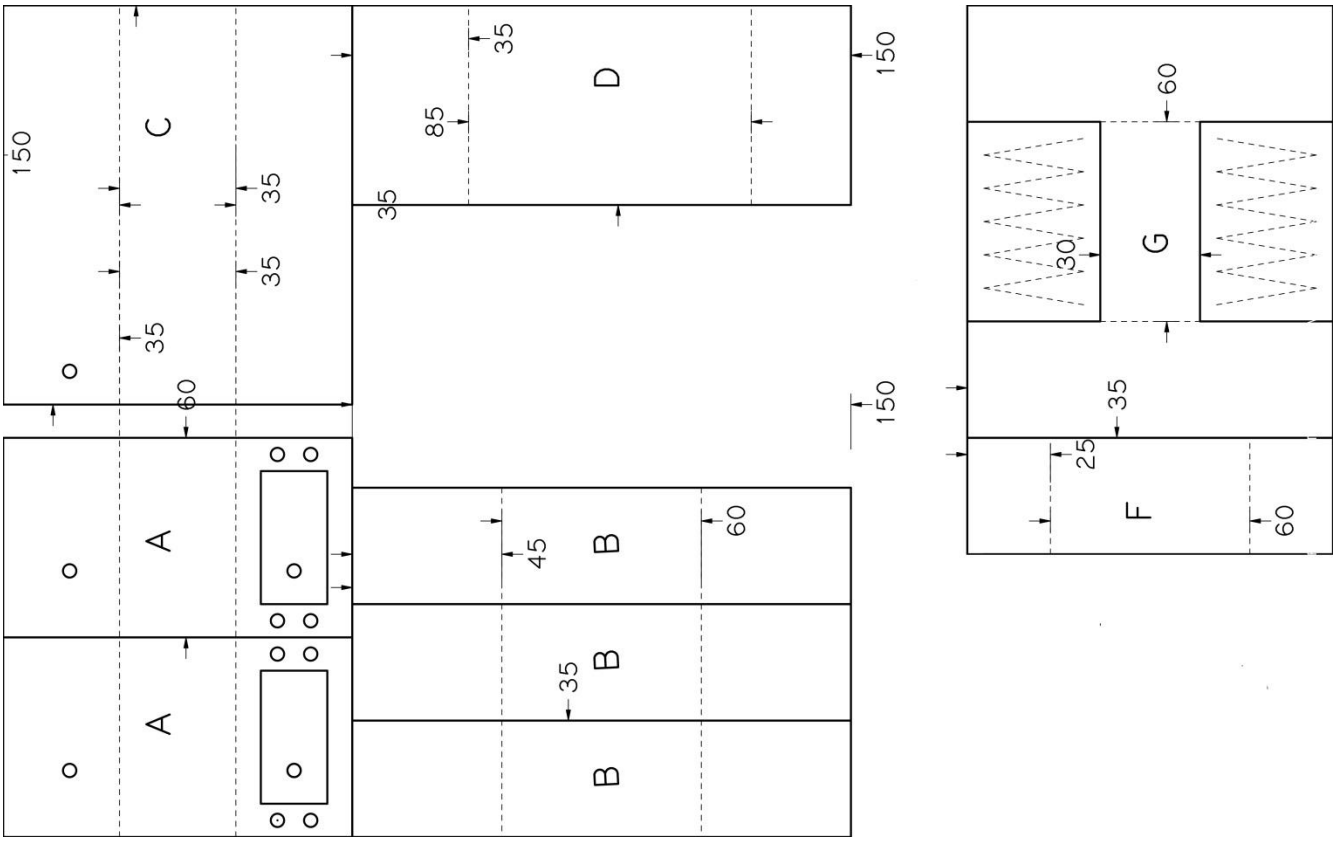


Fig 2.1: *Parts plan - using CADStd.*

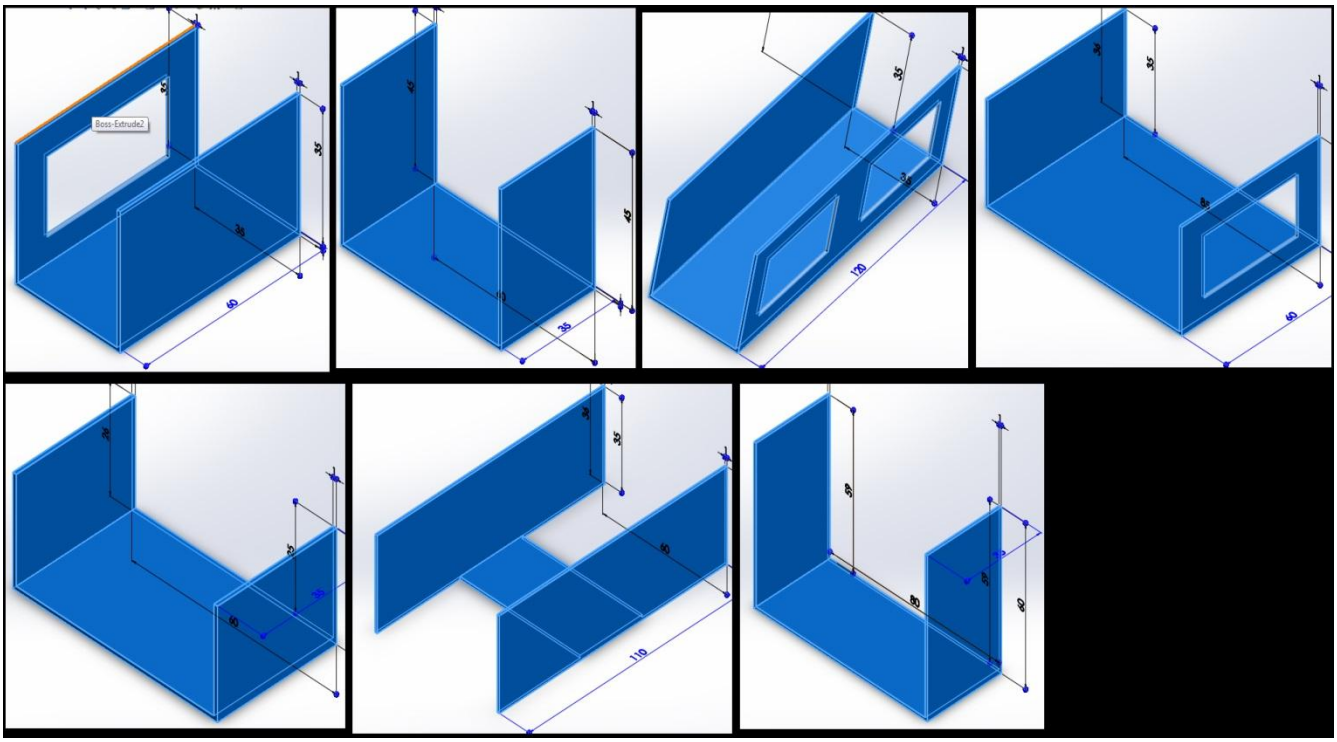


Fig 2.2: *CAD design for the 7 parts with dimensions – using PTC Creo.*

2.2.1 PTC Creo

Creo is a family or suite of design software supporting product design for discrete manufacturers and is developed by PTC. The suite consists of apps, each delivering a distinct set of capabilities for a user role within product development. PTC Creo is also scalable and interoperable which allows it to deliver fast time to value. It helps teams create, analyze, view and leverage product designs downstream utilizing 2D CAD, 3D CAD, parametric and direct modeling.

Fig 2.3: *PTC logo.*

Creo runs on Microsoft Windows and provides apps for 2D design, 3D CAD parametric feature solid modeling, 3D direct modeling, Finite Element Analysis and simulation, schematic design, technical illustrations, and viewing and visualization.
The Creo suite of apps replaces and supersedes PTC's products formerly known as Pro/ENGINEER, CoCreate, and ProductView.

Creo consists of the nine apps:

- Creo Parametric is a 3D design app for parametric modeling (parametric featured based solid modeling). Creo Parametric provides all the capabilities of Creo Elements/Pro (also known as Pro/ENGINEER).
- Creo Direct is a standalone design app for expert and non-expert CAD engineers who want to interact directly with the geometry using a direct modeling approach.
- Creo Simulate is the analyst app used for structural and thermal simulation. This app leverages technology from Pro/MECHANICA.
- Creo Layout is a design app for engineers who want to create concept layout work in 2D, with the intention of ultimately evolving the design to 3D.

2.2.2 Why PTC Creo?

There are two main reasons for choosing PTC Creo instead of any other CAD designing software and they are:

- PTC Creo offers an educational-use-only activation, which allows students and teachers to use the software for free. Except that the Creo Simulate requires a pro activation.
- PTC Creo is easy to use, and offers helpful tools that will come in handy as we will see in the next chapter (mass property, measure…).

Now that each part is designed and implemented separately, all what is left is to assemble them as shown in Fig 2.4.

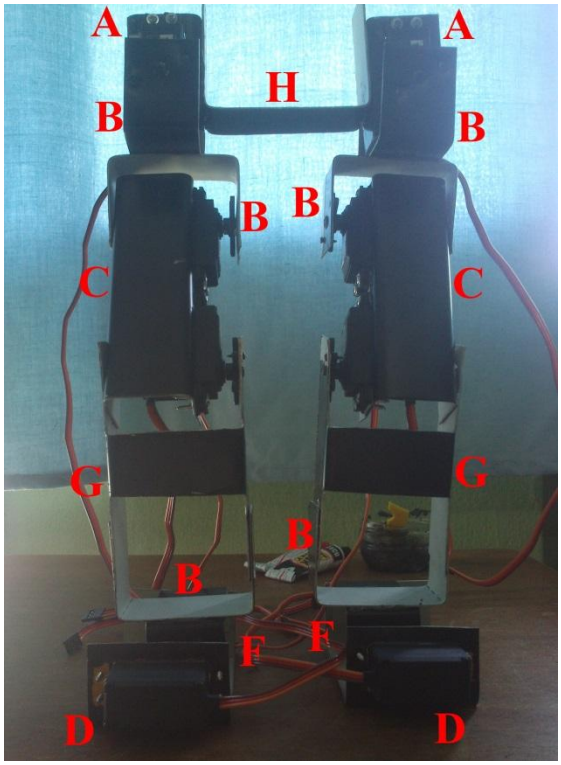


Fig 2.4: *Biped implementation with indexes to all parts.*

As shown, the implemented biped robot has 8 DOF, which means that each leg has 4 joints, two at the hip, one at the knee, and another at the ankle. This type of biped is called Dany-Walker, and they are typical for testing different aspects of bipedalism. Additionally, Dany-walker can be used to implement dynamic walk and give the robot the ability to walk side way.
Within PTC Creo an assembly file is created, so that the robot can be analyzed within the software. In the assembly file, all the previously built parts are imported and assembled in a way that matches the real biped implementation shown in Fig 2.5.

In the figure, the CAD design from a different angle, along with the robot geometrical property; while Table 2.1 describes both geometrical and physical properties of all parts.

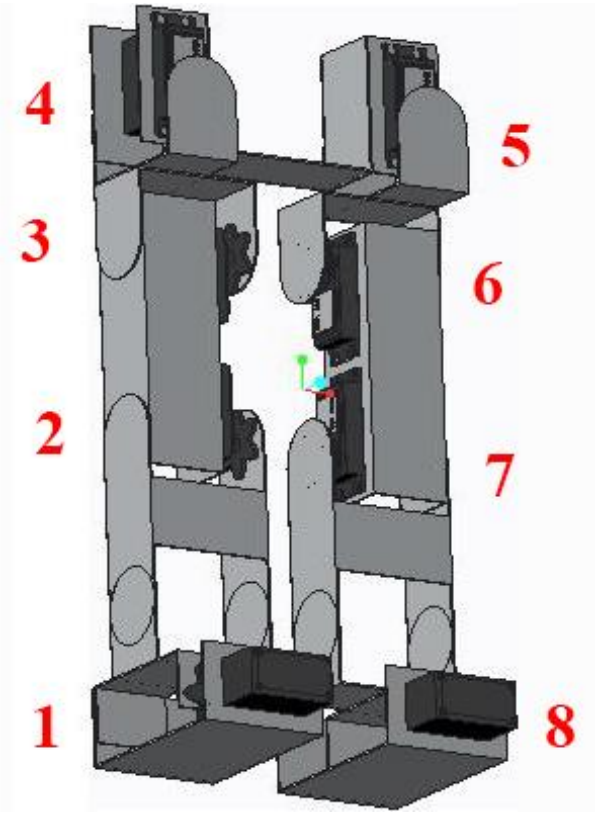


Fig 2.5: *CAD implementation of the entire biped using PTC.*

Part	Weight (g)	Volume (mm3)	Dimensions (mm)		
			Width	Length	Height
Part A	14.75	5465.65	35	60	35
Part B	13.18	4882.11	35	60	45
Part C	29.51	10931.3	35	120	35
Part D	22.86	8265.65	60	85	35
Part F	10.68	3955	35	65	25
Part G	23.99	8884.23	60	110	35
Part H	18.90	7000	35	80	60
Motor	60	32379.5	20.5	40.7	39.5
Leg	293.4	139139	35	60	254
Biped	781.58	370732	35	153.4	326.5

Table 2.1: *Geometrical and physical properties of the different parts.*

2.3 Dynamic requirements

Even though the chosen Aluminum material is the perfect one for reducing the strain on the joints while keeping the robot intact, we still need to quantify that strain to choose the appropriate actuator, because not any actuator can get the job done.
By quantifying the strain, we mean calculating the necessary torque required to move one leg up and down without getting a false reading.

Torque is the tendency of a force to rotate an object about an axis [6], a fulcrum, or a pivot. Just as a force is a push or a pull, a torque can be thought of as a twist to an object. Mathematically, torque is defined as the cross product of the lever-arm distance vector and the force vector, which tends to produce rotation. Loosely speaking, torque is a measure of the turning force on an object, such as a bolt or a flywheel. For example, pushing or pulling the handle of a wrench connected to a nut or a bolt produces a torque (turning force) that loosens or tightens the nut or bolt.
The magnitude of torque depends on three quantities: the force applied, the length of the lever arm connecting the axis to the point of force applied, and the angle between the force vector and the lever arm. In symbols:

$$\tau = r \times F \tag{2.1}$$

$$\tau = \|r\| \|F\| sin\theta \tag{2.2}$$

Where,

τ *tau is the torque vector, and* τ *is the magnitude of the torque,*

r *is the displacement vector (a vector from the point from which torque is measured (typically the axis of rotation) to the point where force is applied),*

F *is the force vector,*

× *denotes the cross product,*

θ *is the angle between the force vector and the lever arm vector.*

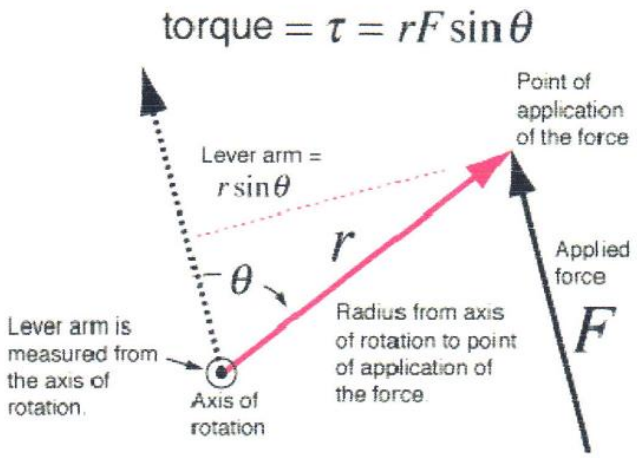


Fig 2.6: *Torque calculation.*

The length of the lever arm is particularly important; choosing this length appropriately lies behind the operation of levers, pulleys, gears, and most other simple machines involving a mechanical advantage.

Using the parameters of a single leg that are shown in Table 2.1, we can use equation 2.2 to get the required torque for our implementation.
F is taken to be the weight exerted by a single leg,

$$F = m \times g = 0.2934\,kg \times 9.80\,\frac{m}{s^2} = 2.875N.$$

R is taken to be 25.4 cm which is the leg height.
θ is taken to be the maximum supported angle which is 90°, then,

$$T = 2.875N \times 25.4cm = 73.033\,N.cm$$

Motors' specifications use a torque evaluated in kg.cm which is equal to the actual torque divided by the gravitational acceleration so that our torque will become, $T_{motor} = \frac{73.033\,N.cm}{9.8\,\frac{m}{s^2}} = 7.45\,kg.cm.$

2.4 Actuators

The main actuator that is used in the implemented system (the biped robot) is the servomotor, which represents a joint within the robot.

2.4.1 Servomotor

A servomotor is a rotary actuator that allows a precise control of angular position, velocity and acceleration. It consists of a suitable motor coupled to a sensor, for position feedback. It also requires a relatively sophisticated controller, often, a dedicated module designed specifically to be used with servomotors.

2.4.2 Why servo-motors

Servomotors are chosen for the following reasons:

- Servomotors simplify the task of constructing the robot, since they have a digital input representing the desired angle.
- Servomotors are more accurate than the rest type of motors.

2.4.3 The chosen servomotor

Since our required torque is evaluated by 7.45 kg.cm, then a servomotor that provides higher torque is needed for our case. The chosen servomotor is the Model 1501MG from the Power HD brand. The gears inside this servomotor are made of metal.

Name: 1501MG Power HD

Operating speed (at no load): 0.16 sec/60° (4.8 V)
0.14 sec/60° (6 V)

Running current (at no load): 400 mA(4.8 V)
500 mA(6 V)

Stall torque (at locked): 15.5 kg-cm(4.8 V)
17 kg-cm (6 V)

Stall current (at locked): 2300 mA (4.8 V)
2500 mA(6 V)

Idle current (at stopped): 4 mA (4.8 V)
5 mA (6 V)

Fig 2.7: *Power HD 1501 parts and specs [7].*

2.5 Controller

Now that we have the frame and the actuators (joints), all that is left is to choose the right controller, which represents the brain of the robot. For this purpose, the Arduino Mega microcontroller is chosen.

2.5.1 Arduino Mega

The Arduino Mega is a microcontroller board based on the ATmega1280. It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable, power it with an AC-to-DC adapter or a battery, in order to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila [8].

Microcontroller	ATmega1280
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	128 KB of which 4 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Fig 2.8: *Arduino Mega with specs [8].*

2.6 The electrical circuit

With all the electrical parts of the biped robot gathered together, we implemented the following electrical circuit of our robot as shown in Fig 2.8.1.
Each one of the servomotors is connected to a digital pin on the Arduino board via its control line, as shown in Table 2.1.1. The two other lines of each servomotor are the power lines; those power lines are all connected in parallel to an external power supply, which supplies a sufficient voltage of 6Volts and a current of 1Amps. Also, the ground of the power supply is connected with the ground of the Arduino board.
The Arduino board is powered using the USB port connected to a computer host, which supplies a sufficient voltage of 5Volts needed to power the board.

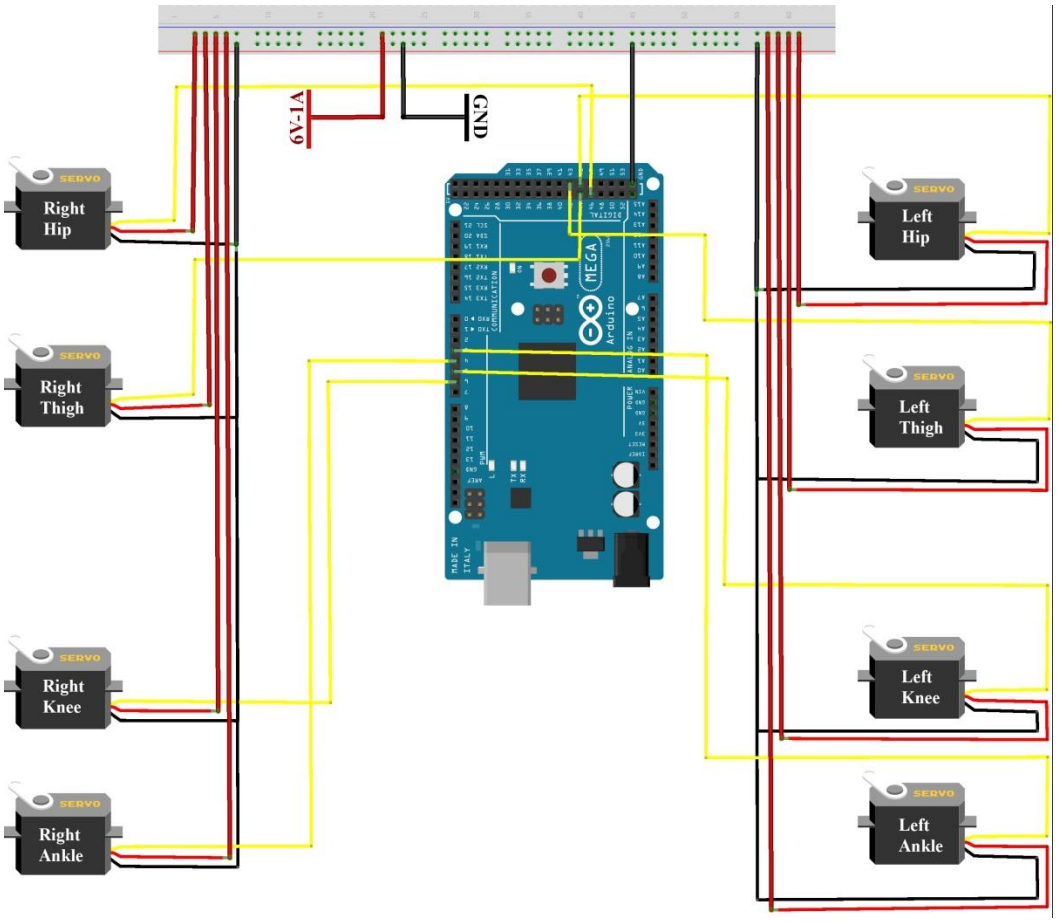


Fig 2.8.1: *Circuit of the Arduino board connected with the Servomotors.*

Servomotor	Right Ankle	Right Knee	Right Thigh	Right Hip	Left Hip	Left Thigh	Left Knee	Left Ankle
Digital Pin	4	6	44	46	45	43	5	3

Table 2.1.1: *The connected digital pins on the Arduino borad to the servomotors.*

2.7 Robot's eye

With the Arduino accounted for, the robot can operate perfectly by just generating a set of angles that will allow it to move forward, backward or in any desired direction. Furthermore, the next feature is also important, since it will allow the robot to see its surrounding and get commands via voice or certain gestures, and even construct a 3d model of the world around it, which can be analyzed later using a personal laptop. This feature is the unique input device that was developed by Microsoft, the Kinect sensor.

2.7.1 Kinect

Kinect is a line of motion sensing input devices by Microsoft for Xbox 360 and Xbox One video game consoles and Windows PCs. Based around a webcam-style add-on peripheral, it enables users to control and interact with their console/computer without the need for a game controller, through a natural user interface using gestures and spoken commands. The first-generation Kinect was first introduced in November 2010 in an attempt to broaden Xbox 360's audience beyond its typical gamer base. A version for Windows was released on February 1, 2012. Kinect competes with several motion controllers on other home consoles, such as Wii Remote Plus for Wii and Wii U, PlayStation Move/PlayStation Eye for PlayStation 3, and PlayStation Camera for PlayStation 4 [9].

Fig 2.9: *The Kinect sensor.*

Microsoft released the Kinect software development kit for Windows 7 on June 16, 2011. This SDK was meant to allow developers to write Kinecting apps in C++/CLI, C#, or Visual Basic .NET.

2.7.2 Kinect technology

The innovative technology behind Kinect is a combination of hardware and software, contained within the Kinect sensor accessory, which can be added to any existing Xbox 360. The Kinect sensor is a flat black box that sits on a small platform, placed on a table or shelf near the television that you are using with your Xbox 360. Newer Xbox 360s have a Kinect port from which the device can draw power, but the Kinect sensor comes with a power supply at no additional charge for users

of older Xbox 360 models. For a video game to use the features of the hardware, it must also use the proprietary layer of Kinect software that enables body and voice recognition from the Kinect sensor. There are three things within the Kinect sensor:

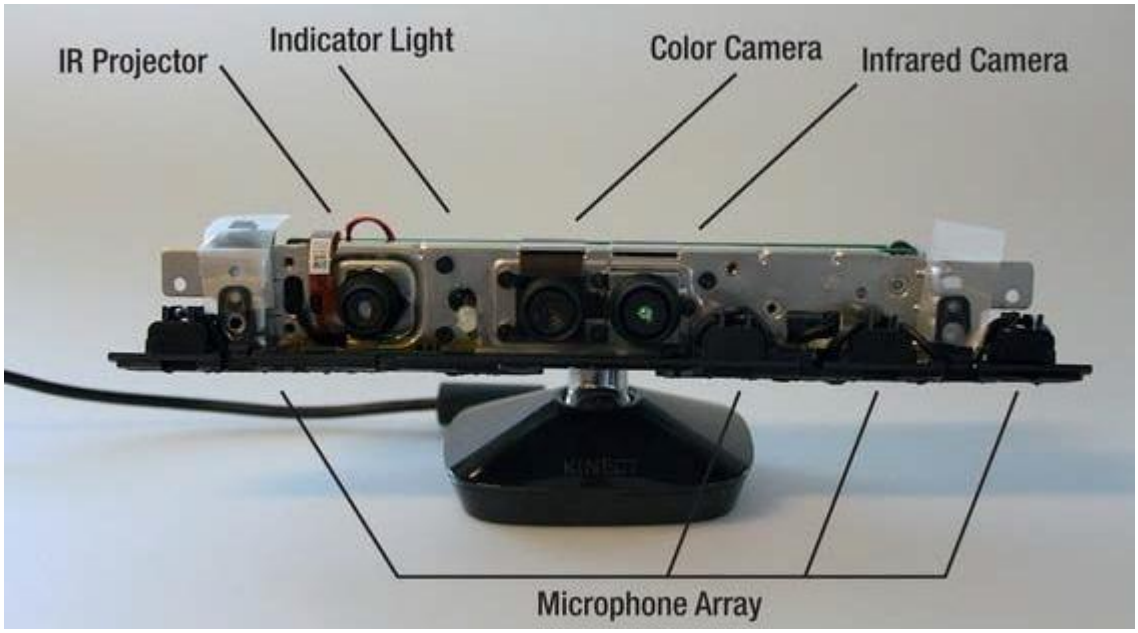


Fig 2.10: *Kinect interior structure [10].*

Color VGA video camera: *This video camera aids in facial recognition and other detection features by detecting three color components: red, green and blue. Microsoft calls this an "RGB camera" referring to the color components it detects.*

Depth sensor: *An infrared projector and a monochrome CMOS (complimentary metal-oxide semiconductor) sensor that work together to "see" the room in 3-D regardless of the lighting conditions.*

Multi-array microphone: *This is an array of four microphones that can isolate the voices of the players from the noise in the room. This allows the player to be few feet away from the microphone and still use voice controls.*

A further look at the technical specifications for Kinect reveals that both the video and depth sensor cameras have a 640 x 480-pixel resolution and run at 30 frames per second. The specifications also suggest that you should allow about 1.2 to 3.5 meters of play space between you and the Kinect sensor, although the sensor can maintain tracking through an extended range of approximately 0.7 to 6 meters.

Kinect's software layer is the essential component to add meaning to what the hardware detects. When you first start up Kinect, it reads the layout of your room and configures the play space you'll be moving in. Then, Kinect detects and tracks 48 points on each player's body, mapping them to a digital reproduction of that player's body shape and skeletal structure, including facial details [10].

2.8 Summary

In this chapter we have explored the necessary parts of the robot:

- The frame of the robot which is made of several aluminum plates that are assembled in a way that offers 8 DOF represented by the following joints: top hip, bottom hip, knee, and ankle, in both legs.

- The servomotors, which are the actuators of this system, are chosen to be the 1501MG high power that offers a torque of 17.7 kg.cm; a torque sufficient enough to allow the leg to generate a gait without getting a false reading.

- The Arduino Mega, which acts as the controller of our system, is a microcontroller that is built with the ATmega1280 microcontroller and have 54 pins, in which 15 of them offer PWM capability.

- Finally, the Kinect that is used by the robot to discover and communicate with the surrounding environment. Either by supplying useful analytical data to the user or by getting command from him.

Chapter III

Kinematic Analysis

3.1 Introduction

In the previous chapter several points where made clear, such as the robot's design, the material used in the modeling of the robot, and the actuators providing the required torque to move the different parts of the robot. Although these points are sufficient to build a biped robot, they are not enough, because the robot needs to move, which is why we are going to analyze the necessary kinematic aspects of bipedalism in this chapter.

The first aspect that will be discussed in this chapter is Forward Kinematics. Forward Kinematics is necessary, because it models the transformation matrices between the different frames that are present within the biped; this is done by using the Denavit-Hartenberg method.
The second aspect is the robot CoG, because it is the key element for the robot's stability; this point is expressed based on the derived forward kinematics tools.

3.2 Forward Kinematics

Forward kinematics is about using kinematic equations on a multi-link robot, be it a robotic arm or biped robot, in order to compute the position of the end-effector, which is on our case the non-supportive foot. These computations are done using known values for the joint parameters [11].
This procedure is needed in several domains such as robotics, computer games, and animation.
The kinematics equations for the series chain of a robot are obtained using a rigid transformation [Z] to characterize the relative movement allowed at each joint, and separate rigid transformation [X] to define the dimensions of each link. The result is a sequence of rigid transformations alternating joint and link transformations from the base of the chain to its end link, which is equated to the specified position for the end link

$$[T] = [Z_1][X_1][Z_2][X_2]\ldots[X_{n-1}][Z_n] \quad (3.1)$$

Where [T] is the transformation locating the end-link. These equations are called the kinematics equations of the serial chain [12].

3.3 Denavit-Hartenberg representation

3.3.1 DH parameters

A robot manipulator consists of several links connected, usually, by single degree of freedom joints, say, a revolute or a prismatic joint. In order to control the end-effector with respect to the base, it is necessary to find the relation between the coordinate frames attached to the end-effector and the base. This can be obtained from the description of the coordinate transformations between the

coordinate frames attached to all the links, and forming the overall description in a recursive manner. For this purpose, the material presented in the previous section for describing the position and orientation of the rigid body is useful for obtaining a composition of coordinate transformations between the consecutive frames. As a first step, a systematic general method is to be derived to define the relative position and orientation of two consecutive links. The problem is to define two frames attached to two successive links and compute the coordinate transformation between them. In general, the frames are arbitrarily chosen as long as they are attached to the link they are referred to. Nevertheless, it is convenient to set some rules for the definition of the link frames. The convention adopted here for a serial chain robot shown in Fig 3.1, is that it has n+1 links, namely, link #0, . . . #n, coupled by n joints, i.e. joint 1, . . . n.

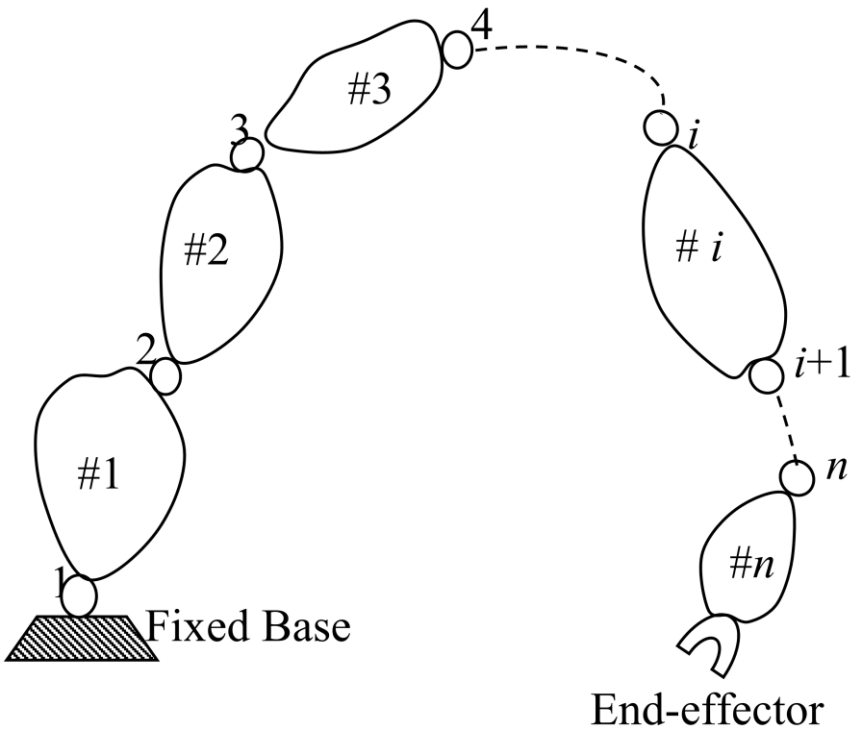


Fig 3.1: *Serial manipulator [13].*

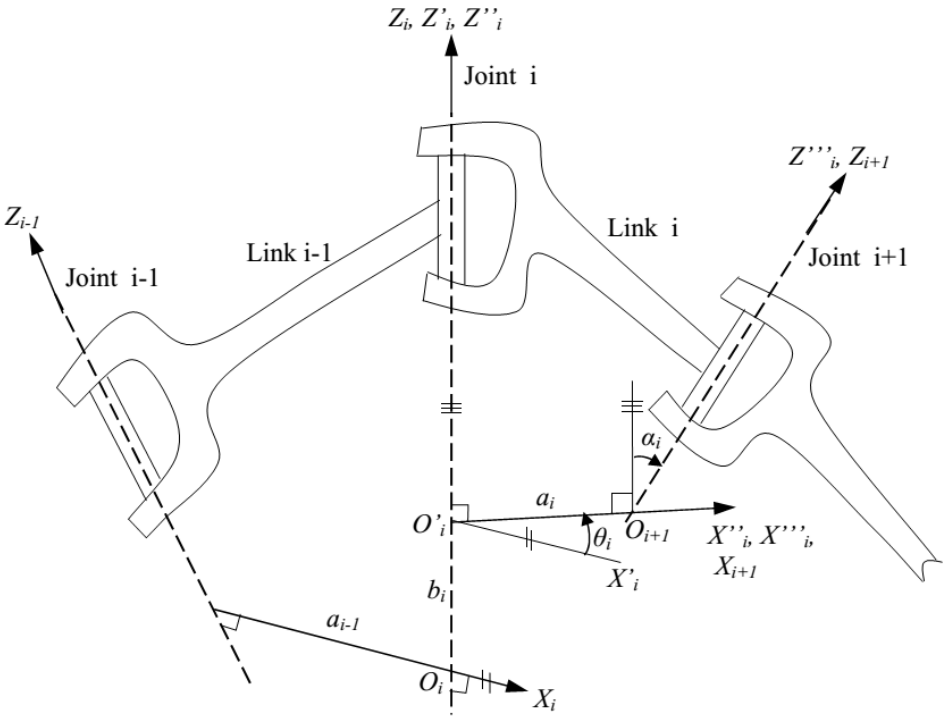
Now, referring to Fig 3.2

a) Let axis i denotes the axis of the joint connecting link i−1 to link i.
b) A coordinate system X_i, Y_i, Z_i is attached to the end of the link i−1 – not to the link i! – for i = 1, . . . n+1.
c) Choose axis Z_i along the axis of joint i, whose positive direction can be taken towards either direction of the axis.
d) Locate the origin, O_i, at the intersection of axis Z_i with the common normal to Z_{i-1} and Z_i. Also, locate O'_i on Z_i at the intersection of the common normal to Z_i and Z_{i+1}.
e) Choose axis X_i along the common normal to axes Z_{i-1} and Z_i with the direction from former to the later.
f) Choose axis Y_i so as to complete a right handed frame.

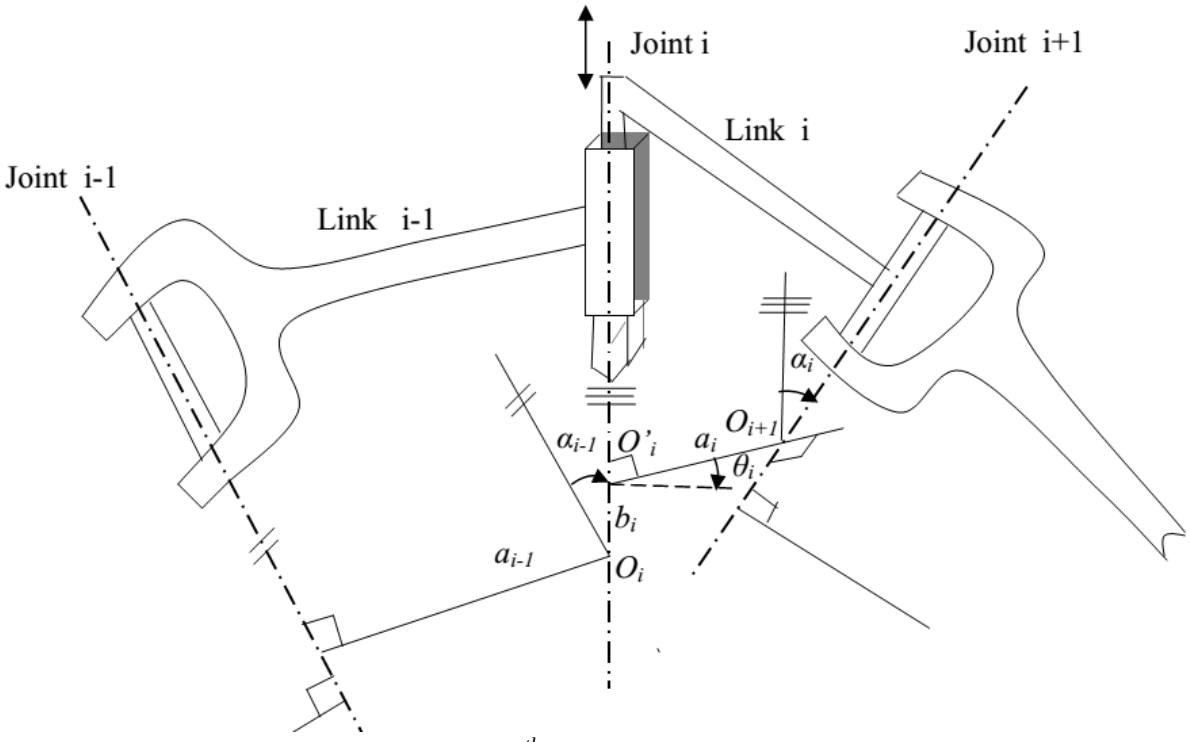
Note that the above conventions do not give a unique definition of the link frames in the following cases:

- For frame 1 that is attached to the fixed base, i.e., link 0, only the direction of axes Z_1 is specified. Then O_1 and X_1 can be chosen arbitrarily.
- For the last frame n+1, the foregoing convention does not apply since there is no link n+1. Thus, frame n+1 can be arbitrarily chosen.
- When two consecutive axes are parallel, the common normal between them is not uniquely defined.
- When two consecutive axes intersect, the direction of X_i is arbitrary.

When joint i is prismatic, only the direction of axis Z_i is determined, whereas the location of O_i is arbitrary.



(a) *The i^{th} joint is revolute*



(b) *The i^{th} joint is prismatic*

Fig 3.2: *Frame convention and Denavit and Hartenberg (DH) parameters [13].*

In all such cases, the indeterminacy can be exploited to simplify the procedure. For instance, the axes of frame n+1 can be made parallel to those of frame n. Once the link frames have been established, the position and orientation of frame i with respect to frame i−1 are completely specified by four parameters known as the Denavit and Hartenberg (DH) parameters. Hence, these frames are also referred as DH frames. The four DH parameters are defined as follows:

a) **b_i (Joint offset),** *The length of the intersections of the common normals on the joint axis Z_i, i.e. O_i and O'_i. It is the relative position of links i−1 and i. This is measured as the distance between X_i and X_{i+1} along Z_i.*

b) **θ_i (Joint angle),** *The angle between the orthogonal projections of the common normals, X_i and X_{i+1}, to a plane normal to the joint axes Z_i. Rotation is positive when it is made counter clockwise. It is the relative angle between links i−1 and i. This is measured as the angle between X_i and X_{i+1} about Z_i.*

c) **a_i (Link length),** *The length between the O'_i and O_{i+1}. This is measured as the distance between the common normals to axes Z_i and Z_{i+1} along X_{i+1}.*

d) **α_i (Twist angle),** *The angle between the orthogonal projections of joint axes, Z_i and Z_{i+1} onto a plane normal to the common normal. This is measured as the angle between the axes, Z_i and Z_{i+1}, about axis X_{i+1} to be taken positive when rotation is made counter clockwise.*

Note that the above four parameters are defined sequentially as one moves from link i–1 to link i+1 through link i. Moreover, the first two parameters, namely, b_i and θ_i, define the relative position of links i−1 and i; whereas the last two, a_i and α_i, describe the size and shape of link i that are always constant. However, parameters b_i and θ_i, are considered variables depending on the type of joints in use.

In particular,

- θ_i is variable if joint i is revolute; and
- b_i is variable if joint i is prismatic.

So, for a given type of joint, i.e. revolute or prismatic, one of the DH parameters is variable, which is called 'joint variable', whereas the other three remaining parameters are constant that are called 'link parameters' [13].

3.3.2 DH Matrix

The DH parameters that were explained represent only the first step of the DH representation. The second step is constructing the DH matrix. The DH matrix is a transformation matrix from one coordinating system to another. It is a homogeneous matrix that has the following form:

$$\begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{(i-1)} \\ \sin\theta_i\cos\alpha_{(i-1)} & \cos\theta_i\cos\alpha_{(i-1)} & -\sin\alpha_{(i-1)} & -\sin\alpha_{(i-1)}d_i \\ \sin\theta_i\sin\alpha_{(i-1)} & \cos\theta_i\sin\alpha_{(i-1)} & \cos\alpha_{(i-1)} & \cos\alpha_{(i-1)}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3.1.1)$$

This matrix is obtained by multiplying the transformation matrices of the following linear transformations:

- Transition of the i^{th} frame along its Z axis with the distance d_i.

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Rotation of the i^{th} frame along its Z axis with the angle θ_i.

$$\begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Transition of the i^{th} frame along the $i\text{-}1^{th}$ frame's X axis with the distance a_{i-1}.

$$\begin{bmatrix} 1 & 0 & \mathbf{0} & a_{(\mathbf{i-1})} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

- Rotation of the i^{th} frame along the $i\text{-}1^{th}$ frame's X axis with the angle α_{i-1}.

$$\begin{bmatrix} 1 & 0 & \mathbf{0} & 0 \\ 0 & \mathbf{cos\alpha_{(i-1)}} & \mathbf{-sin\alpha_{(i-1)}} & 0 \\ 0 & \mathbf{sin\alpha_{(i-1)}} & \mathbf{cos\alpha_{(i-1)}} & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix}$$

The DH matrices are obtained for each two consecutive joints; and, in order to obtain the transformation to get from one frame, say 0, to another, frame 6, then the following DH matrices need to be concatenated in order to do so.

$$^{6}_{0}T = {}^{1}_{0}T.\,{}^{2}_{1}T.\,{}^{3}_{2}T.\,{}^{4}_{3}T.\,{}^{5}_{4}T.\,{}^{6}_{5}T \qquad (3.2)$$

3.4 The robot forward kinematics

Now that we know the theory behind the DH representation, we're going to apply it to our project. First, we are going to get the necessary geometrical properties of the robot, and then use them to obtain the DH parameters. Once that is done, obtaining the DH matrices for all 8 frames is an easy task.

3.4.1 The robot DH representation

As explained before, the DH representation is a useful tool to compute the position of the end factor with respect to the fixed base. For a robotic arm, defining these two elements is easy, but for a biped robot, these elements become quite relative; because it will depend on which foot is going to leave the ground first. Based on this decision, the end-effector and the fixed base are going to change from a biped to another.

In our case, the first foot to leave the ground is the left one. Since the left foot is going to leave first, then it means that the right foot is going to remain on contact with the floor, supporting the robot's weight, which is why it is called the supporting foot; and, it is also going to be considered the fixed base in our analytical study, which means that the left foot is the non-supporting foot, thus, it's going to be the end-factor for our analysis.

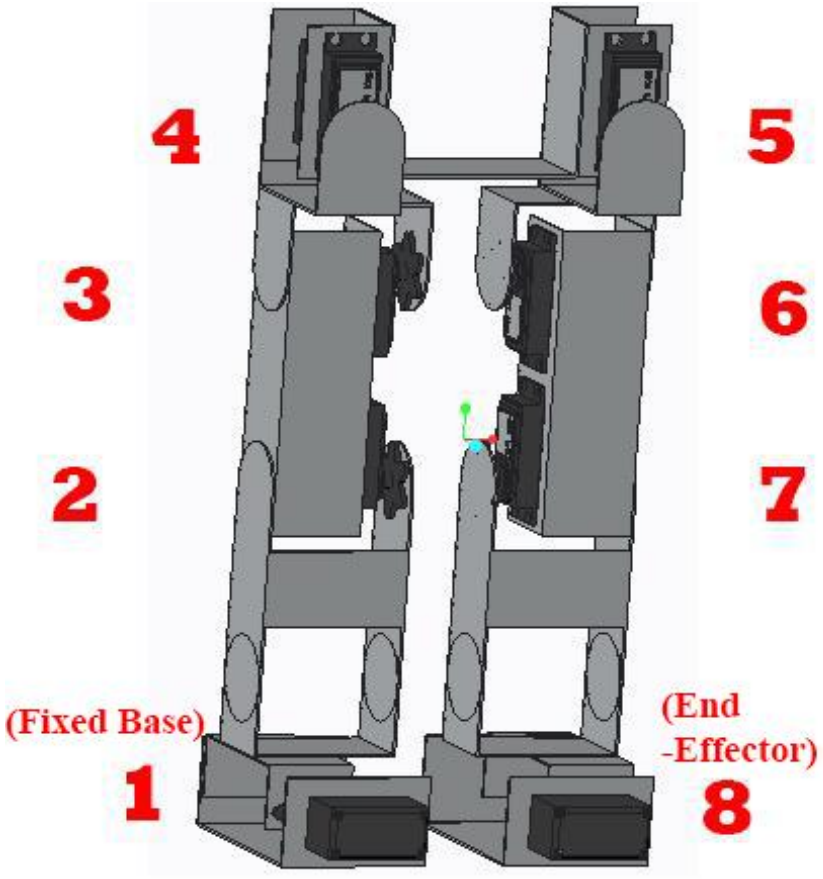


Fig 3.3: *Biped CAD with the joint number indicated.*

Taking this observation into account, we assigned a frame for each joint within our design. In order to assign a coordinating frame, one must know how to position the three axes of it. The first axis to be assigned is the z-axis, since it is the easiest of them all. The z-axis at any given joint point is the servo-motor's axis of rotation. The x axis is a little bit tricky, it goes along the line that link the current joint with the next one; but, for the end factor, there's no next joint so it points to the same direction as the previous x-axis. The y axis is then obtained using the right hand rule.
By applying these rules, we assigned frames to each joint in our robot, which make a total of 8 frames, considering that we have 8 servo motors. Fig 3.4 represents the result of this exercise.

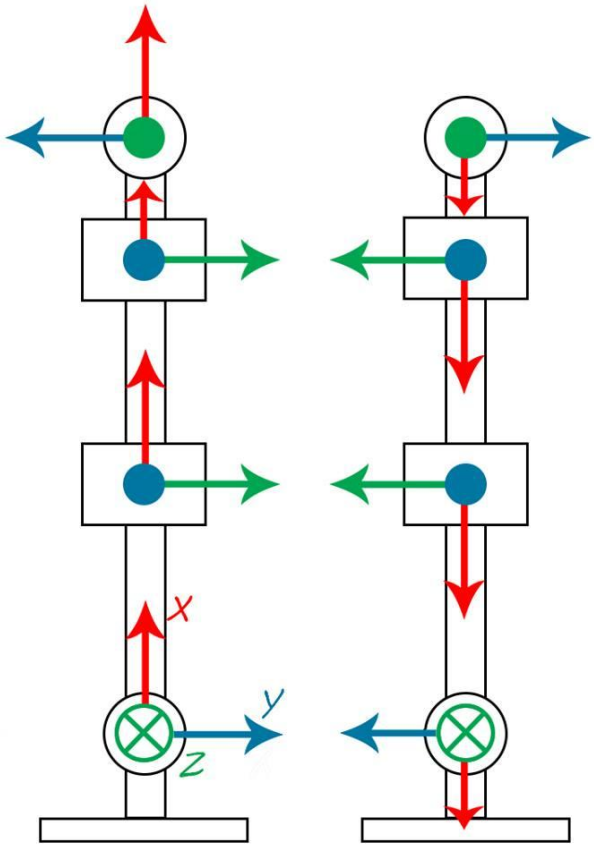


Fig 3.4: *Robot's frames demonstration.*

3.4.2 Robot link parameter

After assigning a frame for every joint that is present in the biped robot, then we need to define some geometrical parameters, which are required for the forward kinematic analysis. These parameters are obtained using the CAD model that was realized in the PTC Creo software. The PTC Creo provides distance measuring tools, which can be found within the analysis submenu.

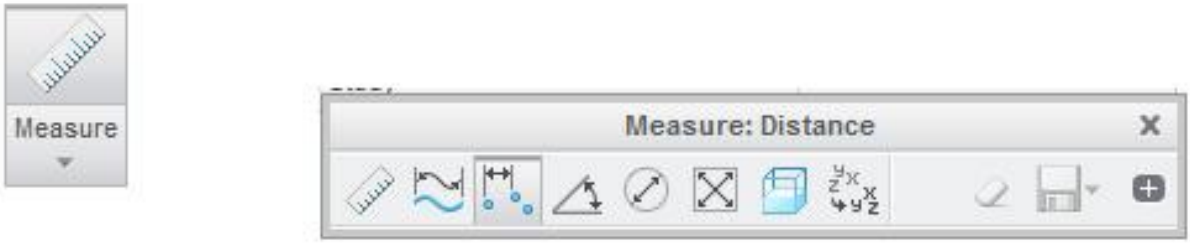


Fig 3.5: *Measure tool and Measure window from PTC.*

Parameters	Interpretation	Value[mm]
a_1	The link connecting the right ankle with right knee	124.35
a_2	The link connecting right knee with the lower right hip joint	82.3
a_3	The link connecting the lower right hip joint with the upper one	59.35
b_3	The Waist length	117
a_4	The link connecting the upper left hip joint with the lower one	59.35
a_5	The link connecting the lower left hip joint with the left knee	82.3
a_6	The link connecting the left knee with the left ankle	124.35

Table 3.1: *Distance between all possible joints.*

i	a_{i-1}	α_{i-1}	b_i	θ_i
1	0	0	0	0
2	a_1	-90	0	0
3	a_2	0	0	0
4	a_3	-90	0	0
5	0	0	b_3	180
6	a_4	-90	0	0
7	a_5	0	0	0
8	a_6	-90	0	0

Table 3.2: *DH parameters that relate the 8 frames.*

Now that the necessary DH parameters are specified, the different transformation matrices can be easily obtained. Using these matrices, the position of the end effector position can be computed by multiplying all of them in the order that was explained before.

3.4.3 Transformation matrices

What follow is a list of all the obtained transformation matrices.

$$ {}^{0}_{1}T = \begin{bmatrix} cos\theta_1 & -sin\theta_1 & 0 & 0 \\ sin\theta_1 & cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{1}_{2}T = \begin{bmatrix} cos\theta_2 & -sin\theta_2 & 0 & a_1 \\ 0 & 0 & -1 & 0 \\ sin\theta_2 & cos\theta_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{2}_{3}T = \begin{bmatrix} cos\theta_3 & -sin\theta_3 & 0 & a_2 \\ sin\theta_3 & cos\theta_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}_{4}T = \begin{bmatrix} cos\theta_4 & -sin\theta_4 & 0 & a_3 \\ 0 & 0 & 1 & 0 \\ -sin\theta_4 & -cos\theta_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{4}_{5}T = \begin{bmatrix} cos\theta_5 & -sin\theta_5 & 0 & 0 \\ -sin\theta_5 & -cos\theta_5 & 0 & 0 \\ 0 & 0 & -1 & -d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{5}_{6}T = \begin{bmatrix} cos\theta_6 & -sin\theta_6 & 0 & a_4 \\ 0 & 0 & -1 & 0 \\ sin\theta_6 & cos\theta_6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{6}_{7}T = \begin{bmatrix} cos\theta_7 & -sin\theta_7 & 0 & a_5 \\ sin\theta_7 & cos\theta_7 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{7}_{8}T = \begin{bmatrix} cos\theta_8 & -sin\theta_8 & 0 & a_6 \\ 0 & 0 & 1 & 0 \\ -sin\theta_8 & -cos\theta_8 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

These matrices depict only the necessary linear transformations to get from a frame to another, but this is not enough, because of the physical limitations of the robot. These physical limitations are due to the fact that the robot is a multi-link body; so, it is necessary to keep the rotation angles of each link under a certain constraint, in order to avoid collision. Based on the robot model shown in Fig 2.5 the following limitations are specified.

θ_1 θ_4 θ_5 θ_8: [-45, 45]

θ_2: [-30, 30]

θ_3 θ_6 θ_7: [-90, 90]

Under these constraints, and using the transformation matrices shown above, one can obtain the (x, y, z) coordinates of a point in all the frames, just by knowing its coordinate in one of them.
For an example, suppose a point B, with its coordinates at frame0 given as follow $B_0(x_0, y_0, z_0)$. To get its coordinates in 8th frame depicted as $B_8(x_8, y_8, z_8)$ multiply the vector $[x_0\ y_0\ z_0\ 1]$ by the transformation matrix ${}^{8}_{0}T$.

$$ {}^{8}_{0}T = {}^{1}_{0}T\,{}^{2}_{1}T\,{}^{3}_{2}T\,{}^{4}_{3}T\,{}^{5}_{4}T\,{}^{6}_{5}T\,{}^{7}_{6}T\,{}^{8}_{7}T \qquad (3.3) $$

The obtained result is a vector with coordinates of point B in the 8th frame padded with a one $[x_8\ y_8\ z_8\ 1]$.

3.5 CoG calculations

The CoG is an important parameter for any static walking robot. The reason for this is that it will help stabilize the robot during its two walking phases (DSP, SSP). The projection of the CoG on the ground is called the CoP. The CoP of the biped robot needs to be at all the time within the supporting surface.

To obtain the CoG of the robot, the CoG of the different parts that compose the robot is computed with respect to a single frame; in this case frame 0 is used. The CoG of the entire assembly is then computed by multiplying the ratio $\frac{M_{part}}{M_{total}}$ of each part by its CoG position vector and then adding them together.

$$\overrightarrow{P_{com}} = \frac{1}{M_{total}} \sum_{i=0}^{7} \overrightarrow{P_{com\,i}}.M_{parti} \tag{3.4}$$

the CoG of each part is computed by integrating the density function of the part multiplied by the volume, with respect to the 3 axes x, y and z. Since most of the plates are made out of aluminum, then this means that the density function is constant. For simplicity, we will consider the density function ρ of the motors to be constant as well.

$$\overrightarrow{P_{com\,i}} = \frac{1}{M_{parti}} \iiint_{x,y,z} \rho(x,y,z).xdx.ydy.zdz \tag{3.5}$$

The following table contains all the parameters that are needed to analyze the weight of our robot.

Part	Density (g/mm3)	Volume (mm3)	Weight (g)
Servo-motor	1.853024e-03	32379.5	60
Part A	2.7e-03	5465.65	14.76
Part B	2.7e-03	4882.11	13.18
Part C	2.7e-03	10931.3	29.51
Part D	2.7e-03	8465.65	22.86
Part F	2.7e-03	3955	10.68
Part G	2.7e-03	8884.23	23.99
Part H	2.7e-03	7000	18.9

Table 3.3: *Physical parameters of the different parts.*

The above table supplies enough information to get the robot CoG using the PTC Creo software. Within the PTC Creo each part is opened separately, and then the density distribution function and the density of each part are entered in the mass property of the analysis submenu. As shown in fig 3.6.

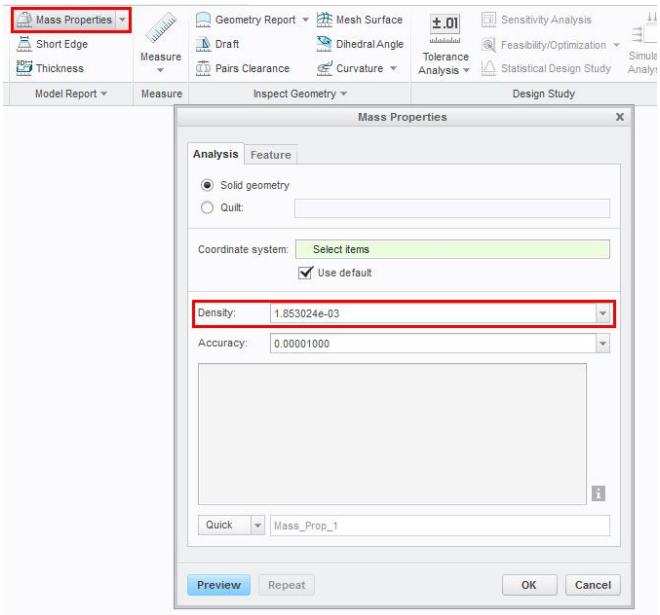


Fig 3.6: *Part mass properties window with the specific density value.*

After entering the appropriate data for each part, the assembly file gets updated. This means that the mass properties of all the parts will be shown in the assembly's mass property, and by using the preview feature, the robot's CoG is previewed, as shown in fig 3.7, along with the necessary data which are shown in table 3.4.

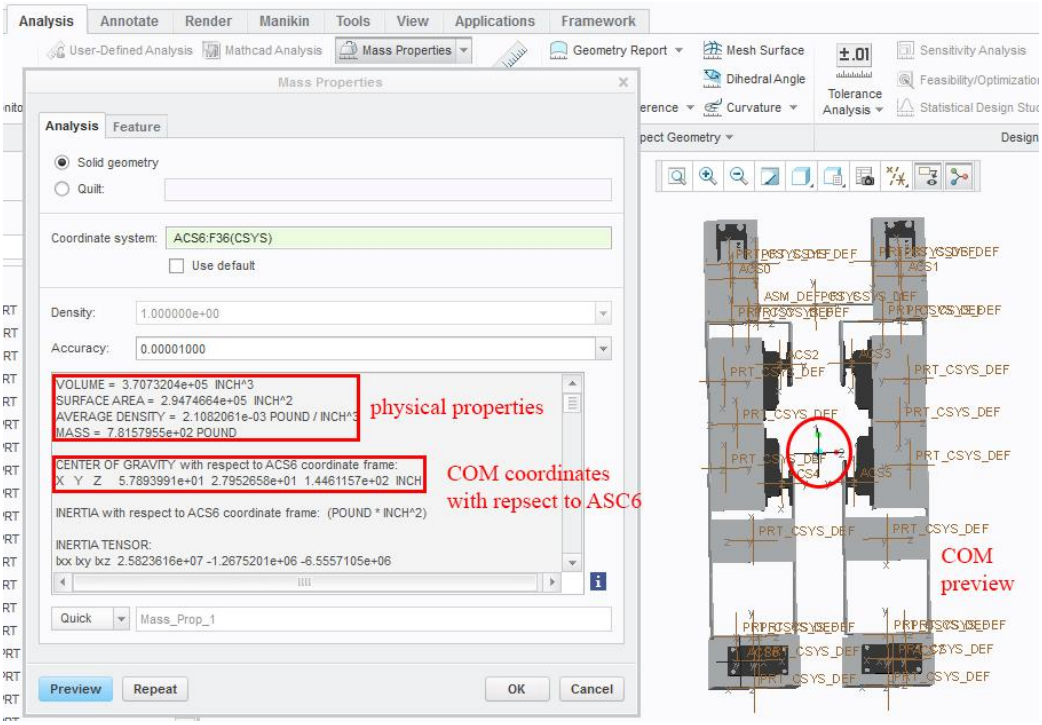


Fig 3.7: *Result of running the PTC CoG preview.*

frame	CoG coordinate			frame	CoG coordinate		
	x	y	z		x	y	z
frame 1	144.61	57.89	27.95	frame 5	121.29	58.36	20.45
frame 2	20.11	13.05	24.86	frame 6	62.20	13.05	24.86
frame 3	-62.20	13.05	24.86	frame 7	-20.11	13.05	24.86
frame 4	-121.29	-58.36	-20.45	frame 8	-144.61	-57.89	-27.95

Table 3.4: *COM coordinates with respect to all 8 frames.*

3.6 Summary

This chapter described the kinematic aspect necessary for analyzing the robot movements. The first aspect was the forward kinematics, which is used to get the end-effector coordinates with respect to the base, which is the supporting foot in our case. Coordinating frames were assigned at each joint; and since the robot contains 8 joints, the assignment task was started from the right ankle to the left one. Once the frames are assigned, we can assign the necessary transformation matrices (called the DH matrices), by using the corresponding DH parameters in from (3.1.1).

The second aspect is the robot's CoP. The CoP of any object is just the projection of its CoG on the ground plane. The task of obtaining the robot CoG is carried out in two steps. First, the CoG of each part is obtain by using equation (3.4), and once this is done, the total CoG is found by applying equation (3.5).

Chapter IV

Motion, Control and Commands

4.1 Introduction

In the previous chapter, the robot's required dynamic analyses were covered. These analyses included the forward kinematics study, which was covered using the parameters and transformation matrices that were computed by the Denavit and Hartenberg method. The coordinates of the robot's CoG was also obtained using the PTC platform.

In this chapter, the fourth and last chapter of this report, all the elements that have been explained so far will be put together, in order to allow the robot to walk forward in a certain way. This implies choosing one of the two methods of stabilizing a biped robot during its walk, either statically or dynamically. Based on the chosen stability constraint, a control system is designed, so that the robot can move while satisfying this constraint.
In addition to the stability issue and the design of the control system, the robot needs to communicate with its user, this means adding the necessary and appropriate hardware and software to receive commands from the user.

4.2 Robot's walk

The biped walking process, or biped locomotion area, has been studied for a long time, but it is only in the past years, thanks to the fast development of computers, that real robots started to walk on two legs. Since then the problem has been tackled from different directions.
First, there were robots that used static walking. The control criterion was to maintain the projection of the CoG on the ground, inside the foot's support area.nFor the dynamic walking robots, the CoG can be outside of the support area, but the ZMP cannot. The ZMP criterion has been broadly used to generate biped control algorithms.

For this implementation, the stability of the robot during the walking phases is kept by applying the first approach, which is static walk.

4.2.1 Static walk

Static walking assumes that the robot is statically stable. This means that at any time, if all motion is stopped, the robot will stay indefinitely in a stable (balanced) position. It is necessary that the projection of the CoG of the robot on the ground must be contained within the foot support area (Fig 4.1).
The support area is either the foot surface, in case of one supporting leg, or the minimum convex area containing both foot surfaces, in case of both feet are on the ground.

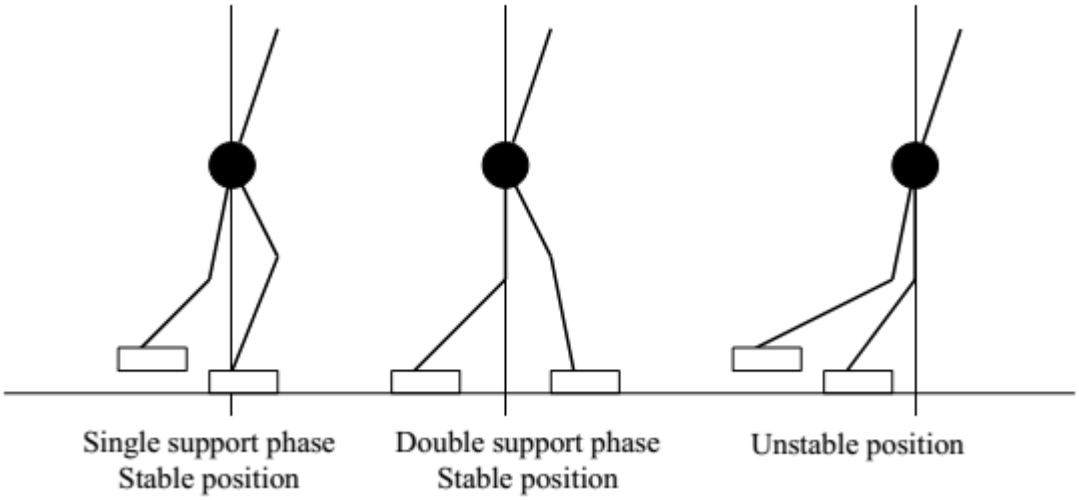


Fig 4.1: *Static walk.*

These are referred to as SSP and DSP, respectively. Also, the walking speed must be low, so that inertial forces are negligible.
This kind of walking requires large feet, strong ankle joints, and can achieve only slow walking speeds.

4.2.2 The robot gait

In order to generate the robot gait using a static walk, the constraint of keeping the robot CoG within the supporting area, as shown in Fig 4.1, during both phases: SSP and DSP, must be achieved.

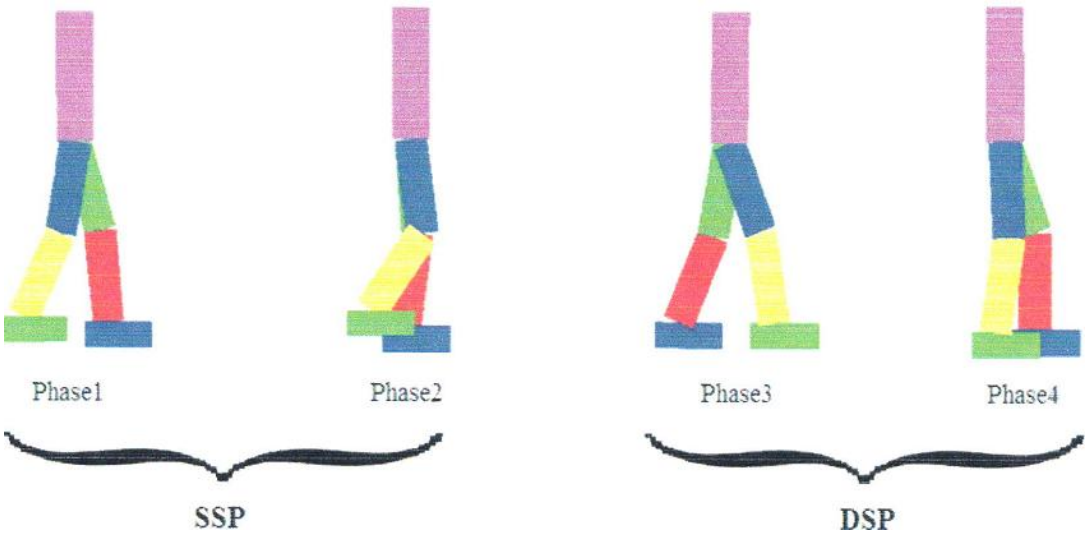


Fig 4.2: *The robot walking phases.*

During DSP, the task of keeping the CoG within the support area is easy. The reason for this is that the support area covers both feet, along with the polygon that links them, as shown in Fig 4.3. However in SSP, the support area is reduced to the surface of the single foot that is still in contact with the ground.

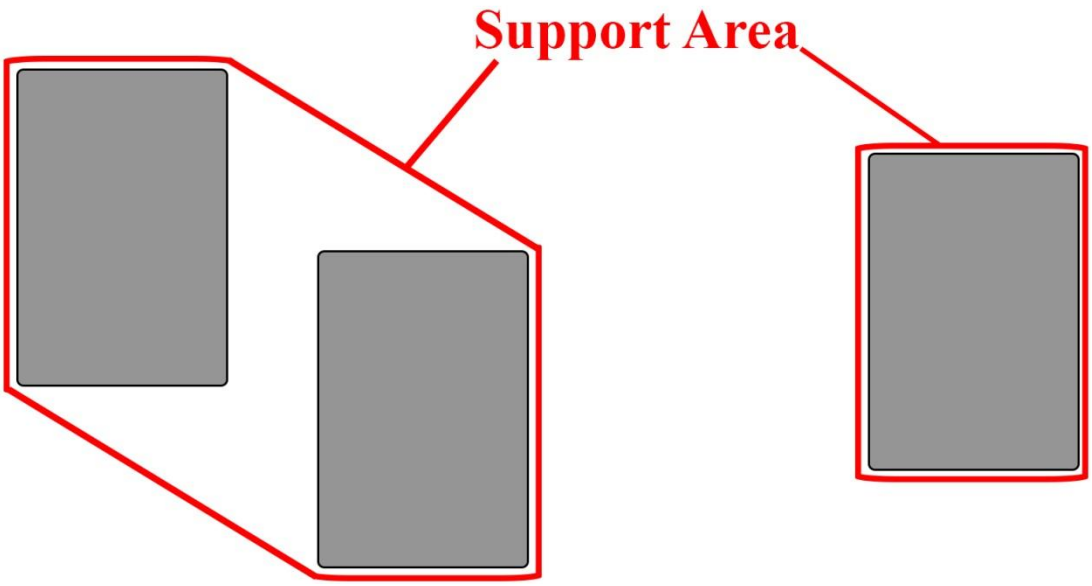


Fig 4.3: *Support area during the DSP and SSP.*

To solve this issue, the robot needs to stabilize itself during the DSP. To do that, it needs to lean left or right with a specific angle θ_r , so that the CoG is within the surface covered by the foot that will remain in contact with the ground during the next phase, which is the SSP.
Using the CoG coordinates, obtained in chapter 3, to do a numerical application of equation (4.1), θ_r is evaluated with 21.83°.

$$\theta_r = tan^{-1}\left(\frac{-Y_{CoG1}}{X_{CoG1}}\right) \quad (4.1)$$

Now that the stability has been taken care of, the robot needs to walk forward when it is ordered to do so. The robot will walk in a way that will allow it to stop always at the initial position, which is also defined to be the default position.

Starting from the initial position, the robot leans to the right, and then move the left foot forward. Once the left foot is in contact with the ground, the robot now leans to the left and brings the right foot next to the left one, returning to the initial position. This sequence defines a single step of the robot. By iterating this single step several times, the robot walks for any given distance in a straight line.

The step of the implemented robot is designed in a proportional way with respect to that of the ASIMO robot. Table 4.1 shows 3 points with their coordinates with respect to frame 0. These points are used to generate a step.

Point	x (mm)	y (mm)	z (mm)
point 1	0	115	0
point 2	25	115	-40
point 3	0	115	-80

Table 4.1: *Foot path for a single step.*

These points can be used only to specify the foot's trajectory by any means of interpolations. A good example of that would be the simplest one, which is Lagrange interpolation; but this would be pointless, because what is required is the sequence of the required joint angles that would get the end-effector to get to these points. This task is carried out by applying inverse kinematics.

4.2.3 Inverse kinematics

Inverse kinematics is doing the opposite task of forward kinematics. Basically, this means that inverse kinematics is about using kinematic equations on a multi-link robot, whether it is a robotic arm or a biped robot, to compute the required joint angles, given a certain position of the end-effector.

Carrying this task is very hard; and, there have been several attempts to solve this kind of problem. The way we approached this problem is by applying successive use of forward kinematics applications with the help of the computer's computational power.

Using the Matlab software, four nested loops are used to generate all possible combinations of θ_2, θ_3, θ_6 and θ_7. The reason for considering only these angles is that the other four, which are θ_1, θ_4, θ_5 and θ_8, are already fixed by the first constraint of static walk, which is keeping the CoG at the supporting area as seen previously.

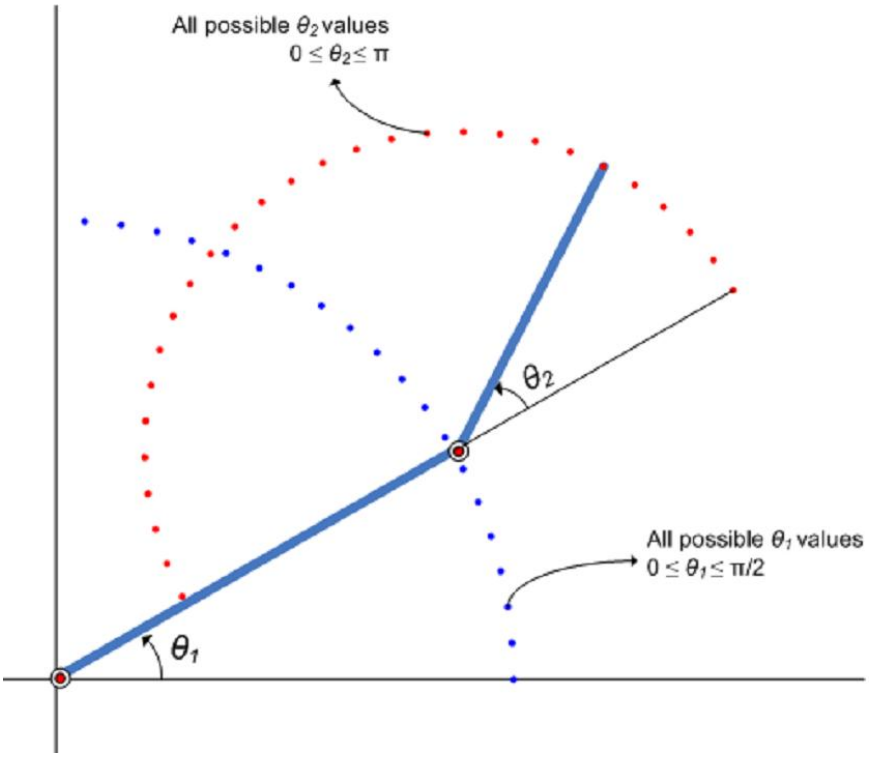


Fig 4.4: *Data generation by Matlab.*

So, after each iteration, the program uses forward kinematics to compute the end-effector position, and checks for the following requirements:

- Is the end-effector is at the required position.
- Does the CoG still hold?

If any combination happens to not meet these requirements, the program stops and returns it.
Applying this technique on the given implementation and for the points specified at table 4.1, yielded to the following set of angles.

Point	θ_2	θ_3	θ_6	θ_7
Point_2	-10	50	0	0
Point_3	-15	-50	10	5

Table 4.2: *Result of applying inverse kinematics.*

4.3 Control system of the robot

Using what we have seen so far, a control system that will allow the biped robot to walk in a static way indefinitely is built. This control system is an open loop control system, without any kind of feedback as shown in Fig 4.6.

Essentially, a control system with feedback loop is better than open loop one; and, since the 1501MG servomotor by HD Power offers digital read of its orientation, then it would be helpful to use those digital reads as a feedback from the different joints. Also, it will help more if we add a gyroscope to each joint to get its orientation. However, even if the orientation is given, the computational time of the inverse kinematic problem is too high, even for the i7 CPU, let alone the Arduino Mega CPU, which is the ATmega1280.

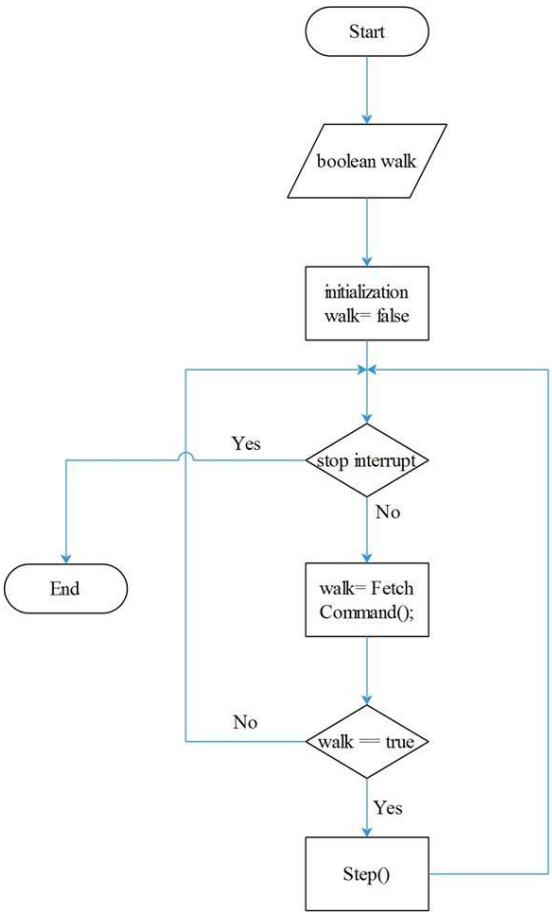


Fig 4.5: *Flowchart of the robot program.*

Fig 4.6: *Block diagram for static walk.*

As demonstrated in Fig 4.5, the robot walks by generating a sequence of steps. These steps start with the initial stance and end with the initial stance, the only thing that will change is the robot's position, which will change by 80 mm along the Z axis of the 0^{th} frame.
Before initiating a step, the robot checks the Boolean variable "walk", if it is true, then the robot proceeds with the step, if it is not, then it stops moving. Initialization of this Boolean is set to false, and using one of the two commanding methods, it will change to either true or false accordingly.

4.4 Receiving commands

The controller of our system, the Arduino Mega, acquires orders using the Bluetooth technology. All that is required is to use Bluetooth module like the one shown in Fig 4.7, and write a simple code to connect the information that is received with the Arduino's serial communication port.

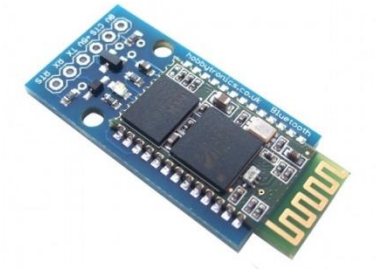


Fig 4.7: *Bluetooth module.*

Now that the Arduino has a Bluetooth module, it can be either commanded by a smart phone or a personal laptop. For smartphone, a small mobile application is used.
The application has a toggle button; the state of the toggle button will decide the state of the robot once the smartphone is connected with the Arduino. Fig 4.8 shows a screenshot of the mobile application developed for smartphones that runs Android Operating Systems.

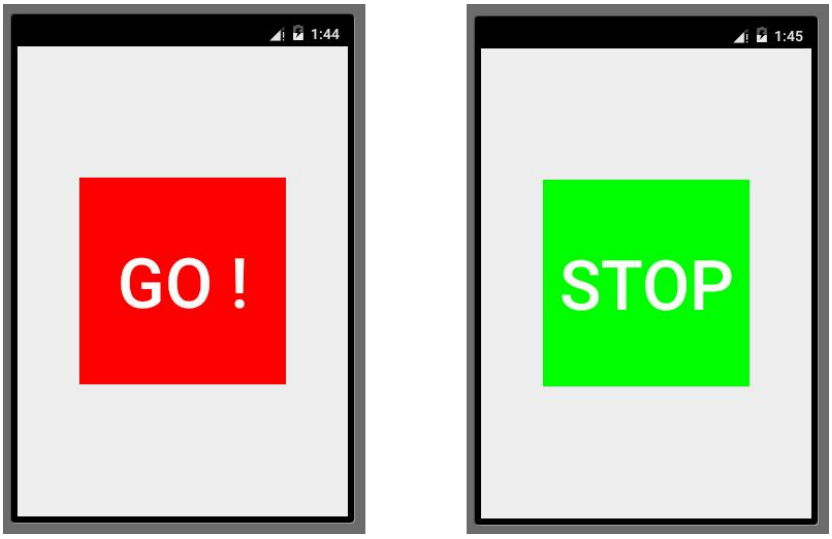


Fig 4.8: *Android commanding mobile app.*

The laptop uses the Kinect sensors to receive commands from the user, which are either gesture commands or voice commands. The Kinect SDK contains an already made code that do this task, all is needed, is to link this commands with the appropriate robot's state, and then send them to the arduino via Bluetooth. Table 4.3 shows the appropriate interpretation of the recognized gesture or speech, while Fig 4.9 shows a screenshot of executing the code contained within the SDK.

Gesture	interpretation	vocal Command	interpretation
thumb up	go to walking state	"Go"	go to walking state
palm	stop walking	"Stop"	stop walking

Table 4.3: *Possible command for the user.*

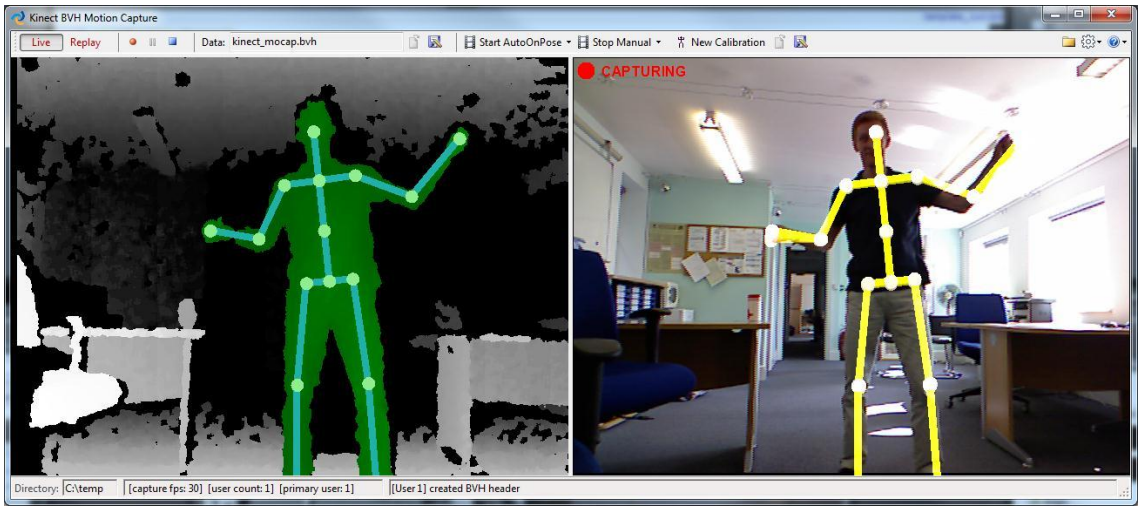


Fig 4.9: *Laptop commanding application.*

4.5 Summary

Using the model described in the 2nd chapter, and by applying the dynamical requirements that were derived in chapter 3, the implemented robot is made to walk using the static walk constraints.
There are several walking behaviors for legged robots, but the static walk is the simplest among them, because it requires a single constraint, which is keeping the robot's CoG within the support area during both walking phases, DSP and SSP. This can be easily satisfied, given that we already have the CoG coordinates from chapter 3. Using the transformation matrices obtained in chapter 3, the sequence of the required joint angles that will allow the robot to move forward is obtained.

The robot uses a simple flowchart to operate. First, it checks for command signals and modifies the walking state based on the received command; if the walking state is walk, then the robot will walk statically, using an open loop control system, where the controller gives a predefined sequence of the angles to the eight present actuators.

Conclusion

In this project we explored the future of robots locomotion, which is using human like leg to make a robot walks. However, the project is still in its early stages, since the biped can only walk on flat surfaces.
To implement this biped robot, we have done the conception, analysis and the implementation of the robot. At the conception phase, a model of how the legs would look like was made. The built model was made based on anatomical analysis of the human leg. This is why the biped contained 8 DOF, expressed by the Hip, Thigh, Knee and Ankle, at each leg of the biped robot.
Once the concept was achieved, it was then designed digitally in the PTC software, which is an open-source CAD software, and using the aluminum plates, we modeled the virtual design into a real one.

We have learned that the most important tasks during the analysis phase were to find the robot's DH parameters and the robot's CoG. Using the CAD of our robot, finding the DH parameters became an easy task. DH parameters are important, since they are used in both forward kinematics and inverse kinematics problems. Finding the DH parameters meant finding the transformation metrics that relate each frame, depicted by a single joint, with the one that follow it. The CoG was automatically calculated in the CAD after supplying the density of each part.
The final stage was the implementation of the robot, which was about making the model that was built previously walk in a certain way; that certain way is the static walk method. Static walk is a simple method of locomotion for legged robots, and the only thing that it requires is to keep the robot statically balance at all times. Using the CoG coordinates, a set of joints' angles were fixed so that the CoP remains within the supporting area of the robot during the entire walk.
The other joints are used to issue the forward movement, by exerting a predefined sequence of angles. This sequence of angles was obtained by mimicking the ASIMO robot's behavior and applying the inverse kinematics.

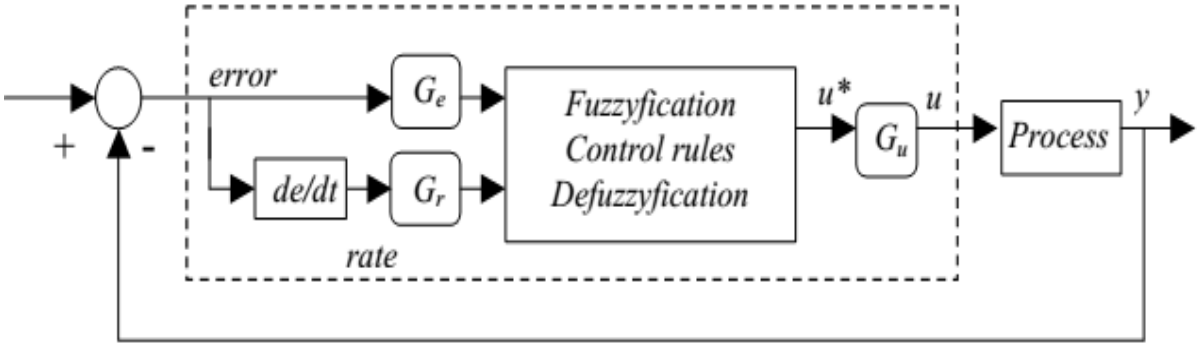
Our contribution was to use to additional servomotors at the hip, to help the robot lean to the sides and maintain its balance. Also, we have used the mobile application and Kinect sensor to control the robot from distance, and make it somehow aware of the environment around it.

We conclude that, although the robot did not exert human like motion, it is considered as a good building block for further work, because it is a Dany-Walker type of robots, and this type of robots can be upgraded to walk dynamically.

Further Work

In a dynamic walking, the center of gravity can be outside of the support area of the robot, but the Zero Momentum Point (ZMP) cannot be. The ZMP is the point where the total angular momentum is zero. The position of the ZMP is computed by finding the point's coordinates (X, Y, Z) where the total torque is zero. The ZMP can be obtained easily using pressure sensors beneath the feet and arranging them in certain geometry. So, with the computed ZMP, the robot must keep it within the support area, for this task several controllers were used, and the best one so far is the fuzzy logic controller.

A good version of the fuzzy logic is the Fuzzy PD incremental controller, since it was already used in many biped robots so far.



In addition to making the robot walk dynamically, several enhancements can be made, such as finding an efficient way to compute the inverse kinematics, allowing the robot's controller to compute it on its own without requiring guidance from a computer; or adding object recognition and depth calculation to it using the Kinect sensor, so that it can go and fetch an object that would be required by the user.

References

[1] Kamphausen, Jörg; *Zweibeiniges Gehen, Seminar Roboter im Alltag*, Department of computer sciences, University of Dortmund, Germany, 2003.

[2] Dienelt, Martin; *Neuere Entwicklungen auf dem Gebiet humanoider Roboter, Wie humanoide Roboter laufen*; Department of Robotics, University of München, Germany, 2003.

[3] L.Kun, Andrew; *A sensory-based adaptive walking control algorithm for variable speed biped robot gaits*, University of New Hampshire, England 1997.

[4] Honda Motor Co; *Humanoid Robot ASIMO; http://asimo.honda.com/*; 2003.

[5] Massachusetts Institute of Technology (MIT); *Leg laboratory; http://www.ai.mit.edu/projects/leglab/*; 2003.

[6] Serway, R. A. and Jewett, Jr. J. W.; *Physics for Scientists and Engineers. 6th Ed*, Brooks Cole, 2003.

[7] Power HD Archive; *Power HD analog servo HD-1501MG datasheet*.

[8] Arduino Official Site; *Arduino Mega overview; http://www.arduino.cc/en/Main/arduinoBoardMega*; 2015.

[9] Microsoft; *Project Natal 101*; Annual E3 press conference, june 1 2009.

[10] Jarrett Webb, James Ashley; *Beginning Kinect Programming with the Microsoft Kinect SDK*, Apress, 2012.

[11] Paul, Richard; *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*, MIT Press, Cambridge, 1981.

[12] J. M. McCarthy; *Introduction to Theoretical Kinematics*, MIT Press, Cambridge, 1990.

[13] S.K. Saha; *Introduction to Robotics*, Tata McGraw-Hill, New Delhi, 2008.