The Design and Construction of an Anthropomorphic Humanoid Service Robot

A thesis presented in partial fulfilment of the requirements for the degree of

Master of Engineering
in
Mechatronics

at
Massey University,
Palmerston North, New Zealand

Peter William Edward Barlow
2011

1 Abstract

This thesis presents the research, mechanical design and construction of the lower half of a biped robot. In the long run this work will be developed further to build a service robot to perform repetitive (and often dangerous) tasks and to help disabled people to carry out everyday tasks.

The aim of this research project is to develop a humanoid with the agility of a 'high end' robot but on a very low budget in comparison. In order to achieve this, several unique mechanical attributes have been proposed and implemented such as dual push rod actuated joint articulation. This technique produces a larger joint torque and reduces leg inertia allowing for the implementation of WCK serial controlled servo modules for actuation. To further increase human-like similarities a toe joint is implemented. This gives the humanoid the ability to stride more elegantly, increase speed control, and reduce energy used for each step.

All the parts of this robot have been manufactured from scratch and most have been CNC machined. Solidworks is used as a 3D modelling package to produce a simulated version of the humanoid to determine dimensions and dynamics before construction takes place. SolidCAM is a computer aided machining package which was used to specify machining paths to produce G-Code. An additional 4^{th} axis was added to the CNC machine solely for the purpose of this project as many parts were too intricate and complex for the standard 3 axis machine.

A biped humanoid requires several types of sensors for balancing. High accuracy and resolution is of paramount importance for the successful control of the humanoid. Various force sensors are reviewed and their advantages and disadvantages are discussed. Gyroscopes and attitude heading reference systems (AHRS) are investigated and tests are performed on all units to obtain operational characteristics and accuracy.

Visual Basic.net has been used for developing software for controlling and monitoring all sensors and actuator modules. Essentially a humanoid platform has been developed with appropriate software allowing for the next stage of development, the development of the gait control algorithms.

2 Acknowledgments

I would like to acknowledge and express my gratitude to several individuals who provided generous amounts of support and cooperation during this project. Thank you to my supervisor, Gourab Sen Gupta who provided a vast range of knowledge and support regarding control systems and sensors implemented in this project, also his support regarding conferences and article publication. Thank you to Clive Bardell for his precise and extremely fast production of parts, Greig McLeay for his extensive knowledge and help with designing and producing a number of the parts and also for teaching me how to use the CNC machine. Lastly I would like to thank a fellow student, Sanjay David, for his help in research, implementation and testing of various sensors in this project.

Contents

List of Figures

List of Tables

1. Introduction

A humanoid, also known as anthropomorphic or biped robot is a robot that mimics the flexibility and characteristics of a human-being. Humans have a long-standing desire to replicate themselves in mechanical form and due to this researchers believe that the humanoid industry will be a leading industry in the twenty first century.[28] Humanoids can not only be used as a companion but also can be implemented into tedious and dangerous situations such as rescue operations or bomb disposing. With huge advances in technologies such as high speed micro processors, low cost high torque servo motors, accurate pressure sensors, vision and motion simulation, and advanced control systems, the reality of humanoids becoming common in everyday life is being brought closer and closer.

This project focuses on designing and building a biped humanoid robot to be used as a service robot, concentrating on the lower body, the legs, and the hips, making it possible for the robot to stand and walk. In the long run this robot will be used to help disabled people perform daily tasks that they would otherwise have trouble with.

At present there are two main areas of production, very high end humanoids, and very low end humanoids. There are very few middle range humanoids. The reason for this is, at present, using a classic humanoid joint design, joint actuators are hugely expensive [38]. To achieve a middle price range product with the flexibility and performance of higher price range products, new and revolutionary joint actuation must be implemented.

1.1. Research aim

The aim of this research is to understand the structure and development of current humanoid robots in order to design and develop a low-cost robot with high-end flexibility. This robot structure will be used as a base for the long term development of a humanoid service robot.

In order to obtain the necessary flexibility, but at a low cost, it is necessary to understand common design limitation, and the effects they have on the overall dynamics of the humanoid structure. Monitoring the position of the robot is necessary for maintaining

balance; therefore many different types of sensors must be determined and implemented into the control system.

1.2. Research objectives

The main objective of this project is to research current humanoid robots in order to develop and build a low cost, yet high-end flexible humanoid.

Firstly, the flexibility and degrees of freedom (DOF) of the humanoid must be determined. This will determine the base structure of the robot where all other characteristics will fall within.

Secondly, the structure's proportions must be determined. Lower leg to upper leg ratio, pelvic height and width, mass distribution, and foot dimensions must be specified.

There are many forms of actuation that are used by developers of humanoid systems. Each form has its advantages and disadvantages with regards to its suitability for this specific humanoid application. It is important to determine which form will be most suited for this research project but more importantly which form will be both suited and within budget constraints.

In order for the humanoid to walk it must be able to balance and monitor its position. Sensors must be used to monitor foot pressures and some form of body position monitoring must be used. These sensors must be suitable for humanoid application so it is necessary to determine what the definition of "suitable" in this application actually is.

1.3. Research outcomes

A vast understanding of human-being and humanoid dynamics and structure was obtained. The result of this was the ability to understand current design issues and develop a solution to the problem keeping in mind the dynamics of a human.

Eight humanoids were designed with different solutions to various, classical, humanoid design issues. Several of these designs were incorporated into the final design which used techniques in joint articulation to overcome these 'classical' issues. The 'classical' design refers to the type of design that is both the easiest to manufacture, and control, and is therefore the most commonly manufactured. Basically this design layout consists of an

actuator layout where each degree of freedom is manipulated by one actuator positioned directly at the axis.

The lower half of the humanoid was manufactured, including the pelvis, hips, knees, ankles, feet and toes. This was entirely scratch build using regular machining techniques, the majority being CNC machined. Components that could be purchased were, e.g. gearing, pushrods, bearings, high tensile aluminium bolts, and servo motors, in order to reduce production time.

For a low cost humanoid such as this one, it was determined that the best suited form of actuation was by DC servo motor. Particularly the type that is reasonably common in the hobby robotics community. The WCK 1111 is not necessarily the most common servo but it has a lot of features that make the servo very desirable for this application, such as, full PID control, torque and acceleration control, and high resolution positioning.

In order to maintain balance it was determined that two main types of sensors need to be implemented. Firstly, pressure sensors in the feet, and secondly, an Attitude Heading Reference System for monitoring body position.

1.4. Chapter Overview

Section 2 covers the research side of this project. Many aspects of current humanoids have been investigated to determine their influence on the functionality of a humanoid. The number of degrees of freedom, the influence a toe and knee, weight distribution, inertia reduction, required sensors, control theory, and actuator selection are all reviewed.

Section 3 covers the design stage. Several different humanoids were designed based around different actuators. The final design consists of pushrod actuation in order to increase joint torque and reduce overall inertia. This 3D SolidWorks CAD model is used for producing machine code for CNC machining.

Section 4 covers the mechanical structure and construction of the humanoid. Aluminium was used for a majority of the machined components as they can be machined very accurately and cleanly. Carbon fibre was used in the leg area between joints as it is extremely strong and allows easy modification of leg lengths.

Section 5 covers the sensors needed for monitoring and controlling the movement of the humanoid. Monitoring center of gravity (COG) is essential so it is necessary to implement an inclinometer or Attitude Heading Reference System (AHRS), also foot pressure can be used to monitor stability so FSR sensors were implemented. All these sensors were tested and a consistency and accuracy was determined. For joint actuation the WCK1111 servo actuator module was used. This module has PID tuning which enables a much more precise positioning. All features of this module are tested and a protocol is established.

Section 6 covers the necessary software produced in VisualBasic.net for control and monitoring of sensors and actuator modules.

Section 7 covers the review of inverse kinematics (IK). Several different approaches are tested but due to the complexity of a humanoid leg are far too complex to be able to be realistically implemented. Therefore a unique 3 dimensional trigonometric approach is proposed which uses basic trigonometry rules making calculation of joints a lot quicker.

Over the course of this master, two articles have been published. One in ENZCon 2010 held in New Zealand and the other in I2MTC 2011 which will be held in China in May 2011. The papers are as follows:

1. P. W. E. Barlow, G. S. Gupta, S. David. (2010). “Humanoid Service Robot”. Proceedings of the Electronics New Zealand Conference (ENZCon 2010).

Abstract;

This article focuses mainly on the mechanical design of a low cost humanoid robot, with a brief background of sensors used for maintaining balance. In order to produce a low-cost humanoid with similar flexibility to a high end humanoid a dual push rod actuated joint articulation was implemented. To further increase human like similarities a toe joint is implemented. This gives the humanoid the ability to stride more elegantly, increase speed control, and reduce energy used for each step. Torque tests show that the serial controlled WCK servo should be able to produce sufficient torque when equipped with a full metal gearbox. An ‘Attitude Heading Reference System’ is used to monitor the pelvis roll, pitch and yaw, along with an inclinometer in the body. Four sensors in each foot measure

pressure which is converted through on-board processing to hexadecimal serial data. This monitors pressure on each corner of each foot. All the sensor information is sent to a RB110 micro computer which is used for controlling the humanoid gait through Visual Basic.NET programming language.

2. P. W. E. Barlow, G. S. Gupta, S. David. (2011). "Review of Sensors and Sensor Integration for the control of a Humanoid Robot". IEEE International Instrumentation and Measurement Technology Conference (I2MTC 2011).

Abstract;

A 2-legged humanoid robot requires several types of sensors for balancing and gait generation. High accuracy and resolution of these sensors is of paramount importance for successful control of the humanoid. In this paper we review some of these sensors and evaluate their suitability for use in a small-sized humanoid. Various force sensors are reviewed and their advantages and disadvantages are discussed. Gyroscopes and AHRS (Attitude Heading Reference System) are investigated and tests are performed on all units to obtain operational characteristics and accuracy. These sensors have been incorporated and evaluated on an actual humanoid which has been built in our laboratories.

2. Literature review

Legged robots have been one of the main topics in advanced robotic research. Projects addressing legged robots usually study the stability and mobility of these mechanisms in a range of environmental conditions for applications where wheeled robots are unsuitable. Recent advances in mechanical and electronic robot components, and the challenge of creating an anthropomorphic model of the human body, have made it possible to develop humanoid robots. This work has materialized into successful projects, such as the Honda humanoids, the HUBO KAIST, HIT-2 humanoid robots, and the MIT Leg Lab robots[3] [7] [8] [41] [10] .

Lately most research has been focused on getting the robot to remain stable as it walks in a straight line. Nevertheless, linear walking is not the only type of movement the biped robot will need in order to explore the real world. It also needs to turn around, lift one foot, move sideways, step backwards, etc., and the issues involved in these movements could be quite different from those of linear walking.

2.1. Current humanoids

2.1.1. ASIMO

ASIMO from Honda, to date, is the world's most advanced humanoid walking robot, with voice, visual, and gesture recognition. ASIMO has an extremely advanced gait control system that makes it able to walk, run, climb stairs, and avoid static and dynamic obstacles [3] [5].

ASIMO, an acronym for Advanced Step in Innovative Mobility stands 130cm tall, weighs 54kg, cost 1million US dollars to build, and has been under development for over 23 years. Figure 2-1 below shows the number of prototype humanoids Honda developed over 23 years.

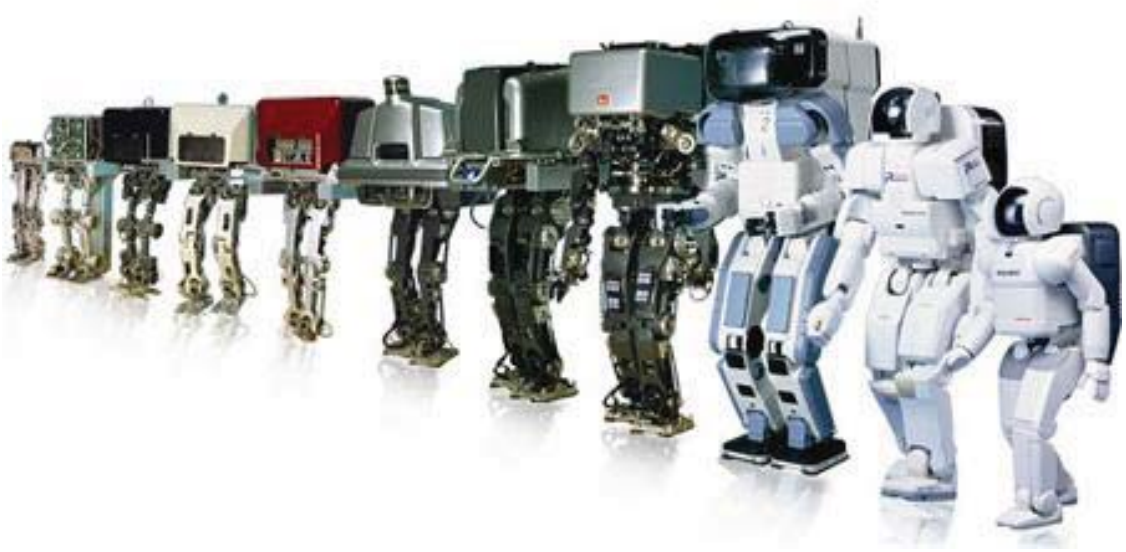


Figure 2-1: ASIMO prototype timeline. (www.world.honda.com/ASIMO/history)

2.1.2. Robobuilder

There are several hobby humanoid robots on the market that are very advanced and desirable for use in the development of humanoid gait control. However there are two that are the most desirable, the first is the Bioloid and the second is the Robobuilder. The reason for these being the most desirable is because they are the only two that have serial controlled servo control, which enables a lot more control of the speed and torque of each joint.

The Robobuilder comes with a graphical user interface which allows the user to produce motion programs for the humanoid to perform. The programs can be controlled with an IR remote control, or it can be completely autonomous. There is a method which allows pose capture which means the user can move the humanoid to a desired position and save a snap shot allowing the user to return to that position later. The system is very flexible and allows the user to program motion controls in C and C# with the help of provided sample codes.

Robobuilder uses a very advanced serial controlled module for actuation. These WCK modules have built in PID control, torque speed, acceleration control, and torque feedback. The PID motion control technology can realise motion control characteristics as precise and

as accurate as industrial servo motors. [42] Figure 2-2 shows some characteristic of the Robobuilder humanoid.

Figure 2-2: RoboBuilder humanoid (www.robobuilder.net/eng/)

This robot is regarded as more of a hobby robot with simulated actions and stances. Walking motions can be simulated, but due to the lack of gyroscopes and force sensors, genuine dynamic gait is not possible on this system.

2.1.3. Robotis Bioloid

The Bioloid is similar to the Robobuilder as it has serial controlled servos and a GUI for motion development. The Bioloid, however, is a more common choice for hobby robot enthusiasts as it is based on a system similar to Lego with the ability to swap almost all parts in order to build 23 different robots.

Figure 2-3: Robotis Bioloid in Kung-Fu stance (www.robotis.com/xe/bioloid_en)

The RoboBuilder and Bioloid are both aimed at the enthusiast who desires fast moving moves like Kung-Fu and dancing as opposed to a platform for developing dynamic gait control algorithms. The user software is therefore tailored to this sort of application and appears to be very difficult to modify to perform for a different type of user.

2.2. Number of DOF

One measure of the success of the design of a humanoid is its flexibility. The amount the humanoid can emulate the agility of its human being counterpart will determine how successfully it will perform in a real world application.

In order for a biped robot, or humanoid, to walk, the system must be able to perform two tasks. Firstly, the system must be able to lift the foot off the ground, and secondly it must be able to swing the foot to its new position. Therefore it is possible to design a functional humanoid with as little as 4 degrees of freedom (DOF), but additional DOF are required for the humanoid to approach the agility of a human being.

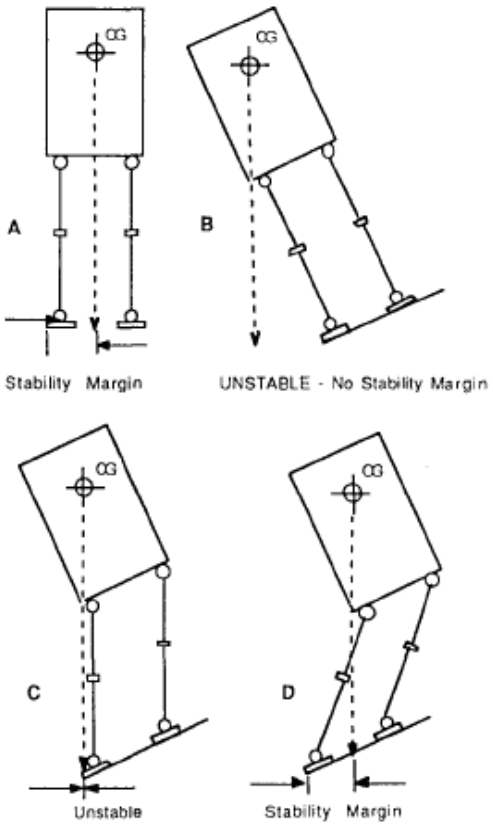


Figure 2-4: Stability margin showing centre of gravity shift. [32] P-298

(Figure 2-4) illustrates the positioning and movement of a humanoid in the frontal plane in order to maintain balance on an uneven surface. 2 degrees of freedom are required to maintain stability in the frontal plane and 2 additional degrees are needed in the sagittal plan in order to maintain 3 dimensional stability [32].

The leg and foot of a human being consists of 30 DOF. [32] The hip is a socket/ball joint which can be modelled as 3 DOF. Technically the knee has 2 DOF but can be modelled as 1 DOF similar to a basic hinge setup. The ankle and foot are extremely complicated, as the foot itself has 22 bones. [39] states that the ankle can be modelled as a 2 DOF joint.

A 4 DOF humanoid lacks the implementation of a knee joint. There have been several very successful 4 DOF humanoids, one of the more so is CURBI of Clemson University. Many very sophisticated gaits have been developed on this system. However, CURBI, along with other

4DOF humanoids demonstrate some very awkward body movements in order to place its foot in the desired position. This is due to their lack of ability to lift their foot off the ground. Instead they must swing the leg out sideways and raise the pelvis in order to achieve the same result [32].

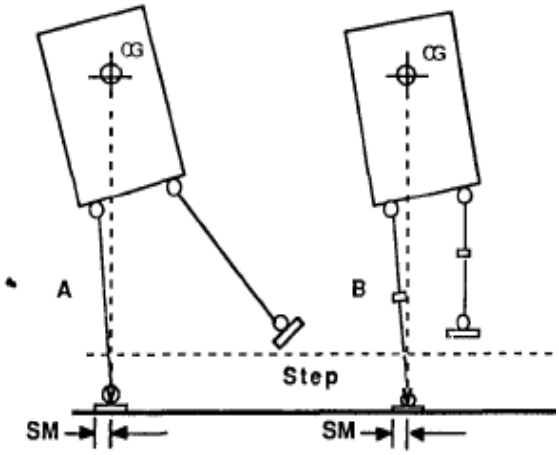


Figure 2-5: The effect the addition of a knee has on stride characteristics. [32] P-299

(Figure 2-5) illustrates the difference between a 4 DOF leg design and a 5 DOF design which implements a knee joint. It can be seen in the left image how the control system must manipulate the leg in order to raise the foot off the ground, it is obvious the effects this will have on the complexity of the system as inertia and momentum forces are far larger.

[32] states therefore that it is necessary to incorporate some form of knee joint whether it regular or inverted in the sagittal plane in order for the biped to traverse uneven terrain. The hinge joint will eliminate the need for awkward body movements and extreme centre of gravity shifting.

A sixth axis can be achieved with the addition of a vertical axis within the hip joint. This rotation results in the hip performing as a humans socket joint does, i.e. 3DOF joint. Without this axis the leg can still move three-dimensionally but the end effecter, the foot, lacks three-dimensional qualities. This third axis permits the leg to rotate which enables the humanoid to walk in a circle. [32]

The ankle is a lot more complex than first observed. Initially it appears that the ankle has a basic 2 degrees of freedom, one in the frontal plane and one in the sagittal plane. When investigated further however it becomes apparent that this is not the case. The human ankle itself has only 1 degree of freedom, acting as a hinge similar to the knee configuration. There are several tarsal bones which have characteristics resulting in a 3 DOF joint between the lower leg and foot. [32] So in simulation it would be necessary to have a 3DOF socket joint in both the hip and ankle. [32] states the rotation axis is only necessary for very complex movements such as rock climbing where the leg must be positioned very awkwardly a lot of the time. So in the case of a walking humanoid an ankle with only 2 DOF can be implemented and still achieve the same agility as a human being.

All bipeds to date lack the complexity of a real human foot. The human foot is very advanced consisting of 22 bones, 20 muscles and approximately 7800 nerves. The fact that there are 7800 nerves for force sensing, which is in turn used for balance, implies that the foot is a very important part of a humanoid robot. [32]

2.3. *Influence analysis on the implementation of a toe joint*

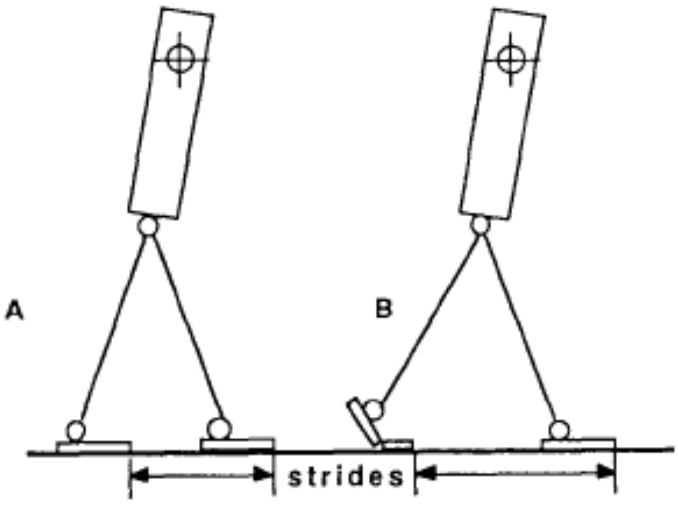


Figure 2-6: The effect the addition of a toe has on stride characteristics. [32]P-300

Corrective movements are made at both the hip and ankle joints. Some postural corrections are made through articulation of individual toes. With a flat foot the degrees of freedom,

are limited to only the ankle, this reduces stride length [7] [32] With a toe joint a stance with both feet on the ground can be maintained for a longer time allowing the COG to be transferred onto the leading foot. This toe arrangement allows the humanoid to walk more elegantly, like a human, as opposed to the more common shuffle type of walking. [14] Research performed on humans show that during fast walking the 'toe-off' torque is used to a much greater extent, this 'toe-off' torque controls the walking speed of a humanoid [43].

The human foot is so complex that it would be almost impossible to develop a mechanically operated foot with the same flexibility, strength, and size as the real one. A singular splitting foot, one toe, is sufficient for the development of a low cost humanoid with a highly intelligent gait process.

2.4. Influence analysis on the implementation of a knee cap

In order to obtain maximum performance from a mechanical system it is good practice to exploit its natural dynamics where possible. For example a bullet exiting a gun is rotating, this rotational inertia exploits the characteristics of the bullet to maintain stability.

In the case of a humanoid robot design, exploiting its natural dynamics can greatly reduce power consumption and, more importantly, reduce control complexity. There are several main areas of a humanoid structure that can be exploited [28] [4]. The first is the implementation of a compliant ankle so that the centre of pressure on the foot travels towards the center of mass of the body [28] [4]. The natural swing dynamics of the leg give the walking motion more of a natural appearance. The final mechanism to be exploited is the adaption of a knee cap. This knee cap enables very easy height control due to the knee joint being limited to motion in only one direction. As well as enabling easy high control, a knee cap greatly reduces power consumption as the knee joint actuators do not have to be continually monitored and manipulated in order to stop the joint from inverting [28].

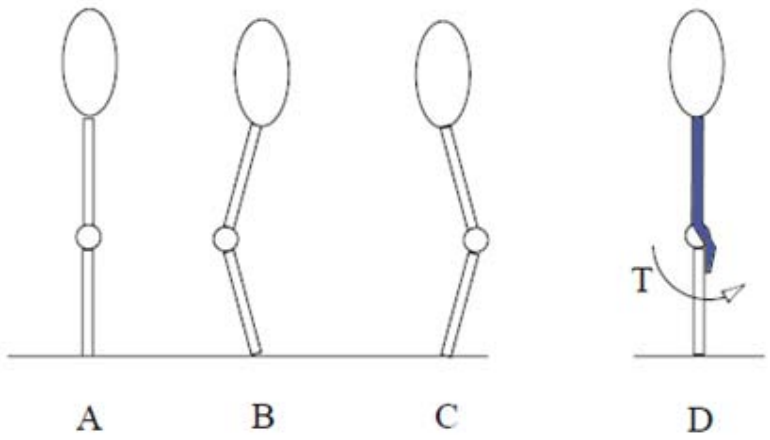


Figure 2-7: Influence analysis of knee cap. [28] P-66

(Figure 2-7) above demonstrates this issue. On the left the body position is directly above the foot resulting in zero torque at the knee joint. If however this joint is non-locking, any small movement either to the left or right will result in the leg collapsing. To control a situation such as this is very difficult as control systems will always exhibit some form of delay, whether in transmission or computation. The result of a control system with a virtual knee cap is a chattering, where the system cannot find its desired position. The result is similar to a basic PID or Fuzzy logic controlled system where a desired position is required but in order to reach this position quickly there must be partial overshoot. With a knee cap the joint can be driven very quickly into position and a minimal torque can be continually applied in order for the joint to maintain rigid.

2.5. Pelvic Pivot and Twist

When walking, the pelvis does not remain static. It is manipulated with the same number of DOF as a hip joint. However, to control the pitch of the pelvis the hip pitch axis is used. This means there are only two DOF in the pelvis yet there are three to control in the system. It is essential that the roll of the pelvis is as close to the centre of the hip axis. This is because the further the pelvis axis moves away from the hip the greater the swing effect. [7] i.e. instead of the pelvis rolling side to side, lifting and dropping the hips, the pelvis will swing the entire lower body side to side. This makes for harder gait control [43].

The second axis of the pelvis is not as crucial. This is the vertical axis which is positioned between the lower body and upper body. It is preferred to have the pivot point close to the lower body as this gives a higher weighted upper body which helps with reducing momentum in the lower body [44].

2.6. Methods of actuation

There are many different forms of joint actuation that can be implemented in the construction of a humanoid robot. Each form has its advantages and disadvantages with respect to the agility and characteristic specifications of the humanoid. Some robots, for example, need power over flexibility and speed, whereas hobby robots need speed to perform the kung-fu style moves desired these days.

2.6.1. Hydraulically actuated

If power is what is required then hydraulic actuation would be regarded as the first choice. Hydraulic actuation has a very similar power to weight ratio as a human muscle. The only issue is that in order to power the hydraulic system a large pump and reservoir is needed which in most cases is too large, and awkward, to have on board the humanoid system. Also to control each ram a bulky control valve is needed, which can be seen on the left of (Figure 2-8) below.

Figure 2-8: Hydraulic actuated humanoid legs. [44] P-15

Figure 2-9: Hydraulic actuated humanoid legs showing control system on left. [44] P-17

Hydraulic actuated humanoid limbs are very compact and resemble the human structure very well. Due to this, hydraulic systems are very attractive to humanoid researchers, which have resulted in large technological growth in hydraulic pump/reservoir and valve systems.

2.6.2. Pneumatically actuated

Pneumatics is a nice form of actuation as they simulate more of an elastic feel [45]. The issue, however, is that pneumatic rams cannot be accurately positioned like hydraulic rams can. They are more suited to a fully open and closed type system where the ram is either fully retracted or at its maximum extension.

There is however another form of pneumatic actuation. Air muscles have become more common in the humanoid development industry as they are very compact and human-like. The most advanced anthropomorphic limb created to date is the Shadow Hand, comprising of 25 individual degrees of freedom, actuated entirely by compliant air muscles [45].

These air muscles are prone to the same disadvantages as the hydraulic systems. They need an off-board compressor/reservoir, along with individual valves for controlling each muscle. They therefore are not suited for a fully stand alone humanoid system.

2.6.3. Passive

Some walking bipeds do not have any means of joints actuation. They rely solely on gravity exploiting the natural dynamics of the biped in order for it to walk. [21] [25] [26] Passive walking robots have been around for a very long time. Passive walking robots with fully

compliant knees, however, are a recent adaption to the research field of this form of biped. The PDR400 is a passive dynamic running biped with knees that was developed at the University of Japan. This biped is the first of its type and has demonstrated the effects that dynamics of the design have on the ability of a passive robot to walk. This biped has unique characteristics such as; hip springs, leg springs, parallel link mechanism, shock absorbers and hyper extension mechanisms which enables the biped to both walk and run.

Passive biped development is solely undertaken to establish and investigate natural dynamics [26] [48]. They cannot be used in a useful real world application due to the fact that they must have a gradual incline to walk down.

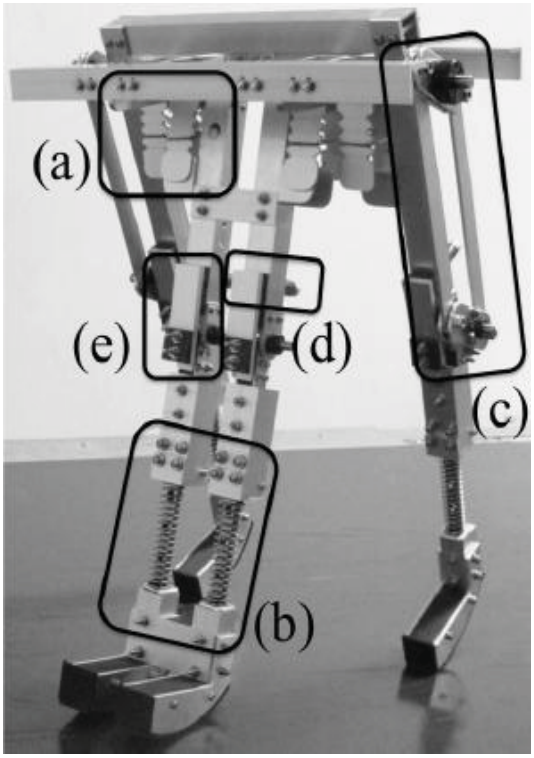


Figure 2-10: Passive walking robot. [26] P-2

(Figure 2-10) above is an image of the PDR400 passive walking robot. This illustrates the number of additional mechanisms the must be developed and implemented in order for the

biped to walk. The system is extremely bulky and does not resemble a humanoid as it is only 2 dimensional.

2.6.4. Electric

By far the most common form of actuation is by DC servo motor. A servo motor has built in position feedback giving it the ability to be easily controlled. In most cases a servo motor is a fully self contained unit which has input for power and position. This differs from a regular DC motor in the fact that the DC motor requires both a driver circuit and an additional encoder to monitor its position.

With all technologies there is a range of performance characteristics each servo achieves. High performance servos are very compact and powerful but are very expensive. Conversely a low cost servo may be compact but in most cases it will be very underpowered for its size [40] [42] In order for a humanoid robot to be as human-like as possible it is necessary to use extremely light and high powered servos due to the very high performance power to weight ratio the human muscle has. The issue with high cost servos is that it greatly reduces the number of research facilities that can afford to undertake humanoid robotics projects which reduces that rate of development. The advantage of using high power low weight servos is the ability to mount the servos in places that will allow for easier control of the humanoid.

2.7. Positioning of actuators

As mentioned, the position of a higher powered servo is not as important and does not result in as many issues as a lower powered servo. The power a servo must produce with respect to its weight grows exponentially the further the mounting point is from the pivot point.

Generally the lower the overall inertia of a component, the better it is. For example it may seem desirable to have heavy feet on a humanoid to help balance, when in fact the effect of this is the opposite. The legs inertia will increase dramatically resulting in the need for a high powered servo in the ankle and hip. A high powered low cost servo will be larger and heavy which will inevitably result in the need for high powered servos further up the body producing an undesirable waterfall effect resulting in an underpowered heavy weighted robot.

2.7.1. Pros and cons of various positions

The advantage of positioning actuators directly at the pivot point on a joint is the large reduction in both manufacturing cost, and the need for precision. The disadvantage of this however, is the lack of the ability to modify in order to change leverage if the product is a prototype [29] [39] [43]. This layout is what is regarded as a 'classical' design, which results in the increased limb inertia.

Mounting the actuator slightly away from the pivot point with gear, or belt drive, gives the ability to modify the overall leverage. The issue with this is that the overall complexity of the product increases, which increases the manufacturing and maintenance times.

A pushrod linkage type layout gives the ability to position the actuator anywhere along the classical humanoid design of the sagittal plane. This means the inertia can be reduced and the overall centre of gravity and weight distribution can be manipulated to exploit the natural dynamics of the robot. With reducing inertia the power of servos can be reduced making it possible for low budget research facilities to undertake such research. The disadvantage of a pushrod layout is the increased control complexity.

2.8. Forms of sensing pressure on foot

2.8.1. Matrix

The National Institute of Advanced Industrial Science and Technology in Japan has developed a humanoid robot named H7. This humanoid is capable of walking on flat hard surfaces and up stairs due to the implementation of inertial/gyroscopic sensors and 6 axis transducers. What the Institute is developing at the moment is a Matrix foot sensor in order to obtain variable pressures over the entire foot to enable the humanoid to walk on uneven and varying consistency surfaces.

This sensor works basically the same as a regular force sensor where there is an interdigitated surface which is then covered with a conductive rubber surface. When this rubber surface is pushed into the interdigitated surface the resistance is varied producing a resultant force. The difference, however, is that the matrix sensor has hundreds of very small sensors over the entire surface which are individually measured with a matrix array controller. [10] [11]

(Figure 2-11) shows the interdigitated fingers of the matrix array. These are generally made the same way as a PCB as it is very easy to modify and fast to produce. [10]

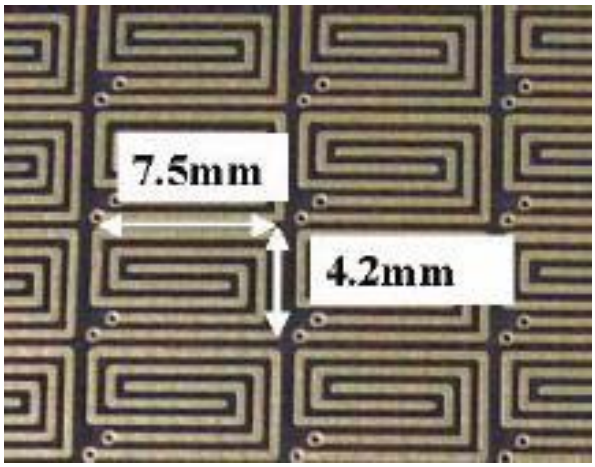


Figure 2-11: Interdigitated conductive matrix array. [11] P-2

(Figure 2-11) demonstrates the effect of a 30gram weight situated in the centre of the matrix sensor.

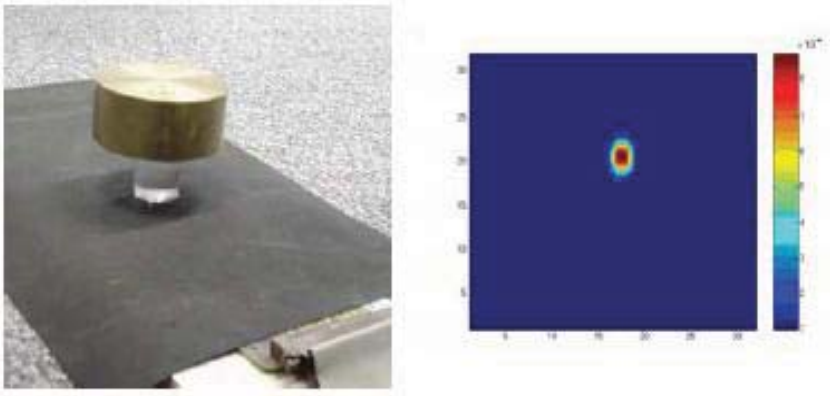


Figure 2-12: Result of force applied to a concentrated area of the matrix sensor. [11] P-4

The issue with these sensors is their complexity when it comes to usabilty. Determining usable data from an array of data is very tedious and requires a lot of processing [10]. The main issue however is their availability. These sensors are under development by several institutes and are yet to be commercially available.

2.8.2. FSR

In order to select a touch sensor, one has to look at the application in order to choose which sensor is the most suitable for a given functionality.

Fabrizio Vecchi and colleagues [46] have experimentally evaluated both the Interlink FSR and the Tekscan Flexi-Force through a series of measurements. They concluded that the FlexiForce sensors present better response in terms of linearity, repeatability, time drift, and dynamic accuracy, while the Interlink FSRs are more robust.

In short, if large changes in force are applied at a relatively high frequency, it appears the Interlink or LuSense sensors should be selected, whereas if large slowly-varying forces are applied infrequently for long durations, the Tekscan sensor is likely to perform better

Since in our application, changes are large and changes in force are applied at a relatively high frequency the interlink FSRs are best suited [47].

In the actual application however, the ability to maintain a consistent, and accurate, area and position of the applied force will be the limiting factor in terms of sensor accuracy and precision. Indeed, the time needed for the sensor to relax is an important variable. When not taken into account, it may induce errors in the measurement [46].

2.8.3. 6 Axis transducer

A six axis transducer measures force in all directions. In most cases these sensors are circular and are mounted at the face and bottom of the sensor. A six axis transducer could be used in conjunction with pressure sensors positioned on the sole of the foot. This sensor would be positioned between the ankle and the foot, basically allowing a means of torque monitoring directly at the ankle. The physical implementation of such a sensor would be easy, the only issue is that they are very expensive and out of price range for this research project.

Below in (Figure 2-13) is an example of six axis transducer. This particular model, the MINI40, is made by a reputable company called ATI Industrial Automation and its torque range is suited for this humanoid application.

Figure 2-13: ATI MINI40 miniature 6 axis load cell. (www.bl-autotec.co.jp/FA/01_02ati/mini40.html)

These transducers are constructed from high yield strength stainless steel with aircraft grade aluminium face plates.

SENSING RANGES	Calibrations		
AXES	SI-20-1	SI-40-2	SI-80-2
Fx, Fy (+N)	20	40	80
Fz (+N)	60	120	240
Tx, Ty (+Nm)	1	2	4
Tz (+Nm)	1	2	4

Table 2-1: Sensing ranges of the Mini series transducer

These transducers have a high signal to noise ratio due to their silicon strain gauges. These provide a signal 75 times stronger than conventional foil gauges. This signal is amplified, resulting in near-zero noise distortion [16].

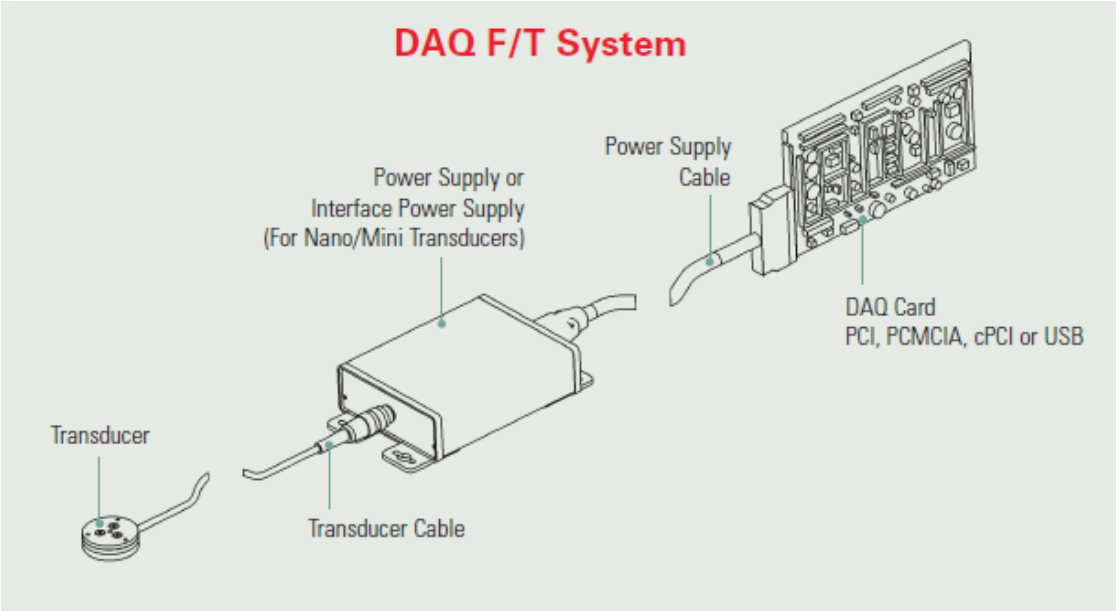


Figure 2-14: DAQ system layout for ATI transducers (www.bl-autotec.co.jp/FA/01_02ati/daq_system _layout.html)

The issue with these sensors is that even though the sensor itself is extremely small the additional systems and controllers needed for data acquisition are very large. It can be seen in (Figure 2-13) above the size of the additional equipment needed. Obviously the processing could be performed through simple analogue to digital conversion on the micro computer controller. For this project however it is desired to have as much of the processing performed on board the sensor itself, reducing processing time on the controller.

2.9. Position monitoring

In order to maintain balance a control system must monitor two main aspects of the humanoid. Balancing robots are a typical example of an inverted pendulum system. An inverted pendulum system consists of a mechanical system where its centre of gravity is above its point of support [6]. A system like this is naturally unstable and without input will fall over.

2.9.1. Centre or Gravity

The centre of gravity, or COG, is the point of the robot at which the robot will balance in all directions. This is ideally situated in the pelvis of the humanoid [6]. Control and monitoring of the COG is essential for maintaining balance as an inverted pendulum system will

naturally try and rotate to become a regular pendulum, which in the case of a humanoid simply means the humanoid has fallen over [20].

2.9.2. Zero Moment Point

The ZMP method was introduced by Miomir Vukobratovic in 1968. Miomir's method states that, the point with respect to which dynamic reaction force at the contact of the foot with the ground does not produce any moment (i.e. the point where total inertia force equals 0), assuming that the contact area is planar and has sufficiently high friction to keep the feet from sliding [27].

Ground reaction force (GRF) and ZMP are usually measured with a series of sensors embedded in the feet. Pressure sensitive transducers, foot switches, strain gage based sensors, force sensitive resistors, and novel force-torque transducers are the different kinds of sensors which can be used to calculate the ground reaction force and ZMP. In this project Force Sensing Resistors (FSR) are used to calculate the ZMP. But before looking into the FSR's and their implementation in this project a concept of Centre of Pressure (COP) which is interconnected with ZMP needs to be considered.

2.9.3. Centre of pressure (CoP)

COP is a ground reference point where the resultant of all ground reaction forces acts. At this point, it is assumed that all of the forces that act between the body and the ground through the foot can be simplified to a single ground reaction force vector and a free torque vector. [24] (Figure 2-15) shows a single resultant force derived from 4 corner forces.

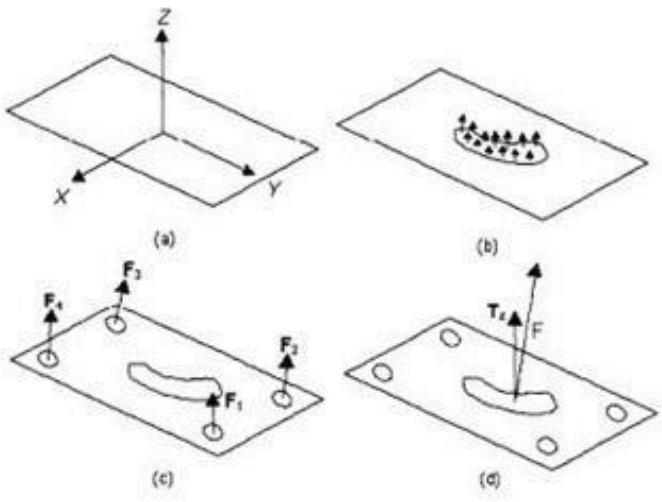


Figure 2-15: Center of Pressure resulting corner forces. [27] P-6

So, for flat horizontal ground surfaces, ZMP = CoP.

At any point P under the robot, the reaction can be represented by a force and a moment MGRF

$$\boldsymbol{M}_{grf}\left(\dot{\boldsymbol{r}}_{zmp}\right)\big|_{horizontal}=0$$

$$\text{At } \boldsymbol{P}: \quad \boldsymbol{M}_{grf}=\left(0,0,M_z\right)$$

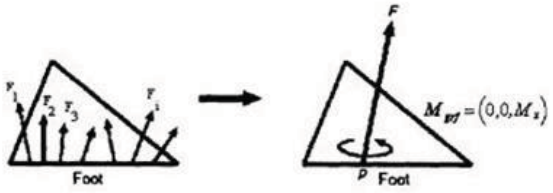


Figure 2-16: Kudohs explanation of localising forces in the ZMP. [27] P-7

In (Figure 2-16) Kudoh et el, explain that around the ZMP (localized at rZMP) the moment around the horizontal axis are zero and there is only a component of moment around the vertical axis. The resulting moment of force exerted from the ground on the body about the ZMP is always around the vertical axis. At the ZMP is a reference point at the ground in which the net moment due to inertial and gravitational forces has no component along the (horizontal) axes (parallel to the ground). The trajectory that the ZMP follows is utilized and

planned such that they are within the supporting polygon defined by the location and shape of the foot print [26][24].

In short Miomir Vukobratovic's idea of ZMP can be summed up as;

a) Leg motion is prepared in advance

b) Body motion is calculated so that the ZMP (CoP) exists in the supporting foot of the robot.

2.10. Humanoid Control

For a good control process for a walking robot, 3 things need to be considered
1) Gait process: the sequence of leg movements

2) Body movement for supporting legs

3) Foot placement

Before looking into the above 3 aspects, a few basic concepts need to be understood. To fully understand and control a walking robot, one needs to grasp the concept of stability involved in walking.

Stability, in walking, is the capability to maintain the body posture given the control patterns. There are two forms of stability; it can either be static or dynamic stability. Statically stable walking implies that a posture can be achieved even if the legs are frozen or when the motion is stopped at any time, without loss of stability. Whereas, dynamically stable walking implies that stability can only be achieved through active control of the leg motion. Static walking systems can be controlled using kinematics models, whereas dynamic walking requires the use of dynamical models (Inverse Kinematics). [49]

Motion of a legged system is called walking if in all instances at least one leg is supporting the body [18]. If however, there are instances where no legs are on the ground, it would be considered that the humanoid is running. Walking can be either statically stable or

dynamically stable, but however with 2 legs it is almost always dynamically stable. While running it is always dynamically stable.

Other aspects which affect the control of a walking robot are the Leg control Patterns and the number of Degrees of Freedom (DOF) available in the legs. [18]

Leg control patterns have 2 major states:	Stance: On the ground. Fly: In the air moving to a new position.
The Fly state has 3 major components:	Lift phase: leaving the ground. Transfer: moving to a new position. Landing: smooth placement on the ground.

2.10.1. Humanoid Gait

Gait is the pattern of movement of the limbs during locomotion over a solid substrate [1]. It can be simplified by saying that gaits determine the sequence of configurations of the legs, a sequence of lift and release events of individual legs. In walking at least one leg is always in contact with the ground. In bipedal locomotion, this results in periods of double support, in which both legs make contact for some time in the gait cycle.

Gaits can be divided into main classes –	Periodic gaits or Non-periodic gaits

Periodic gaits are when the same sequence of movements are repeated e.g. walking forward in a straight line. Where non-periodic gaits have no periodicity in the control and should be controlled by the layout of the environment.

The number of possible events N, in a gait process, for a walking machine with K legs is given by: $N = (2K – 1)!$ [18]

As we have 2 legs on your robot we get a value $K = 2$ so therefore $N = (2*2 – 1)!$ giving 6 possible events possible - Lift left leg, lift right leg, release left leg, release right leg, lift both legs, release both legs.

2.10.2. Gaits – Human vs. Humanoid robot

People and humanoid robots are not statically stable. For people, standing up and walking appear effortless, but we do not realize we are actually using active control of our balance. Humans use muscles and tendons to do this and robots use motors.

Biped walking is similar to rolling a polygon, where polygon side length equals the step length. And as a step gets shorter it becomes more like a rolling circle. In order to remain stable, the robots Center of Gravity (COG) must fall under its polygon of support. Where, the polygon is basically the projection between all of its support points onto the surface. And so we can say, in a biped robot, the polygon is really a line. The center of gravity cannot be aligned in a stable way with a point on that line to keep the robot upright. [18][41] (Figure 2-17) illustrate this principle.

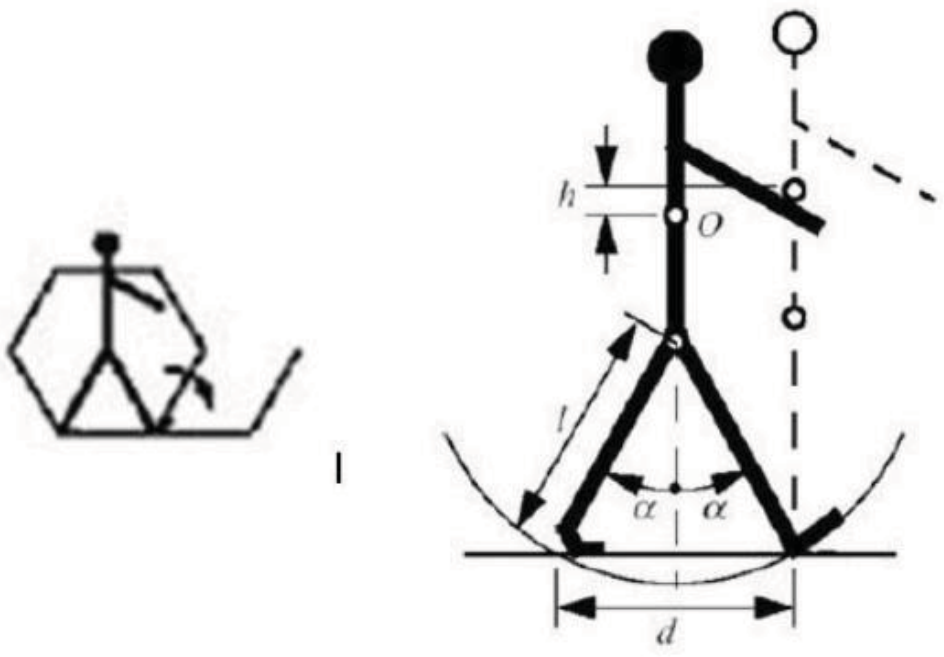


Figure 2-17: Depicting Biped Walking to a rolling Polygon

3 Mechanical Design

3.1 *Dynamics*

It is recommended to bring motor weight up as high as possible. [7] [9] [28] i.e. if possible bring the ankle motors up into the knee, bring the knee into the hip, and the hip into the body. However this layout does not need to be completely implemented as it is not entirely necessary, and increases complexity. Therefore only two characteristics were used in this project, raising the ankle into the knee, and raising the hip into the body.

3.2 *Determining Humanoid Dimensions*

The specifications for this project were very lenient as the humanoid was being built from scratch right from the beginning. A height of about 2 to 3 feet was specified to be an appropriate height. The dimensions of a human being are very geometric which is demonstrated by Leonardo Da Vincis' classic 'Flower of Life' shown in (Figure 3-1).

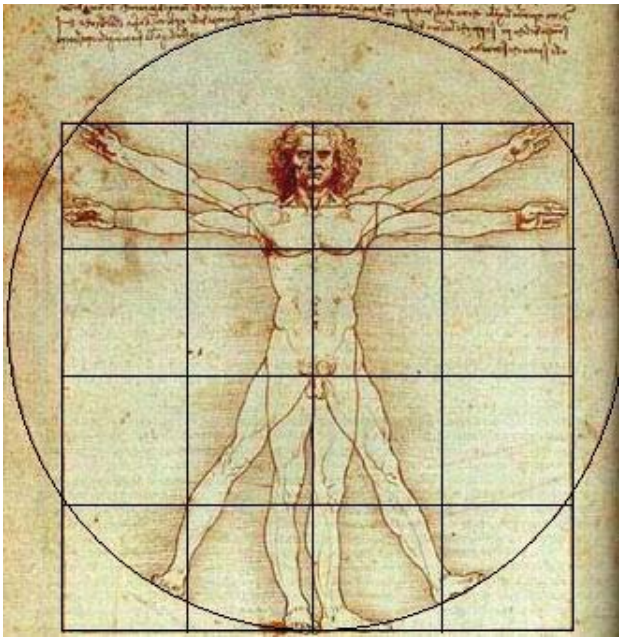


Figure 3-1: Leonardo Da Vincis' 'Flower of Life' representing the dimensions of a human being (www.astoriabrown.com/sacred-geometry.html)

(Table 3-1) Shows the dimensions of an average human being. On the right are the dimensions determined for the humanoid as this would result in a 750mm high humanoid. This is using a scaling factor of 0.425.

	Human Measurement (mm)	Humanoid Measurement (mm)
Foot - Knee	420	179
Knee - Hip	420	179
Hip - Shoulder	620	263
Shoulder width	400	170
Shoulder – Elbow	300	127
Elbow – Wrist	270	115
Hip width	250	107
Foot length	230	98
Foot width	100	42

Table 3-1: Human being and determined Humanoid dimensions

3.3 *DOF*

To develop a humanoid with the agility of a human-being it has become apparent that, firstly, the hips must perform as a socket joint. i.e. 3 degrees of freedom, enabling 3 dimensional movement. There must be 1 DOF in the leg producing a knee to enable the leg to be lifted above the ground. It is not necessary to develop the foot and ankle area to the same complexity as a human-beings', instead a simple 2 DOF ankle with a single splitting foot to simulate a toe is sufficient. (Figure 3-2) below is a schematic representation of the number of DOF that are implemented in the humanoids leg and hip area. This area alone has a total of 14 DOF, this does not include the pelvis area which has an additional 2 DOF.

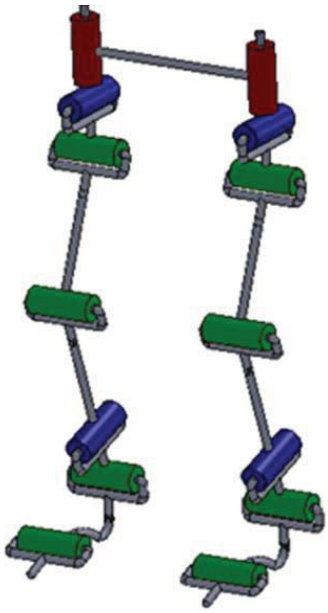


Figure 3-2: Schematic representation of DOF

3.4 *Initial Designs*

Many different concepts were considered throughout the design and researching stage. A majority of these designs have been discarded and it was decided that the overall structure would incorporate a dual actuated ankle to increase power and to allow for desirable actuator placement. The overall proportions of the humanoid are based on scaled down, ideal human proportions.

3.4.1 Designs using DC motors and harmonic drives

The first two designs, (Figure 3-3), incorporated DC motors with harmonic and planetary drives. To produce a dual acting ankle, DC motors were placed in the knee with linear roller ball screws to act as a linear displacement. To actuate the knee a similar linear ball screw is proposed with the motor placement in the hip. The pelvis area of the body, including the hip area, at this stage is based on classical design where the motors are situated directly at the axis and act as a bearing.

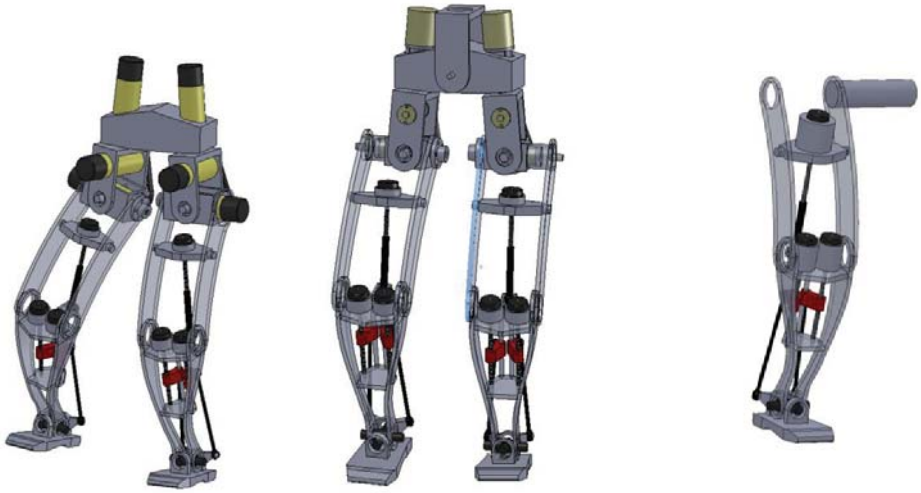


Figure 3-3: CAD designs of concept 1 & 2 using DC motors and harmonic drives

There are many issues with these particular designs. Firstly and most importantly, the DC harmonic drive motors that were found to be suitable in size and torque are far too expensive, at $1500 each. If this was disregarded there are still several significant issues. For example, each axis would need its own angular encoder, and each motor would need a dedicated motor driver with associated pulse width modulation. This greatly increases the complexity of the system both in manufacturability and controllability.

3.4.2 Designs using WCK and AX servos

Several other iterations similar to designs 1 and 2 occurred before Serial Controlled actuators were discovered. Benefits and specifications of these servos are discussed further on in this thesis. The servos used in designs 4 and 5 are the WCK1111 and the RX-28 which at the time were assumed to have similar network protocols. (Figure 3-4) on the left shows a RX-28 as the knee actuator. The reason for this is because initially it appeared that the knee would require the largest torque and therefore the RX-28, which produces 28kg/cm of torque, was proposed. After further research however, it became apparent that the largest torque is situated at the hip joint [34]. After further investigation of the network protocols it was found that the WCK servos are a full-duplex protocol whereas the RX (Dynamix) is only half-duplex which would require additional circuitry. The solution to this was simple, 2 WCK servos were used at the knee.

This particular design allows lower specification servos to be used due to the novel dual pushrod actuation implemented in both the ankle and the hip. Having two servos actuate both roll and pitch of the ankle and hip ensures that both servos support the load of every movement, producing a joint with twice as much torque potential as a classical single actuator design. Also with the addition of pushrods we are able to move the servos further up in the body to reduce inertia. These two unique design characteristics are what have made the development of this low cost, high agility humanoid possible.

Figure 3-4: CAD designs of concept 4 & 5 respectively

3.4.3 Designs using WCK serial servo modules

(Figure 3-5) on the left shows design number 6. This has the beginnings of the implementation of a toe, the addition of pelvic pitch, roll, and also the hip yaw servos have been positioned closer to the centre of the body with pushrods in order to increase how compact the pelvic area is.

(Figure 3-5) on the right illustrates design number 7 which is the first design resembling the final layout of the system. This has a carbon fibre tubular leg section which makes for an extremely lightweight, simple to build, structure. Most importantly, however, is that these sections allow for the possibility of reducing or increasing the height of the humanoid with

ease. This design has also moved the hip actuators above the pelvis yaw pivot in order to increase how compact the pelvis area is.[15] discusses the importance of weight distribution in a humanoid with respect to leg design.

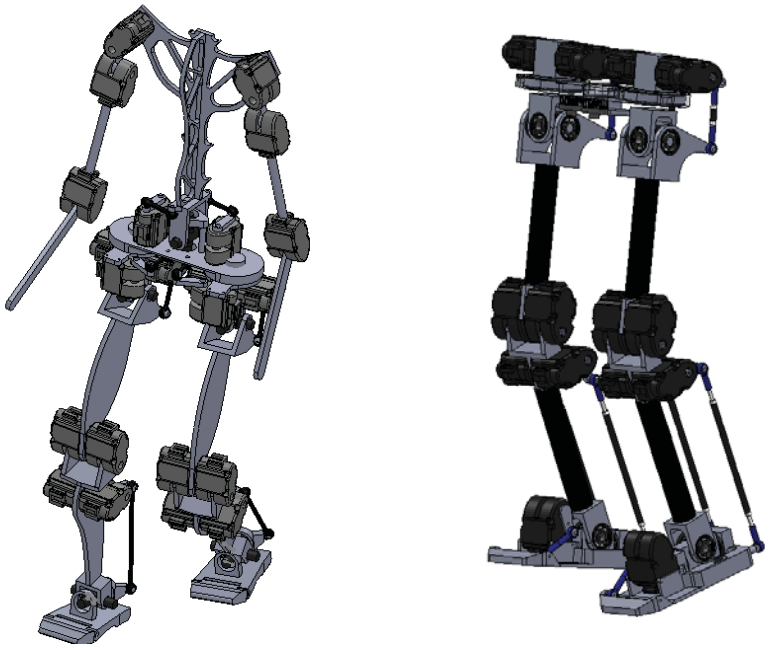


Figure 3-5: CAD designs of concept 6 & 7 respectively

This is not the final WCK module design. The final design in described and discussed in greater depth in section 4. Through many iterations it became clear what the most efficient servo positioning and orientation was. The roll and pitch servos of the hip are positioned above the pelvis, the hip yaw servos are positioned in the center/bottom of the pelvis. This results in a total of 3 servos per hip. The knee uses two servo modules in parallel purely to increase total available torque. The ankle is actuated in a similar manner to the hip except with now yaw actuation. The ankle servos are position up high right by the knee in order to reduce overall rotational torque which is reviewed and described in the following section. There are a total of 8 servos modules in each leg allowing for 7 degrees of freedom. There is then the pelvis rotation and twist resulting in a total of 18 servo modules in the lower body.

3.5 *Inertia reduction*

Below are basic calculations to show that the inertia has decreased by 50%. To control the joints some form of linear actuation would have to be used in order to translate movement from the knee to the ankle. Linear actuator/screws/push rods would be the most cost effective solution. Since a linear pushrod solution is being used it is possible to use two actuators to control both movements. i.e. Instead of one motor for roll, and one for pitch, both motors will manipulate both roll and pitch. This technique doubles the power of each joint meaning that less expensive modules can be implemented.

In order to make the pelvis and hip area as compact as possible, gears are implemented in the hip vertical axis, and pelvis roll and yaw axis. (Figure 3-6) shows classic actuator positioning, and proposed positioning.

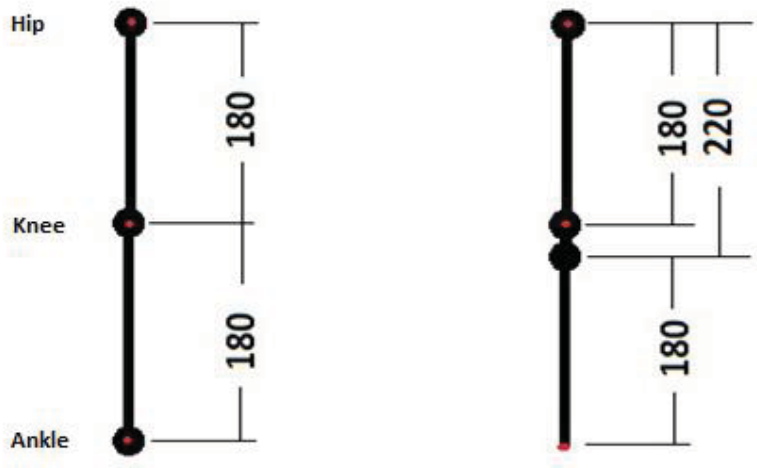


Figure 3-6: Demonstrating inertia reduction between classical design and proposed design

(Figure 3-6) Left shows classic actuator positioning with motors at the bottom of the leg where as the right shows the proposed layout.

Calculating leg inertia:

$$I = mr^2$$

(3-1)

The basic inertia equation is shown above in (3-1). This assumes the mass is situated at a particular radius with a zero mass link between the two. The real robot however does not have a mass-less link in the leg but for means of simulation this is suitable.

Servo weight, 55g each

Servos mounted in the foot.

$$0.11 \times 0.18^2 = 0.00356 \frac{kg}{m}$$

(3-2)

To evaluate the inertia of a classic leg layout viewed from the hip, firstly the inertia of the knee joint is calculated above in (3-2). The mass of two servos is 0.11kg which are situated 180mm away from the hip. This results in a total rotational inertia of 0.00356 kg/m at the knee joint.

$$0.11 \times 0.36^2 = 0.01426 \frac{kg}{m}$$

(3-3)

Secondly the inertia of the servos situated at the base of the leg, or foot, must be calculated which is shown in (3-3).

Total inertia:

$$0.00356 + 0.01426 = 0.01782 \frac{kg}{m}$$

(3-4)

(3-4) refers to the total inertia of the entire classic leg layout.

Servos mounted in the knee.

$$0.11 \times 0.18^2 = 0.00356 \frac{kg}{m}$$

(3-5)

$$0.11 \times 0.22^2 = 0.00532 \frac{kg}{m}$$

(3-6)

The calculations are them performed exactly the same for the proposed new layout with however a modification to the lower section of the leg. Instead of the second calculation positioning the servos at 360mm they are positioned at 220mm which is shown in (3-6). (3-5) is exactly the same as the classic layout as the servos are positioned the same.

Total inertia:

$$0.00356 + 0.00532 = 0.00889 \frac{kg}{m}$$

(3-7)

(3-7) shows a total inertial for the proposed leg layout of 0.00889 kg/m. This is derived by summing (3-5) and (3-6).

$$0.01782 \times \% = 0.00889$$

(3-8)

Using a simple calculation the percentage difference of both systems can be determined. A classic system has an inertia of 0.01782 and the proposed new layout would have an inertia of 0.00889 kg/m. This is a 50% reduction which can be seen by solving (3-8). This analysis is purely theoretical and is only used to show the large influence module positioning has on overall dynamics. This is not used to evaluate or even estimate the inertia of the actual system that was developed.

3.6 *Discussion and conclusions*

The first step to designing a humanoid robot is to determine the basic overall layout. For example, the dimensions of the legs, pelvis width, and foot size must be specified. The dynamics must be determined, for example how many degrees of freedom are needed, and what is the flexibility and overall movement of each joint. It became clear that a major issue restraining the performance of a humanoid was the power to weight ratio of actuators. Low budget research institutes cannot afford high power, low weight, actuators. This in turn limits what can be developed at their institute, inevitably stunting the overall humanoid development.

To be able to use low cost actuators it was necessary to implement a unique form of actuation. This form of actuation would both increase the torque applied to the joint due to a dual servo effect but also reduce inertia of each leg by 50%.

To ensure the humanoid would have the same flexibility as a humanoid being it is necessary to have a total of 6 degrees of freedom (DOF). 3 in the hip to simulate a ball joint, 1 in the knee to simulate a hinge, and 2 in the ankle to produce a roll and pitch axis. It was decided that to enable the humanoid to stride like an actual human being it was necessary to adopt a single splitting toe.

4 Computer Aided Design & Mechanical Structure

There were several iterations from the 1st design to the final. For example there were changes in bearing sizes, material thicknesses, pivot points, and gear ratios. Solidworks is a 3D CAD package which was used to produce a complete 3 dimensional model which is a great help in avoiding conflicting parts and dimensions. These models are also used for CAM software which is used in CNC computer aided machining.

4.1 *Pelvis*

One of the many issues with designing a humanoid robot is the difficulty of positioning pivot points in desired locations. In most cases a pivot point will be altered from an ideal position due to constraints such as actuator position and size, resulting in a pivot point 'swing' effect. Below in (Figure 4-1) is an illustration of the effect undesired pivot point positioning can have on components. The left image shows a component with its pivot positioned directly in the centre of the part, the result being a rotation about that point with the centre of mass of the part remains still. On the right is an illustration of a pivot moved away from the centre of the component. This results in a 'swing' effect which, instead of simply rotating the component, swings the whole component from side to side. This is undesirable in the case of a humanoid robot as it alters the centre of gravity characteristics making control much more difficult.

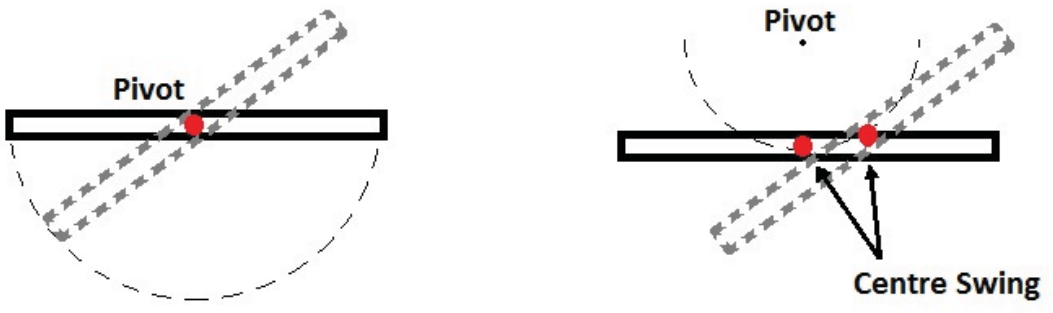


Figure 4-1: Effect pivot placement has on pelvis trajectory

(Figure 4-2) below shows the 3-D model of the hip roll, and pitch mechanisms. The blue portion shows the roll axis, the red shows the yaw axis, and the green component is the pelvis itself. In order to acquire a pivot directly at the centre of the pelvis area, and to also

maintain a compact pelvic area, gears were used. Gearing is able to be used in the pelvis roll, and pitch, as well as hip yaw due to the fact these axes do not require much articulation but do require a high torque. With the implementation of gears the system has achieved a very compact layout with a very low overall rotational inertia.

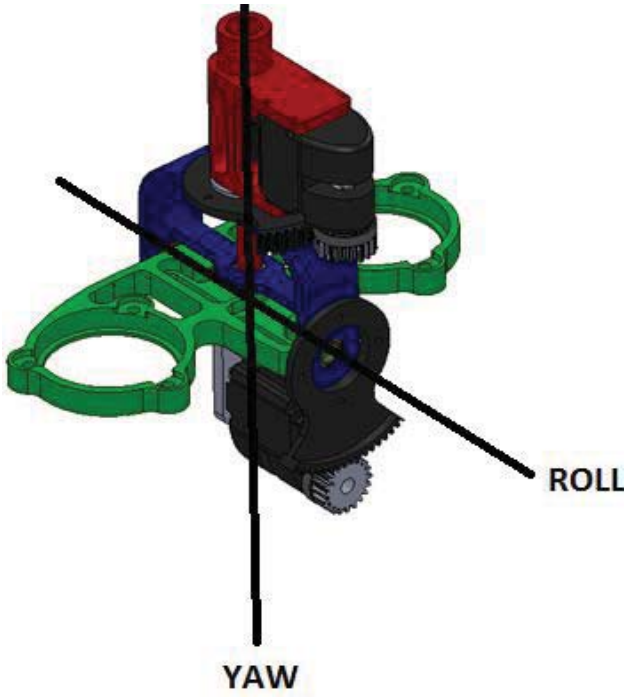


Figure 4-2: CAD design demonstrating roll and pitch of pelvis area

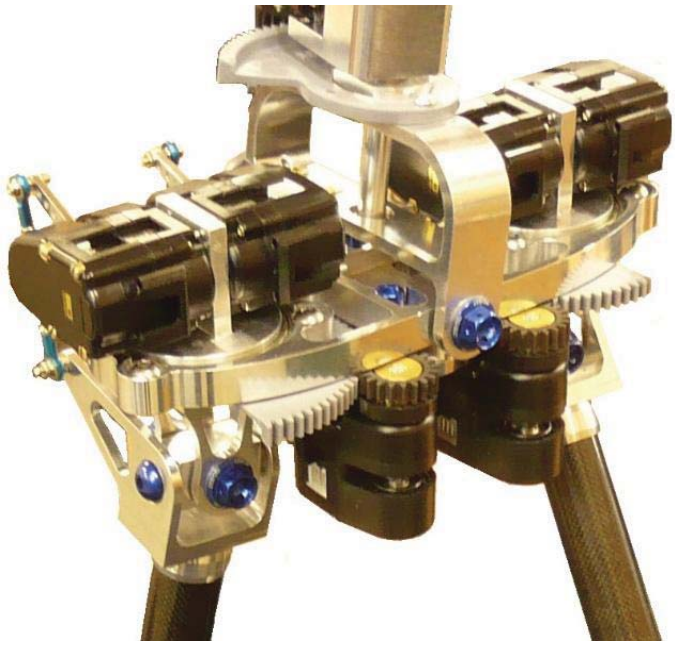


Figure 4-3: Completed pelvis area of actual humanoid

Above in Figure 4-3 is the entire pelvic area of the completed humanoid robot. This area has the same flexibility as a human being.

Figure 4-4: CAD design of Pelvis next to the actual CNC'd part

Above in (Figure 4-4) is the completer pelvis alongside the CAD design. This was the first piece to be machined

4.2 *Hip*

A human hip is a ball joint. This means that there are 3 degrees of freedom. To achieve this in a mechanical application an extra vertical axis is added to a standard pitch and roll type configuration essentially producing a yaw axis. [22] discusses the implementation of a unique double spherical joint as a hip which is very appealing. The issue with it however is the complexity of construction and implementation would be far too great.

To manipulate the roll and pitch of the hip, the two push rods connecting the actuators to the green component are manipulated. If both actuators are moved either up or down together it will move the leg directly forward or backward. If one actuator is moved upward and the other downward the leg will move sideways, therefore the leg can be moved to any angle with a combination of these two effects. A yaw axis is introduced with a pivot point through the centres of both pitch and roll axis. This is moved with a drive gear mounted in the centre.

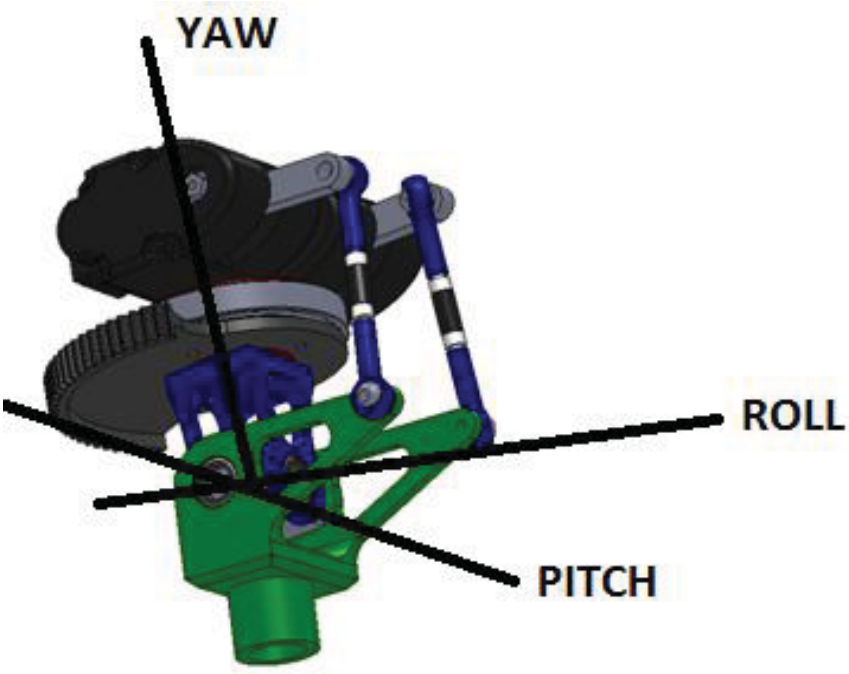


Figure 4-5: CAD design demonstrating roll, pitch and yaw of hip area

(Figure 4-5) above shows a 3 dimensional model of the hip are produced in Solidworks. This essentially represents a ball/socket joint producing 3 degrees of freedom. The yaw, roll and pitch axes can be seen in this figure. (Figure 4-6) shows the completed hip area of the humanoid robot.

Figure 4-6: Completed CNC'd hip area of the actual humanoid robot

Below in (Figure 4-7) is the part in the hip which produces the pitch axis. This, like most parts, is machined from a single piece of aluminium making the component a lot more accurate.

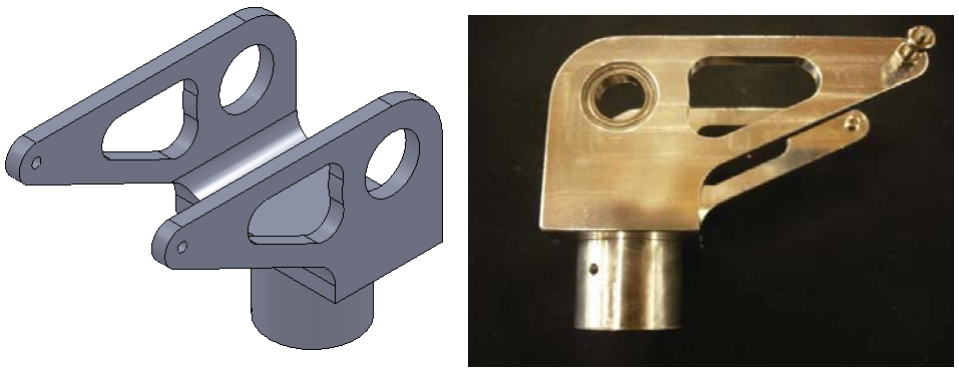


Figure 4-7: CAD design of hip component next to completed CNC'd part

4.3 *Knee*

The beauty of the WCR servos is the fact that they can be mounted at both the top and centre of the servo. The ability to be mounted in the centre allows for a much more compact knee area because of the reduction of added bearing supports that are usually needed on a standard servo.

The structure is comprised of two servos sandwiching the knee cap forming the upper portion of the knee. The lower knee is a Y shaped billet machined part, which mounts into the centre of both upper servos. This also acts as a mounting point for the two ankle actuator servos which are also sandwiched into place with a help of custom made, high tensile, 100mm long bolts. These bolts were custom made from bicycle spokes as it was found that they were not readily available.

(Figure 4-8) below shows both the upper and lower servo positions and the implementation of the knee cap to enable simple height control. The left image is of the actual knee demonstrating how clean and compact the design is.

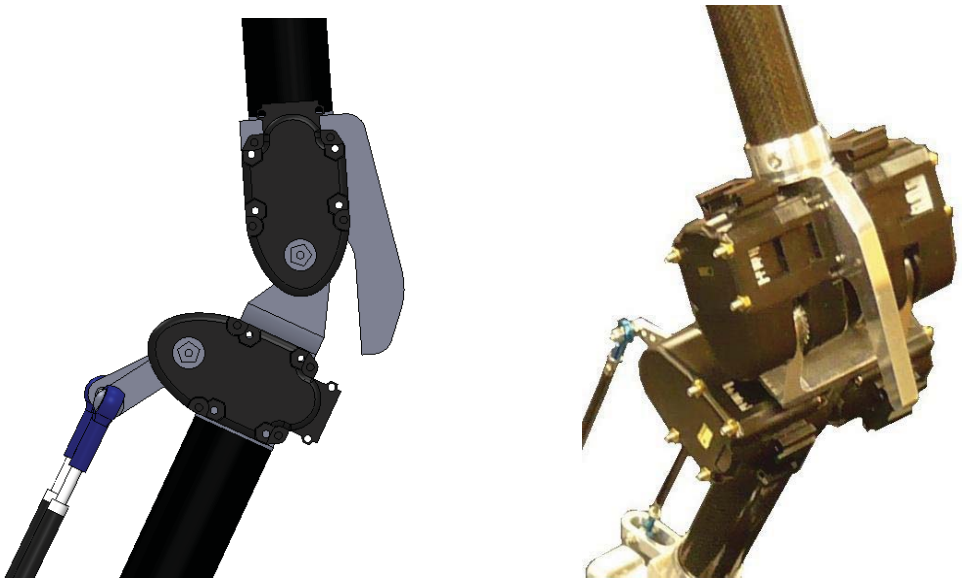


Figure 4-8: CAD design of knee joint next to completed knee joint of the humanoid robot

(Figure 4-9) is the most complex and time consuming part produced in this project. Even after optimising the component for machining, i.e. cutter depths and mounting points etc, it still took 8 hours to machine. This part is the knee joint which houses the two ankle servo modules and also acts as a fixing point for the knee servo module.

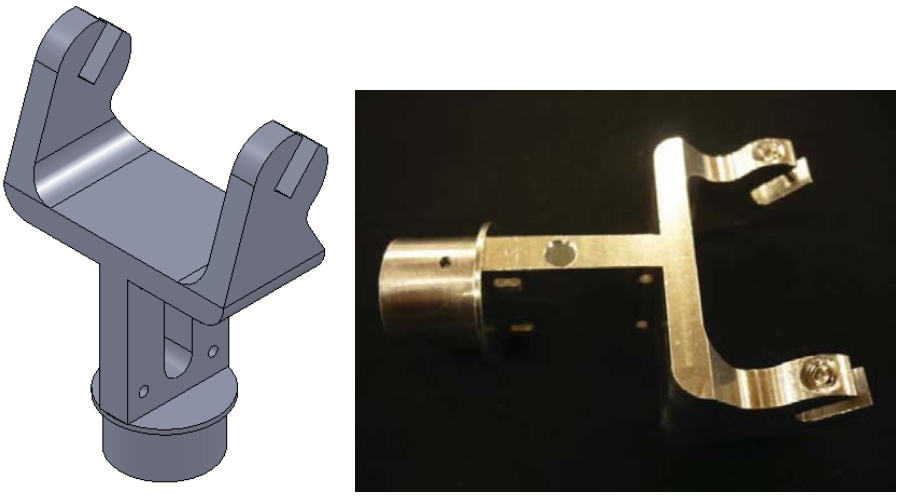


Figure 4-9: Knee component used to house servo modules for ankle actuation

4.4 *Ankle, Foot and Toe*

All bipeds to date lack the complexity of a real human foot. The human foot is very advanced consisting of 28 bones, 20 muscles, and approximately 7800 nerves. The fact that there are 7800 nerves for force sensing, which is in turn used for balance, implies that the foot is a very important part of a humanoid robot [32].

Corrective movements are made at both the hip and ankle joints. Some postural corrections are made through articulation of individual toes. With a flat foot the degrees of freedom (DOF), are limited to only the ankle which reduces stride length. With a toe joint a stance with both feet on the ground can be maintained for a longer time allowing the COG to be transferred onto the leading foot. This toe arrangement allows the humanoid to walk more elegantly, like a human, as opposed to the more common shuffle type of walking [14].

The human foot is so complex that it would be almost impossible to develop a mechanically operated foot with the same flexibility, strength, and size as the real one. A singular splitting foot, one toe, is sufficient for the development of a low cost humanoid with a highly

intelligent gait process [28] [17]. (Figure 4-10) shows the articulation of the single splitting foot.

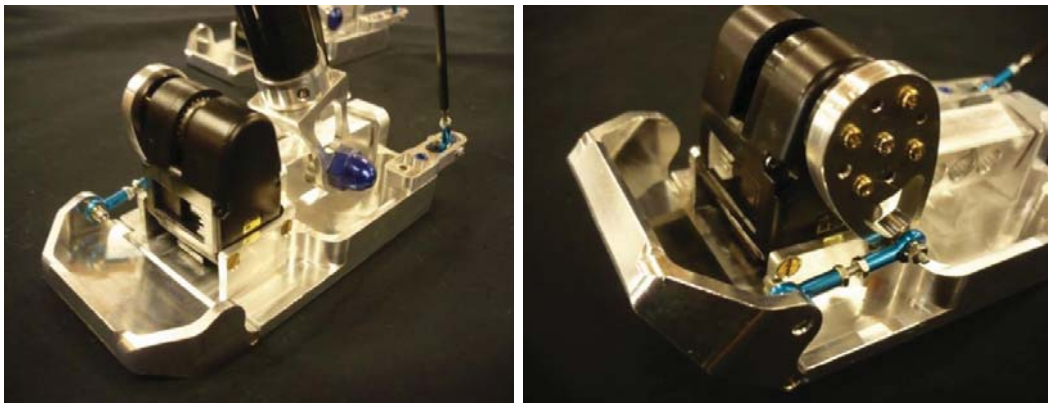


Figure 4-10: Articulation of the single splitting toe

The foot is actuated the same way as the hip is, minus the yaw axis. The fact that the actuators for the roll and pitch of the foot are situated up near the knee makes for a very lightweight clean foot design. This also allows more room for placement of the toe actuator which in most classical design cases would be in the way of the roll actuator.

Initially the toe joint was going to be comprised of thin spring steel mounted to both the foot and the toe. This was to produce a spring affect to reduce the torque required to support the robot on its toes. It was decided however that this would not be as reliable as an actual hinge joint which could have a spring added in the future if required.

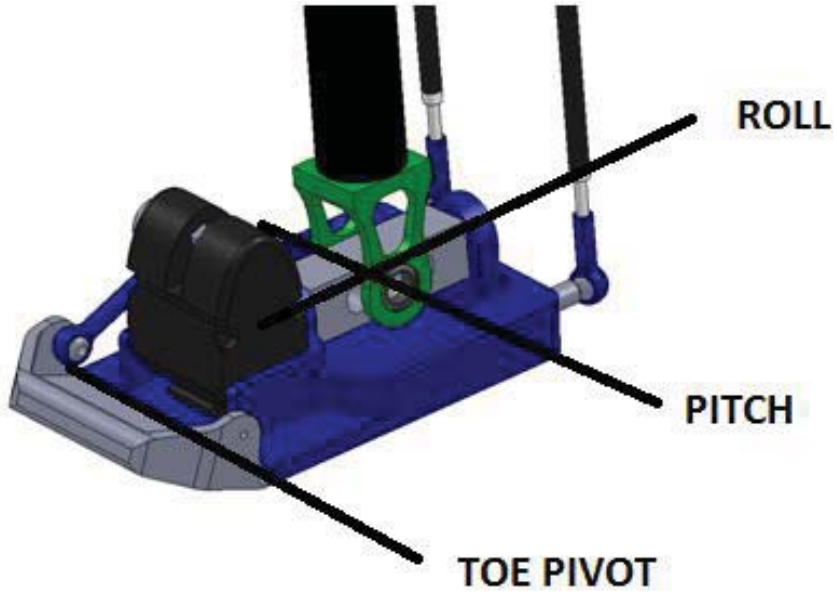


Figure 4-11: CAD design demonstrating pitch and roll of ankle area

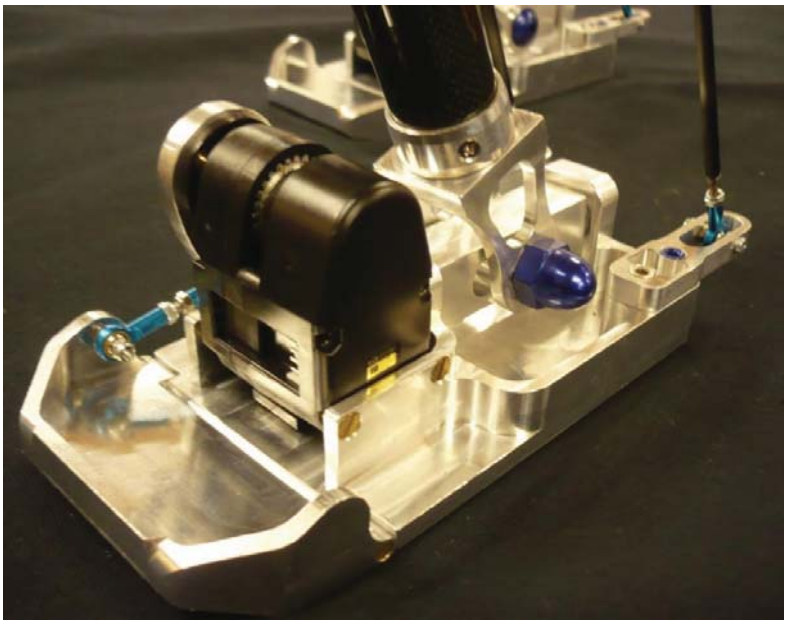


Figure 4-12: Completed foot and ankle area of humanoid robot

(Figure 4-11) and (Figure 4-12) show the CAD model and final construction of the foot and ankle area. This area is out of proportion with respect to a human being but is done as it gives added balance to the robot due to larger foot surface area.

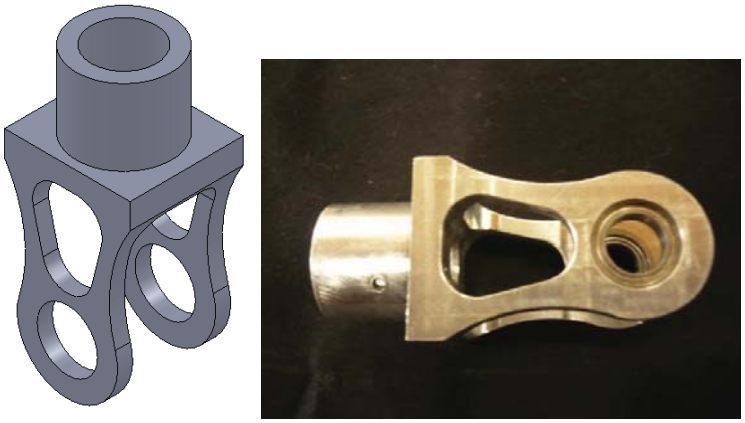


Figure 4-13: CAD design and actual machined part of lower ankle pitch joint

Refer to Appendix D for workshop drawings of components that were machined on a regular milling machine.

4.5 *Bearings*

The joints, especially hips and ankles, need to have as little slop as possible. The amount of slop in a joint is greatly magnified over the length of a limb. This cannot be completely eliminated but it can be greatly reduced by implementing ball bearing races in each joint. The manufacturing time and cost increases considerably, but the precision and life expectancy of the joint increases. Usually ball bearings have a large amount of slop axially. This means dual bearings must be used on a shaft and spaced sufficiently to stop the slop. This in some cases can be impractical, for example in a hip joint where space is very scarce. In this case a large diameter low profile bearing can be used. A larger diameter bearing does not necessarily have less slop, but due to the larger diameter will reduce the angle the slop produces.

4.6 *Structural Material*

For a commercially produced product aluminium billet machining is not an ideal means of producing a part. It is time consuming, very costly, and very wasteful of material. However, in the case of a prototype, aluminium is an idea material as it machines very easily, cleanly, and precisely, whereas something like PVC plastic tends to result in a warped, furry edged product.

4.6.1 Carbon fibre legs

Carbon fibre tubing was used in the leg section for several reasons. In a structural sense carbon fibre is extremely lightweight and strong so is therefore appealing for such an

application. In the case of this humanoid though, almost any material would be structurally sound, but carbon fibre tubing was used because there is an extremely large selection of various sized diameter, and length, tubing in the hobby industry. Whereas, although aluminium tubing can be a lot easier to find when it comes to thin walled sections, usually large lengths must be purchased.

Tubing was selected as it would produce a path for wiring of servos to follow in order to produce a much cleaner leg. Also tubing or just sectioning in general, enables the ability for each section length to easily be modified.

Carbon fibre is an easy material to work with except when fixing the carbon to another material such as an aluminium bracket. In this case the tubing would be mounted at the top and bottom to an ankle, hip or knee joint where made of aluminium. [28] developed a biped robot called Spring Flamingo which used carbon fibre tubing. They determined tubing must be supported equally internally, and externally, to distribute the load equally and prevent the tube from collapsing or splitting . To achieve this, the aluminium part that the tubing is mounting to has a boss that inserts into the tube. A retaining ring is pressed on the tubing externally, essentially sandwiching the carbon fibre between aluminium. (Figure 4-14) below demonstrates the external aluminium rings.

Figure 4-14: Carbon fibre leg joint with aluminium retaining rings

4.7 *Finite Element Analysis (FEA)*

FEA testing was performed on several parts to ensure that their factor of safety, (FOS), was sufficient. The strength of these parts is far greater than required and, in the case of failure, human safety is not at risk so this area does not need to be stressed. This robot is only a prototype and is therefore stronger and heavier then desired, but as research continues parts can be refined and tested producing a more robust lightweight part.

All components that were tested had a safety factor far greater than 1. Instead the FEA was used not to determine a failure rate but to determine where stress points were located. This information was then used to modify the design to either move or eliminate the stress point producing a more desirable design. Doing this showed the effect of wall thicknesses, and corner radius's, on the overall characteristics of the model. This would also help in future designs.

Figure 4-15: FEA test of pelvis part showing higher stress in the red area

(Figure 4-15) above is an illustration of the force distribution on the pelvis. This is a redefined model as it became apparent that undesirable forces were being focused at the hip support areas. A few modifications to wall thicknesses and wall placements were done and as a result a very evenly distributed force was achieved. It can be observed that there is a very high force, the red portion, in the centre of the pelvic. This however is not structural, instead it is just used as a separation for bolt holes for support of servos.

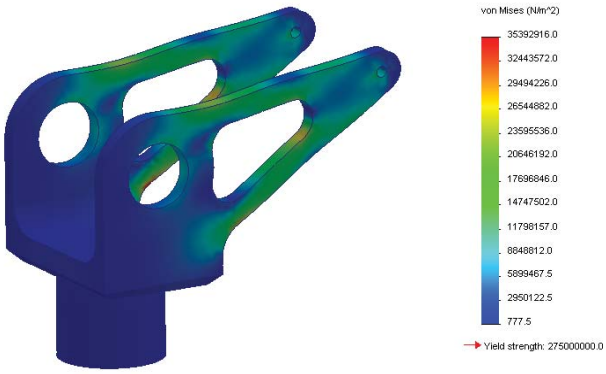


Figure 4-16: FEA test of hip pitch joint showing a very equal force distribution

(Figure 4-16) shows the stress distribution on the lever portion of the hip assembly. It can be seen that the force is very well distributed as there is a constant change in colour as opposed to a high stress point which would jump up in colour. The lower stress areas are in blue and the higher in red.

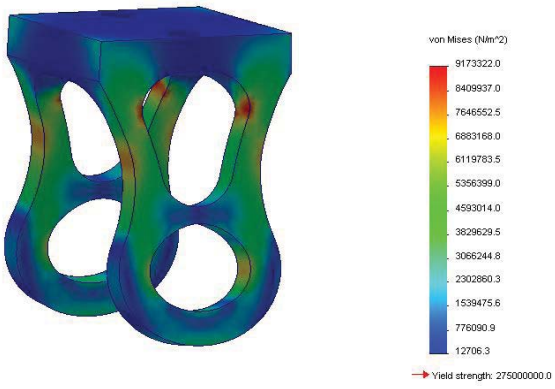


Figure 4-17: FEA test of hip roll joint showing high stress areas in red

(Figure 4-17) shows the hip and ankle joint. This joint takes most of the weight of the robot and is subjected to larger shock forces than other parts. In red are the high stress areas of the part, which in this case are situated at the low radius areas by the thinnest part of the wall. These radiuses were increased which reduced this stress point significantly. Even with a high stress point however this part had a far greater safety factor than realistically necessary.

4.8 *Weight Properties*

In order to calculate and manipulate the inertia of the overall system it is imperative to acquire mass properties of individual parts. (Table 4-1) shows the weight properties of each part. Obviously the mass distribution of individual parts will vary depending on their size and shape. In the case of this research project the exact weight distribution is not entirely necessary and only a basic understanding is needed. Therefore each mass is assumed to be situated in the centre of the part.

Part Name	No. Of Parts Required	Weight (g)	Total Weight (g)
Pelvis Yaw	1	26.27	26.27
Pelvis Roll	1	20.28	20.28
Pelvis Main	1	38.47	38.47
Hip Yaw	2	17.35	34.7
Hip Roll	2	4.87	9.74
Hip Pitch	2	12.27	24.54
Hip Roll & Pitch Connector	2	5.20	10.2
Hip Yaw Servo Mount	1	9.70	9.70
Knee Upper	2	9.94	19.88
Knee Lower	2	13.45	26.90
Ankle Pitch	2	7.64	15.28
Ankle Pitch and Foot Connector	2	8.61	17.22
Foot	2	45.79	91.58
Toe	2	12.18	24.36

Table 4-1: Mass properties of individual components used in inertia calculations

4.9 *Computer Aided Machining*

Computer Notation Control, or more commonly known, CNC machining was one of the most important parts of manufacturing this prototype humanoid. Firstly the CAD design is imported into a Computer Aided Machining (CAM) software where a user defined machining path is defined. It is a common misconception that the CAM software will convert all CAD data into usable machine code for you without much human interaction. In fact the user must define every machining path, tool, tool depth, tool feed rate, spindle speed, safety heights, off set, finishing cut, start and finish points of every single cut that takes place. Once this is done the CAM software will produce a G-code which is a machine code that can be read by the CNC machine.

Every part, except for the carbon fibre rings and the pelvis yaw axis, was machined on the CNC machine. This, along with the high precision and intricacy of each part, meant a lot of machining time was required. A lot of the parts are very small, thin walled, and didn't have any adequate positions for clamping which made for a very interesting machining process. A lot of the components, such as the hip and ankle joints, could not actually be produced on the standard 3 axes CNC machine. A mechanical 4^{th} axis with a Dividing Head was implemented on the Massey University CNC machine solely for this project. This enabled parts to be mounted horizontally in a standard chuck which could then be rotated to any angle the user desired. This could not be done on the fly like a dedicated controlled 4^{th} axis could but it still meant that parts that required 90 or 180 degree rotation multiple times could easily be achieved. (Figure 4-18) below on the left shows the operations performed to machine the pelvis. (Figure 4-18) on the right shows the profile path in orange that tool number 16 will take to machine out the bearing pocket for the hips.

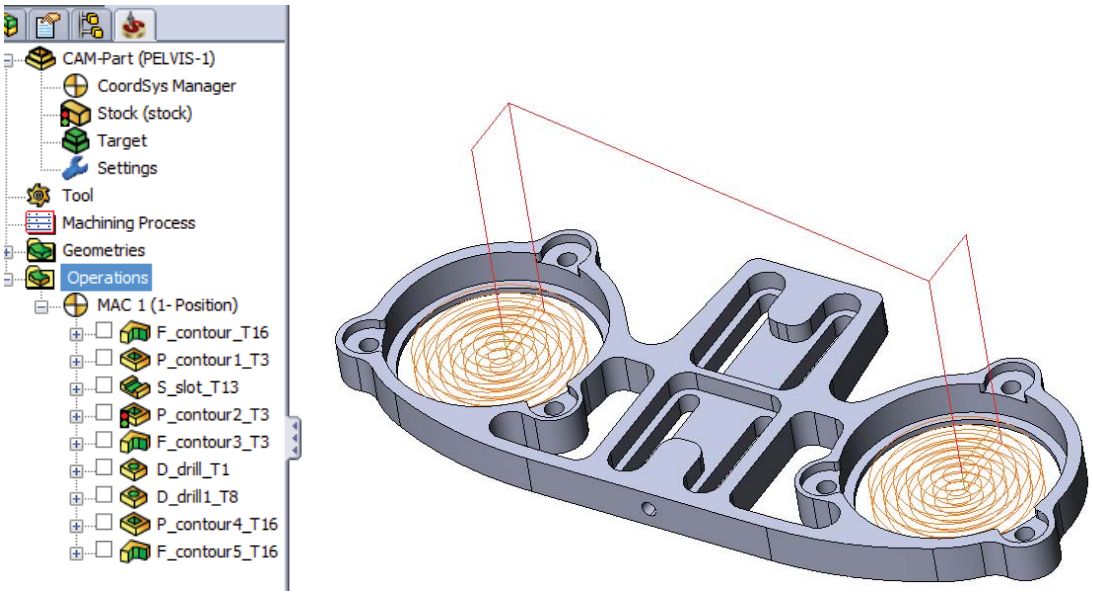


Figure 4-18: Computer Aided Machining (CAM) operations and tool paths

SolidWorks CAM software has a feature that allows a 3-D simulation of the entire machining process. This allows the user to ensure that they have produced the correct operations. For example it is very common to accidentally make the tool follow the incorrect side of the profile path, essentially machining away the part that you want to be keeping. (Figure 4-19) shows how the entire process is simulated in a 3-D Solid representation.

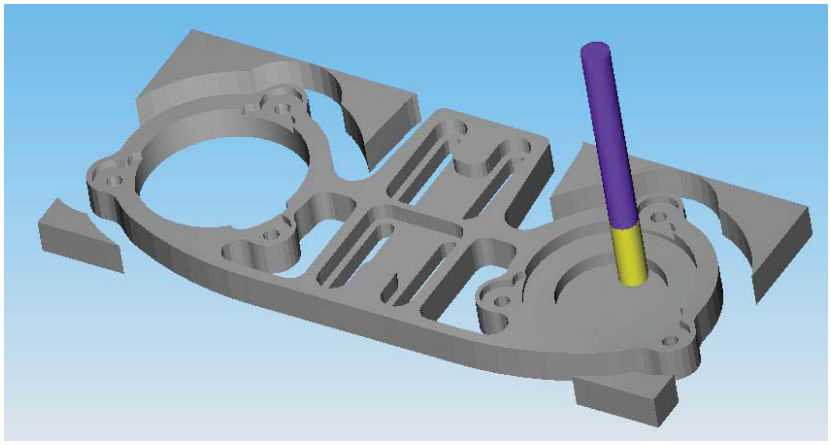


Figure 4-19: 3D representation of the machining path in CAM software

```
O5000 /* DSF.CNC */
/* 11-JAN-2011*/
G90 G17
G80 G49 G40
G54
G91 G28 Z0
G90
M01
N1 M6 T16
T16
/* TOOL -16- MILL DIA 8.0 R0. MM */
G90 G00 G40 G54
G43 H16 D16 G0 X35.294 Y37.985 Z70. S3000 M3
M8
/*----------------------*/
/*P-CONTOUR4-T16 - POCKET*/
/*----------------------*/
  X35.294 Y37.985 Z50.
  Z0.
G1 Z-6. F100
G3 X35.294 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X35.294 Y40.785 I0. J-5.5
G1 Y43.585
G3 X35.294 Y43.585 I0. J-8.3
G1 Y46.385
G3 X35.294 Y46.385 I0. J-11.1
G1 Y49.185
G3 X35.294 Y49.185 I0. J-13.9
G1 Y51.985
G3 X35.294 Y51.985 I0. J-16.7
G1 Y54.785
G3 X35.294 Y54.785 I0. J-19.5
G0 Z50.
  Y37.985
```

Above is a snippet of G-Code that is produced by the CAM software. This particular snippet is the bearing pocket for the hips. In bold it shows the settings that must always be checked on the CNC machine before commencing operation. It is essential to ensure that the correct tool, spindle speed, and feed rate are being used.

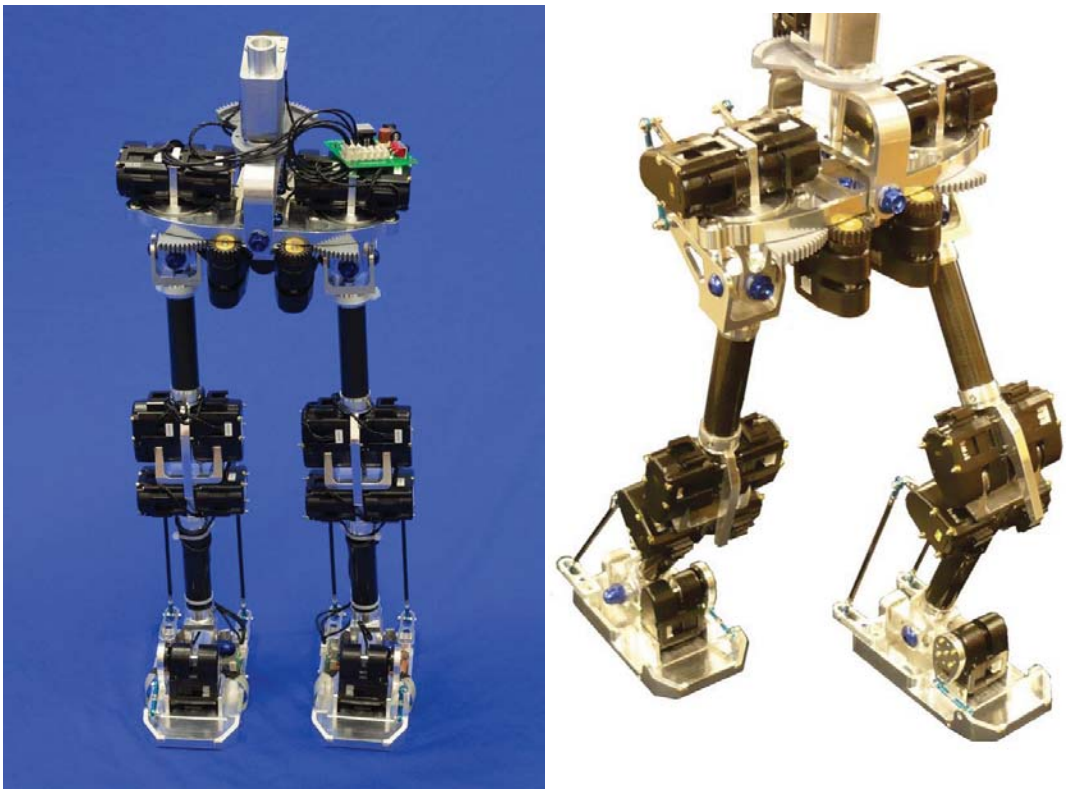


Figure 4-20: Completed lower half of the humanoid robot MURPH

Figure 4-20 show the completed lower half of the humanoid robot named MURPH. MURPH is an acronym for Massey University Robotics, Pushrod controlled Humanoid.

4.10 *Discussion and conclusions*

This humanoid robot is very complex in comparison to similar priced humanoids. The need for such intricacy became apparent early on in the project as it became obvious that low budget 'classic' designs lack the agility of their human counterparts. To avoid the issues plagued by 'classic' designs, this humanoid was designed and built to exploit as many mechanical factors as possible. The physical structure of the humanoid is built as lightweight and as agile as possible, consisting of high tensile aluminium, carbon fibre tube, and carbon impregnated plastics.

Solidworks is a very advanced 3D modelling program which allows the user to test designs to ensure their dimensions are correct. The software is also used to establish weight properties, finite element analysis can be performed to determine high stress areas of parts,

and, most importantly, these designed parts can then be imported into a computer aided machining software which is used to determine machining profiles and tool paths for the CNC machine.

Due to a majority of the components being very lightweight their wall thicknesses were extremely thin. This produces a problem when machining, as aluminium flexes when not supported sufficiently, and this becomes an issue for fitting bearings. A lot of the parts that required machining from a top plane and a sagittal plan had to be machined on a 4^{th} axis as they did not have sufficient positions for clamping. With a addition of a 4^{th} axis, very unusually shaped parts could be machined, but a very high degree of accuracy still maintained which is usually not possible when parts need to be manoeuvred for re machining.

All joints contain roller ball bearings races. This is to reduce friction allowing the servo to perform to its full potential but, most importantly, it produces a slop free joint which is very important in a humanoid. A very small amount of slop in a joint gets significantly magnified by the end of the manipulator and would greatly increase the complexity of the control system as it would struggle to perform accurate foot placement.

High modulus carbon fibre is used in the legs and pushrods. This is to enable easy modification of leg lengths which may be necessary in the development stage as this humanoid is only the first prototype.

To conclude, a complete humanoid robot with revolutionary joint actuation and placement has been designed. The lower half up to the back bone has been completely fabricated from scratch, including 90 percent of components CNC machined from billets of material. The components are machined and constructed to a high accuracy to ensure the lowest amount of slop in each joint which helps reduce the complexity to control algorithms. The mechanical structure is more compact than regular systems due to the implementation of push rods for actuation, and gear drives for optimal module placement. The effect of a compact system is apparent in the control system as momentum and rotational inertia becomes a large factor in the agility of the humanoid.

5 Electronic System

Actuators, sensors and the controller form the core of the electronic system which is detailed in the following sub-sections.

5.1 *Actuators*

5.1.1 Servo-actuator

There are two types of readily available servos for hobby robotics. The classic PWM controlled servo, and the new serial controlled servos. The serial controlled servo means they can be linked in one long daisy chain so there is no need for individual PWM or driver circuits.

The serial controlled servos are much more desirable due to their easier control and built in features. There are two servos to choose from that meet torque specifications and are affordable. The WCK 1111 and the AX-12 are pretty similar with good torque and priced at about $55 each.

They both have a nice network protocol. The WCK1111 is similar to AX-12 but with extended functionalities. It has full PID control so they are very quiet and don't have the noise issue that sometimes plagues worn AX-12s. PID also lets you tune for an elastic feel, which is desirable for a humanoid limb as exact positioning is not ideal.

They have a nice dynamic brake feature which is a third power state. ‘Holding Position’, ’Torque’ and ‘Free Motion’, just like AX-12 and the third Dynamic Brake mode where they are in between the two. In this mode there is no power consumption but they still offer substantial amount of strength, although with some force you can move them. Torque feedback works better on Robobuilder WCK1111 servos than on AX-12. However it is still not very accurate.

The main issue with Robobuilder servos are gears breaking. The WCK 1111 already come with 2 metal gears and some of the new servos come with the 4th gear, the main one, in metal too. The only gear that needs replacing is gear number 1 which then makes for a full

metal gear box eliminating gear breakage and upping the maximum torque from 11kg/cm to 14.5kg/cm.

Something as basic as an Arduino or 8051 micro can be used to operate Robobuilder WCK servos, whereas control for AX-12 is a little more complicated as you need to split the single control pin into standard TX RX control lines. Robotis AX-12 is still the leader in the servo area. They are seen as the best quality and the best product if you're into the "real deal". Robobuilder is portrayed as a "cheaper" alternative to the AX-12.
Below in (Table 5-1) shows the specification characteristics of the WCK1111 module

WCK 1111	Specifications
Communication	Multi drop Full Duplex UART serial communication
Baud Rate	4,800bps ~ 921,600bps(8 levels)
Extension	Max 254 modules per channel(ID 0~253)
Operating Voltage	6VDC ~ 10VDC
Speed	Max No Load Speed 0.15 sec/60°
Stall Torque	Max 13kg·cm
Max Power	1.1W
Gear Ratio	1/241
Control Mode	Position Control, Speed Control, Torque Control
Control Angle	0~254(Standard Resolution), 0~1,022(High Resolution)
Operating Angle	0°~269°(Standard Resolution), 0°~333°(High Resolution)
Resolution	8 bit/1.055°(Standard Rosolution), 10 bit/0.325°(High Resolution)
Error Range	±0.8°(Standard Resolution)
Speed Level	30 levels(Position Control Mode), 16 levels(Wheel Mode)
Case Material	Engineering Plastic
Gear Material	POM+Metal
Size	51.6 mm x 27.1 mm x 36.4 mm
Weight	55g

Table 5-1: WCK1111 module specifications

Since these modules are full duplex there are 4 pins. 1 Ground, Vcc, a Transmit and a Receive. Figure 5-1 shows the pin allocation of the servo modules.

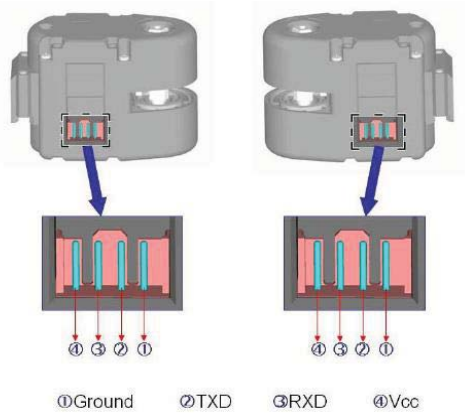


Figure 5-1: Pin representation of WCK module. [42] P-21

5.1.1.1 Protocol

The WCK module adopted a dedicated communication protocol named “Multi drop Full Duplex UART two way serial communication protocol”. Basically the communication is carried out in a repeated cycle of the controller sending a command packet to the WCK module, and then the WCK module sending back a response packet. (Figure 5-2) shows the basic send receive protocol between a module and computer.

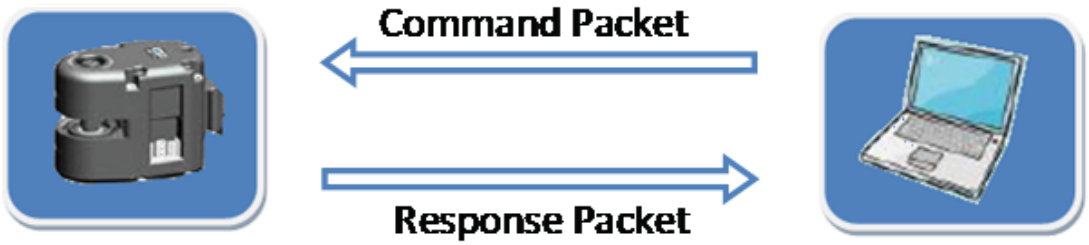


Figure 5-2: WCK module protocol representation

When the user requires communication directly from a PC, or aftermarket controller, an additional circuit is needed between the controller and module. This is because the module protocol is at TTL voltage level, whereas a signal from a RS232 is at serial voltages. (Figure 5-3) illustrates where the additional circuit must be implemented into the system.

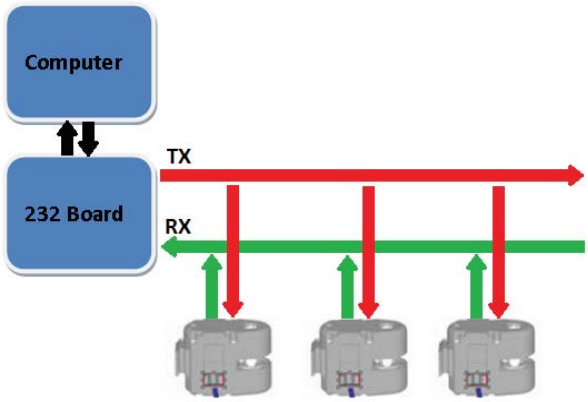


Figure 5-3: WCK module to PC protocol though additional 232 board

The additional circuit shifts the TX and RX signals to a usable voltage. Below is the circuit diagram of the circuit which uses a ST232 IC which is similar to the MAX232. There is a limit to the number of servos that can be driven from this circuit as it only has a very low drive current. If more than 5 modules are connected in series the circuit will saturate and the user will be unable to drive any modules. (Figure 5-4) shows the circuit diagram of the board produced using the ST232, similar to the MAX232, for driving servo modules from a computer.

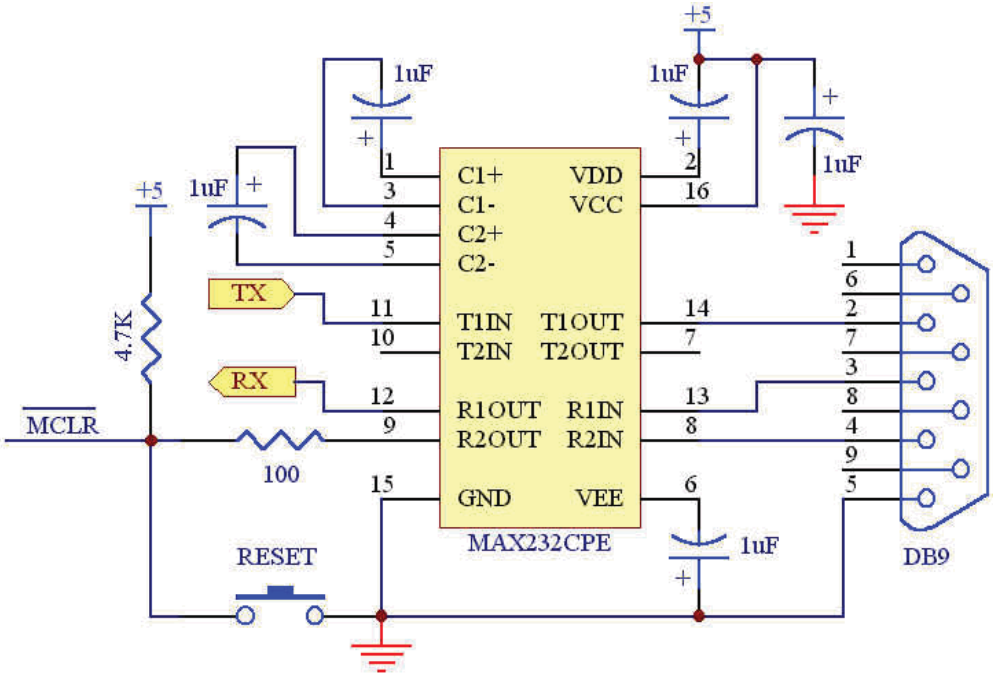


Figure 5-4: MAX232 schematic diagram

5.1.1.2 Positioning

The position control of the WCK module is determined by a packet sent by the user. Once the correct packet is sent the module will move to the specified position. Even though these servos can perform 360 degree rotation they have a dead spot where the servo cannot stop at. The degree of usable rotation depends on the resolution the command packet requests. 8 Bit is low resolution and allows the servo to move 269 degrees whereas in 10 Bit high resolution the servo can move 333 degrees. (Figure 5-5) shows on the left the degree of movement a standard resolution command will achieve and on the right shows a high resolution command.

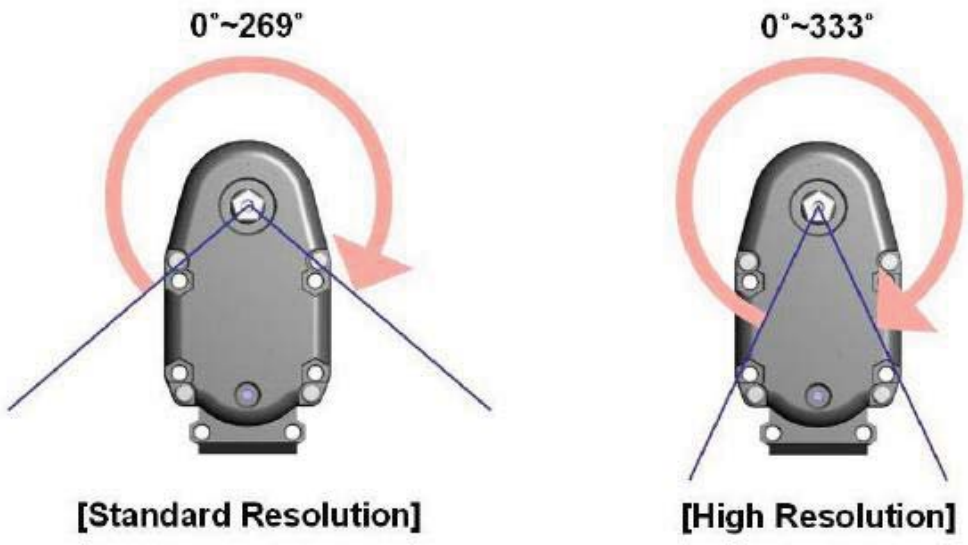


Figure 5-5: WCK module operation angles. [42] P-41

The Control Angle is the angle that is used by the controller to determine which position to move to. The Movement angle is the actual physical angle the servo moves to in degrees. The Unit Angle is the physical displacement of rotation angle caused by executing a single unit of Control Angle. (Table 5-2) shows the angle characteristics of the WCK1111 module

Control Angle	0~254(Standard Resolution) 0~1,022(High Resolution)
Movement Angle	0°~269°(Standard Resolution) 0°~333°(High Resolution)
Unit Angle	269°/255=1.055° (Standard Resolution) 333°/1023=0.325° (High Resolution)
Control Angle	±0.8°(Standard Resolution)

Table 5-2: WCK module operation angle characteristics

5.1.1.2.1 *Low resolution*

In order to move the actuator in low resolution mode a 4Byte packet is sent to the TX pin on the module. This is regarded as 8 Bit communication as the target packet uses all 8 bits of the target byte.

Command Packet

1 Byte	1 Byte	1 Byte	1 Byte
Header	**Data 1**	**Target 1**	**Checksum**

- Header = 0xFF or 255
- Data1 =

Torque			ID				
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**

- Target1 = 0 to 255
- Checksum = (Data1 XOR Target position) AND 0x7F

Response Packet

1 Byte	1 Byte
Load	Position

Load = 0 to 254 , Position = 0 to 254

Example Packet

Below is an example of the serial packet that must be received by module ID 3 to a target position 200.

1 Byte	1 Byte	1 Byte	1 Byte
FF	03	C8	4B

High resolution

High 10-Bit resolution mode increases the degree of operational movement and also increases the precision.

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
Header	Data1	Data2	Data3	Data4	Data5	Data6	Checksum

- Header = 0xFF
- Data1

Mode (=7)			Arbitrary Value				
7	6	5	4	3	2	1	0

- Data2 = 0xc8
- Data3 = ID(0 to 253)
- Data4 = Torque(0 to 254)
- Data5 = target position(Upper(H3) 3 bit)

			X		Position (H3)		
7	6	5	4	3	2	1	0

- Data6 = target position(Lower(L7) 7 bit)

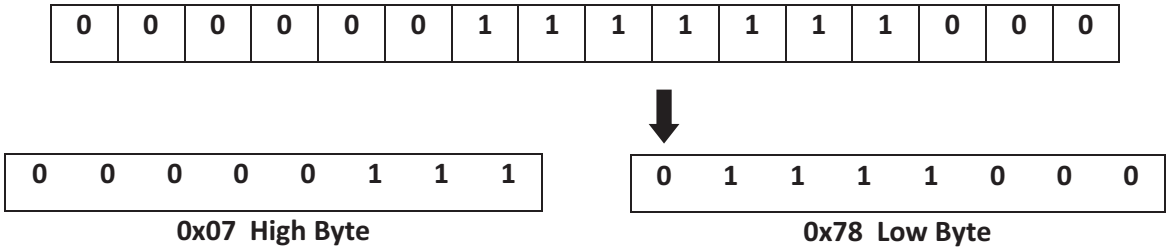
Position (L7)							X
7	6	5	4	3	2	1	0

- Check sum = (byte2 XOR byte3 XOR byte4XOR byte5 XOR byte6 XOR byte7) AND 0x7F

Above is the method proposed by the WCK user manual. This data sheet is not entirely correct resulting in a lot of confusion when it comes to calculating Data5 and Data 6. The way that the lower byte is acquired is misleading, as it says bit 0 is ignored and just to use the upper 7 bits when in fact this is not the case at all. Instead the entire high byte is shifted to the left by one bit and bit 8 of the lower byte is replaced with a 0. Below is an example for acquiring the low and high byte for the high resolution value 1016.

Example:

Below is the binary representation of the value 1016.



The arrow above shows the addition of a 0 after shifting the high bytes to the left by one bit. Below is the packet that must be sent to move the servo to position 1016.

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xE0	0xC8	0x03	0x00	07	78	0x54

5.1.1.3 PID tuning

The fact that these servos have PID control is what makes them unique. PID control allows the user to tune the actuator to more of an elastic feel which in a humanoid application is preferred [29] (Figure 5-6) (Figure 5-7) (Figure 5-8) show the response characteristics after the addition of P, I and D respectively. It can be seen that with just P implemented there is large overshoot and a very long settling time, whereas with PI and D there is no overshoot and still a very high acceleration.

(Figure 5-6) shows the response characteristics when Proportional gain control is added.

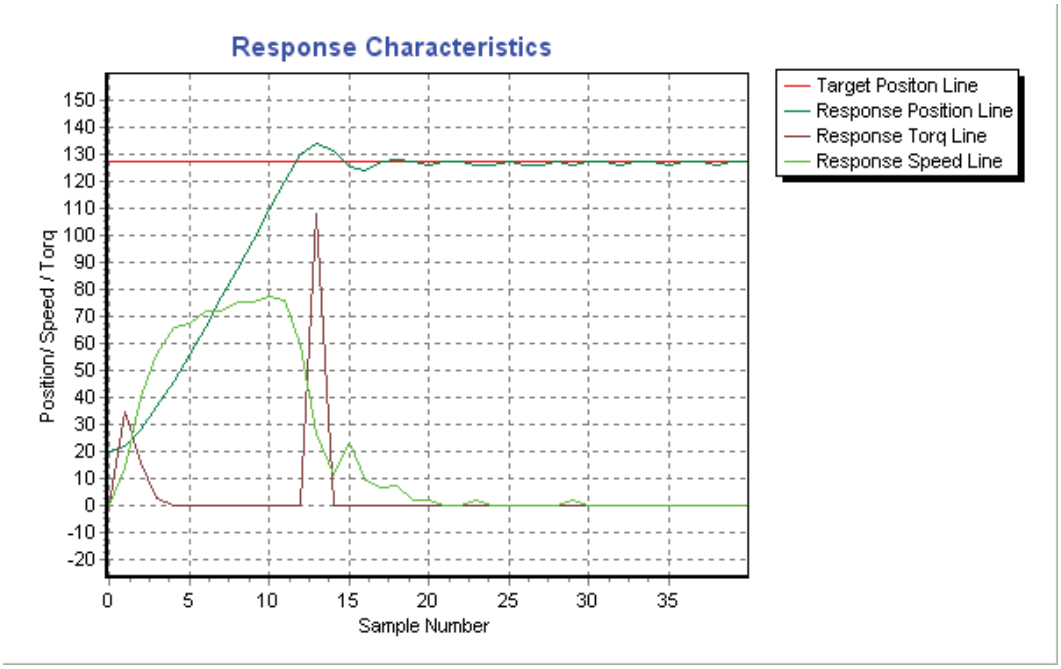


Figure 5-6: Module characteristics with P gain. [42] P-29

(Figure 5-7) shows the response characteristics when Proportional and Derivative gain control is added.

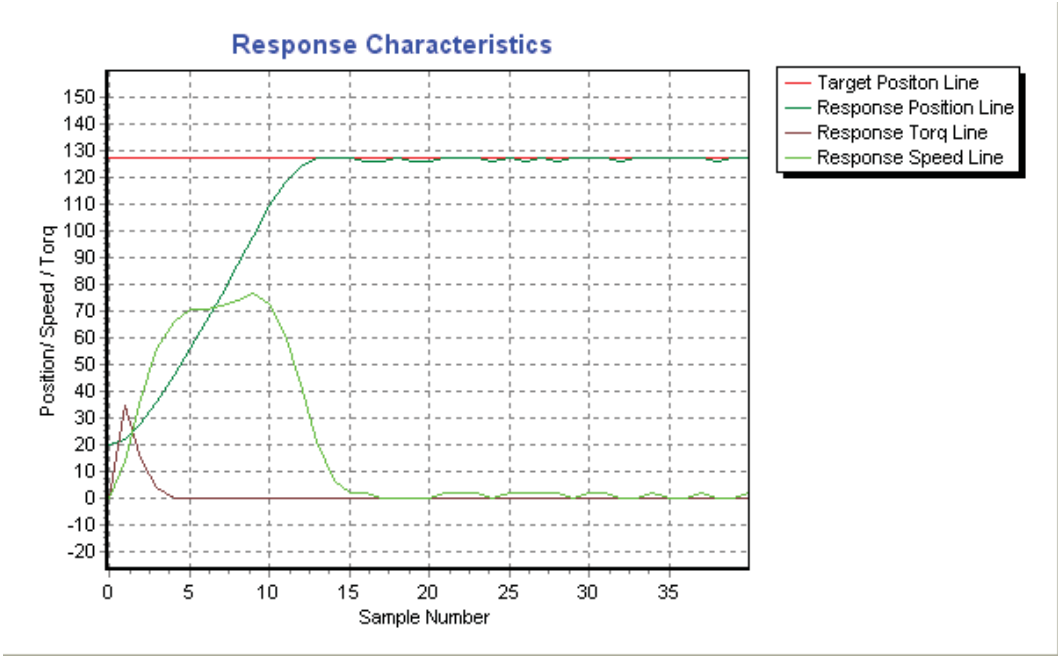


Figure 5-7: Module characteristics with P D gain. [42] P-29

(Figure 5-8) shows the response characteristics when Proportional Integral and Derivative control is added.

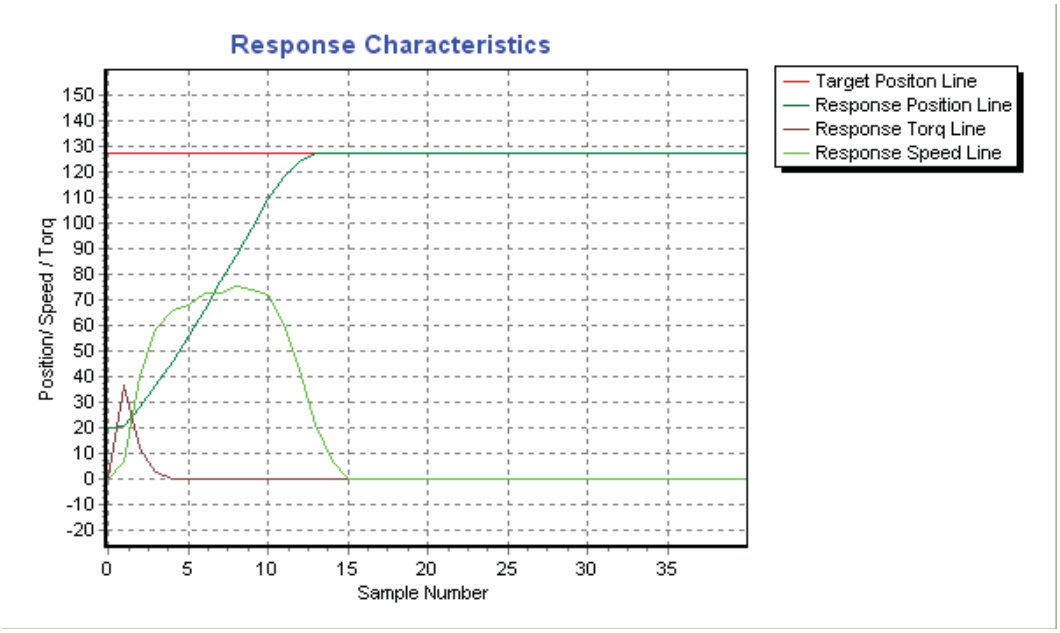


Figure 5-8: Module characteristics with PID gain. [42] P-30

With PID implemented the torque is greatly reduced as there is not overshoot resulting in sudden direction changes. This reduces the overall power consumption of these modules.

To set the P, I and D the user must send a set command similar to the position command. It's recommended that users set P gain and D gain according to the formula of P gain = 1.5 x D gain.

Command Packet for setting P and D

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
Header	Data1	Data2	Data3	Data4	Checksum

- Header = 0xFF
- Data1

Mode (=7)			Arbitrary Value				
7	6	5	4	3	2	1	0

- Data2 = 0x09
- Data3 = new P-gain (1~254 recommended)
- Data4 = new D-gain (0~254 recommended)
- Checksum = (Data1 XOR Data2 XOR Data3 XOR Data4) AND 0x7F

Example: ID 0, P gain = 100, $\mathrm{D\ gain} = 100$

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xE0	0x09	0x64	0x64	0x69

Command Packet for setting I

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
Header	Data1	Data2	Data3	Data4	Checksum

- Header = 0xFF
- Data1

Mode (=7)			Arbitrary Value				
7	6	5	4	3	2	1	0

- Data2 = 0x16
- Data3 = new I gain(0~10 recommended)
- Data4 = Data3
- Checksum = (Data1 XOR Data2 XOR Data3 XOR Data4) AND 0x7F

Example: ID 0, I gain = 100

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xE0	0x16	0x00	0x00	0x76

5.1.1.4 Speed & Acceleration

The maximum speed and acceleration can be specified on these modules. The advantage of this feature is that once the speed and acceleration is set it is fixed at that value until altered again. This means that less programming and control needs to be specified at each command, instead only the position needs to be specified. The disadvantage however is that once the speed and acceleration do need to be changed a new packet must be sent first then a second for the position resulting in a delay.

Command Packet for setting Speed and Acceleration

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
Header	Data1	Data2	Data3	Data4	Checksum

- Header = 0xFF
- Data1

Mode (=7)			Arbitrary Value				
7	6	5	4	3	2	1	0

- Data2 = 0x0D
- Data3 = Speed(0~30)
- Data4 = Accel(20~100), Accel = acceleration range
- Checksum = (Data1 XOR Data2 XOR Data3 XOR Data4) AND 0x7F

Example: ID 0, Speed = 30, Acceleration=100

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xE0	0x0D	0x1E	0x64	0x1D

5.1.1.5 Torque Overload

The torque overload feature is also unique to the WCK module. This feature ensures that the motor is not overloaded resulting in failure of one of the gears in the gearbox. This is basically a current monitoring feature which has a set point ranging from 400mA to 1.8A. Once this overload point is reached the power to the motor is cut and the motor freewheels for at least 0.5 seconds where it will then try and move to its next requested position.

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
Header	Data1	Data2	Data3	Data4	Checksum

- Header = 0xFF
- Data1

Mode (=7)			Arbitrary Value				
7	6	5	4	3	2	1	0

- Data2 = 0x0F
- Data3 = 33 to 199

No.	Data3	Current(mA)
1	33	400
2	44	500
3	56	600
4	68	700
5	80	800
6	92	900
7	104	1000
8	116	1100
9	128	1200
10	199	1800

- Data4 = Data3
- Checksum = (Data1 XOR Data2 XOR Data3 XOR Data4) AND 0x7F

Example: ID 0, Overload = 104 (1Amp)

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	**0xE0**	**0x0F**	**0x68**	**0x68**	**0x6F**

5.1.1.6 Continuous rotation

At present, continuous rotation of actuators is not needed. It may be necessary in the future however and these actuators allow this option. Since there is a dead spot the servos cannot be stopped at a particular location when in this mode, instead a continuous speed and direction can be set. This mode is called ‘wheel’ mode and the user has 16 speeds to select from.

Command Packet for setting 360 degree continuous rotation

1 Byte	1 Byte	1 Byte	1 Byte
Header	Data1	Data2	Checksum

- Header = 0xFF
- Data1

Mode (=7)			Arbitrary Value				
7	6	5	4	3	2	1	0

- Data2

Direction				Speed			
7	6	5	4	3	2	1	0

Direction : 3(CCW), 4(CW)

Speed : 0(stop), 1(min)~15(max)

- Checksum = (Data1 XOR Data2) AND 0x7F

Example: ID 0, revolve CCW at speed level 15

1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xC0	0x3F	0x7F

5.1.1.7 Baud Rate and Torque testing

Transmission through RS232 inevitably results in a delay between sending and receiving the command. If multiple modules are connected together the time delay is even greater. Usually this time delay is negligible but for a system that requires precise control

characteristics it is important to know the delay times so that the system can be designed accordingly. Below in (Table 5-3) is the corresponding delay time for sending 1 Byte and 1 Command (4 Bytes) for all baud rates that the module supports.

Baud Rate (bps)	1 Byte Transmission Time (μs)	1 Command (4 byte) Transmission Time(μs)
4800	2.083	8.333
9600	1.042	4.167
38400	0.260	1.042
57600	0.174	0.692
115200	0.087	0.347
230400	0.043	0.174
460800	0.022	0.087
921600	0.011	0.043

Table 5-3: WCK module transmission speeds

To ensure that the specified servo module torque was correct a real world test was performed. This consisted of a mock up of the knee to the same proportions that are proposed for the humanoid robot. The lower portion of the leg was fixed to a wooden support which had a horizontal slide that allowed the upper portion to move linearly. (Figure 5-9 and Figure 5-10) show the knee at full crouch and at full extension respectively. A weight was attached to the top of the leg and tests were done to determine the maximum torque the actuators could actually produce. The weight was gradually increased until the actuators stalled, the supply voltage was increased and the test was repeated.

Figure 5-9: Jig for physical torque test at full crouch

In actual operation at the development stage of the humanoid it will not be necessary to be in such a position. Instead the leg will be subjected to a maximum crouch only 30 percent of what it is actually capable of meaning that the maximum weight obtained by the tests could actually be a lot larger.

Figure 5-10: Jig for physical torque test at full extension

Table 5-4) and (Figure 5-11) show the test results obtained. In the data sheet it recommends running the actuators between 6-10volts, ideally at 8.4 volts. In this test a maximum supply voltage of 14 volts was supplied, this was only for a very short time as the modules did become quite hot. At 12 volts however the modules performed exceptionally as they did not get much hotter than running at 8.4 volts but were able to supply a far greater torque. It still remains to be seen how much the life expectancy is reduced by running at such a high voltage.

Supply Voltage (V)	Maximum Weigh Lifted (g)
6	675
7	690
8	700
9	720
10	750
11	810
12	855

Table 5-4: Results of maximum lifting weight at various voltages

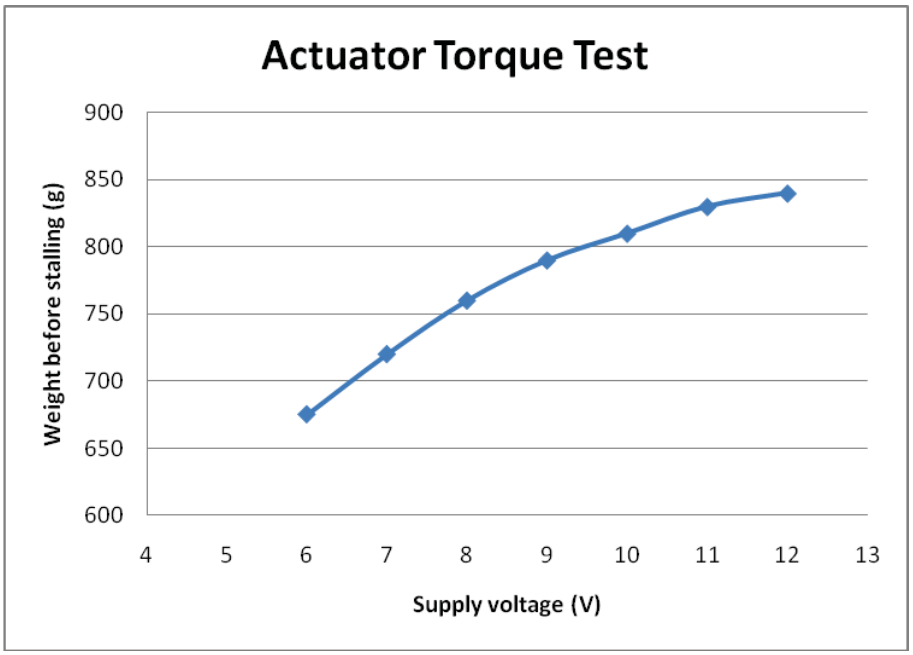


Figure 5-11: Graph of results from physical actuator torque test

It was observed that at lower voltages the difference in voltage had a greater effect on torque than up at the higher voltages.

5.2 *Sensors*

5.2.1 Attitude Heading Reference System

To monitor the position of the body and pelvis some form of sensor must be implemented. An Attitude Heading Reference System (AHRS) consists of 3 Gyroscopes, 3 Accelerometers and 3 Magnetometers. These systems are currently used in intelligent gait humanoids [33]. An on board processor calculates the rate of change with respect to time outputting an absolute position in Hexadecimal. The magnetometer is used as a horizontal reference to compute the rotational angle whereas the gravity is used for the Roll and Pitch. The AHRS is shown in (Figure 5-12)

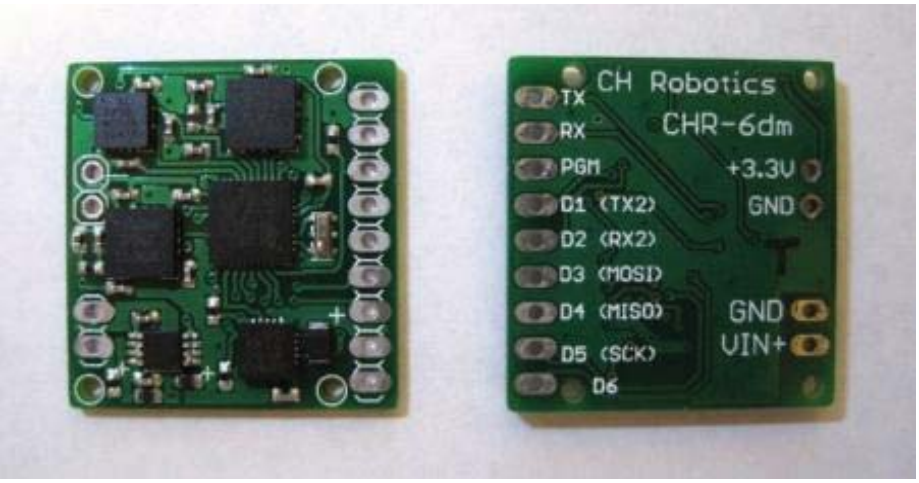


Figure 5-12: Top and Bottom view of the CH 6dm AHRS (www.chrobotics.com)

Communication with the CHR-6dm is performed over a TTL (3.3V) UART at 115200 Baud. The AHRS can be configured to transmit raw sensor in addition to angle estimates, and the transmission rate can be configured in 1 Hz increments from 20 Hz to 300 Hz. Alternatively, the CHR-6dm can operate in "silent mode," where data is transmitted only when specific requests are received over the UART. Regardless of the transmission mode and rate, internal angle estimates are maintained at over 500 Hz to ensure long-term accuracy.

A major disadvantage of using MEMS systems for navigation/positioning is that they typically suffer from drift. Drift is the expression for an accumulated error in the position calculations. Because the system is continually adding minute changes to its previously

calculated position, any errors in measurement, however small are accumulated resulting in a gradual drift effect.

To try and reduce the possibility of drift there are several algorithms and filters that can be implemented. The CH 6dm has a built in Extended Kalman Filter (EKF) which through statistical models, simulates the potential error in calculated value and actual value. EKF performance can be tuned by adjusting the process noise covariance matrix and the measurement noise covariance matrices (there are two measurement noise matrices - one for the accelerometers, and one for the magnetometer).

The process noise matrix is used to specify how much the EKF "trusts" data from the gyros with respect to data from the magnetic sensors and accelerometers. The lower the values along the diagonal of the matrix, the more the rate gyros are trusted. Conversely, if the diagonal elements are large, the gyros are trusted less and the accelerometers and magnetometers are weighted more heavily.

The CH 6dm has full calibration capabilities of accelerometers, gyroscopes, and the magnetometer. This is necessary when the sensor is not mounted exactly as desired. (Figure 5-13) show the user interface which allows for calibration and displays rate gyro, accelerometer, and angle values.

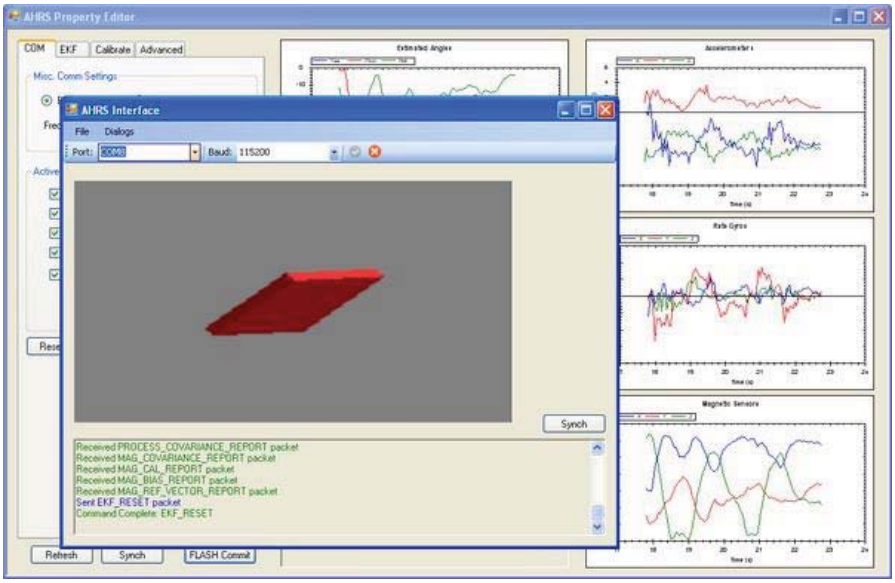


Figure 5-13: AHRS user interface with a 3D representation of the orientation (www.chrobotics.com)

5.2.1.1 Protocol

Each packet received by the AHRS must begin with the three byte sequence "snp" to signal the beginning of a new packet. The fourth byte is Packet Type indicator (PT), which identifies the packet being received.

Packet Structure

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
S	**N**	**P**	**PT**	**D1**	**Dn**	**Checksum**	

PT is the function that the user wants the AHRS to perform. If the user requires positional data, and the sensor is set to “listen mode”, the user must request the GET_DATA function.

In Listen Mode, 0x01 causes the AHRS to transmit data from all active channels in a SENSOR_DATA packet.

D1								D2								D3 D4	D5 D6
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	1st active channel	2nd active channel
yaw	pitch	roll	Yaw rate	Pitch rate	Roll rate	mx	my	mz	gx	gy	gz	az	ay	az	0		

The first two bytes following the data length byte indicate which channels are active. The actual

sensor data is contained in the remaining data bytes.

If all channels are active, then data is given in the following order:

yaw, pitch, roll,

yaw_rate, pitch_rate, roll_rate,

mag_z, mag_y, mag_x,

gyro_z, gyro_y, gyro_x,

accel_z, accel_y, accel_x

Data bytes D3 and D4 correspond to the yaw angle, D5 and D6 to the pitch angle, etc.

5.2.1.2 Calibration

The 6dm AHRS provides automated calibration routines which help simplify integration. These routines include; rate gyro bias calibration, magnetometer hard and soft, iron calibration, and accelerometer calibration. However for more accurate applications it may be necessary to individually calibrate various sensors in the AHRS. There are a variety of calibration procedures that can be performed ranging from very simple to more complex, these are listed below.

- Rate gyro bias calibration
- Rate gyro scale factor calibration
- Accelerometer bias calibration
- Magnetometer soft and hard iron calibration
- Accelerometer and magnetometer reference vectors
- Accelerometer and rate gyro cross-axis misalignment correction

5.2.1.3 Extended Kalman Filtering

The Kalman filter addresses the general issue of noise contained in measurements observed over time with the intention of producing values that are closer to the true values [34].
The Extented Kalman filter is used if the process being estimated is non-linear which is a common occurrence in an autonomous vehicle [34].

The 6dm AHRS has built in Extended Kalman filtering which makes the sensor one of the most accurate AHRS in the low budget range. The disadvantage of EKF is that in general it is not an optimal estimator, it is difficult to implement, difficult to tune, and only reliable for systems that are almost linear [34].

Invariant extended Kalman filtering (IEKF) is similar to EKF but is more suited for non-linear systems as it combines the advantages of both EKF and 'symmetry preserving filters' which is a very recent adaption to filter theory [34]. IEKF at present is only available in high grade AHRS so therefore EKF will be sufficient for the initial development of the humanoid robot.

The EKF can be tuned on the 6dm through the calibration mode but is not advised as it is tuned to a reasonable standard from the factory and there is no 'reset' for the EKF.

5.2.2 Inclinometer

An AHRS cannot be used near any magnetic interference due to its internal magnetometers. A sensor is needed on the pelvis which is surrounded by aluminium and electric motors. So to calculate pelvic angle a 2 axis inclinometer is used which works similar to the AHRS except it does not rely on magnetic sensors.

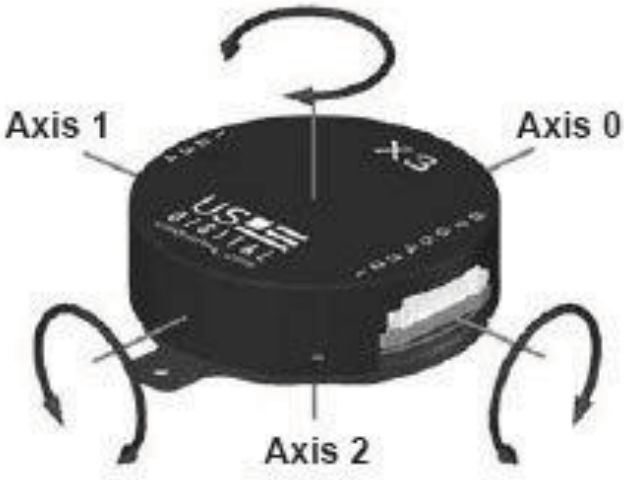


Figure 5-14: US Digital X3 MEMS Inclinometer (www.usdigital.com/products/inclinometer/absolute/x3m)

The X3M is an absolute inclinometer utilising MEMS (micro electro-mechanical systems) technology to sense tilt angles over a full 360° range in the pitch and roll axes.

This is a very industrial product. The outer casing is made from aluminium and can be mounted very securely without the chance of breakage.

5.2.2.1 Protocol

The RS-232 serial port has a configurable baud rate, 8 data bits, and no parity. The X3 does not initiate transmission so a request must be sent. For example the following hexadecimal packet is sent to request the angle of axis 1:

Sent Command: (00, E0, 01)

Notes: (address, command, axis)

The information that is returned is as follows:

Response: (00, 02, 37, 4e, 79)

Notes: (angle1--------, checksum)

00, 02, 37, 4e represents the angle divided by 1000.

In this case the angle is 145230 i.e. 145.23 degrees.

To convert these four values in vb.net an array was made to store each value the multiplied as follows:

00 + 02x65536 + 55x256 + 78x1 = 145230

Note: The hex values were converted to decimal before calculation.

In order to obtain accelerometer reading we must send the ‘Read All Data’ request.

For example:

Sent Command: (00, A0)

Notes: (address, command)

Response: (FF, FF, F9, 89,| FF, FF, F8, 01,| FF, FD, 73, 66,| 0D, C1,| 00, 00, 02, 5C,| 00, 00, 04, 28,| FF, FE, 82, 25, 00, 00,| 00, 01, B7)

Notes: (angle0, angle1, angle2, Temperature, accel0, accel1, accel2, serial number, checksum)

To acquire accel1 do the same calculation as above. i.e. accel1 value = 00000428 (hex) this equates to 0.01064g.

00 + 00*65536 + 04*256 + 40*1 = 1064/10000

There are many other requests that can be sent in order to receive information and also for calibrating and manipulation characteristics of the MEMS.

5.2.2.2 Drift Testing and Results

A static and a dynamic test was performed to measure drift. The static test consisted of the unit being mounted to a solid object that was not moving; in this case the concrete floor was used. The unit was then powered on, zeroed and left for 30 minutes; the difference in value was recorded for both axes.

The dynamic test included the use of the WCK servos which are being used in this humanoid application. The unit was zeroed on the same object as the static test, mounted to the servo and a program was set up to tilt the unit to +20 degrees then back to -20degrees. This was repeated for 30 minutes then the unit was removed, place back on the object where it was zeroed and the difference was recorded.

Figure 5-15: Dynamic drift testing of X3M MEMS inclinometer

The tests were performed 4 times and the results are displayed in (Table 5-5 and Table 5-6) below.

Static Test No.	Time tested (min)	Error (deg)
1	30	0.0112
2	30	0.0113
3	30	0.0113
4	30	0.0111

Table 5-5: Drift testing results from static test

Dynamic Test No.	Time tested (min)	Error (deg)
1	30	0.0912
2	30	0.1023
3	30	0.0901
4	30	0.0899

Table 5-6: Drift testing results from dynamic test

These results show that the X3 is very accurate and very consistent. The static test shows that there is only a 0.01 degree drift in the positive direction. The dynamic test is sufficiently larger but still only about 0.1 of a degree.

5.2.3 Foot Pressure Sensor

5.2.3.1 Interlink 402 FSR

5.2.3.1.1 *Theory of operation*

The Interlink 402 FSR is comprised of 2 polymer films: one with a conductive surface and the other with printed electrodes facing the first. Contact between the two surfaces causes the conductive layer to short circuit the printed electrodes, thereby reducing the electric resistance of the component. Typically, its resistance will drop from 1 Mohm to about 10 Kohm for an applied load of 100 g (a force of roughly 1N) to 10,000 g.

5.2.3.1.2 *Construction*

The most basic FSR consists of two membranes separated by a thin air gap. The air gap is maintained by a spacer around the edges and by the rigidity of the two membranes. (Figure 5-16) One of the membranes has two sets of interdigitated fingers that are electrically distinct, with each set connecting to one trace on a tail. The other membrane is coated with FSR ink. When pressed, the FSR ink shorts the two traces together with a resistance that depends on applied force.

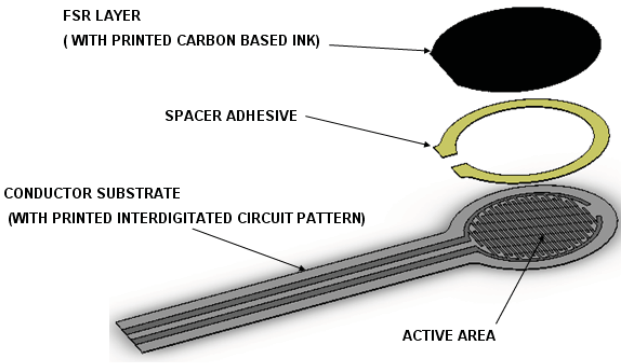


Figure 5-16: Construction of FSR pressure sensor. [30]

Around the perimeter of the sensor is a spacer adhesive that serves both to separate the two substrates and hold the sensor together. This spacer typically has a thickness between 0.03mm and 0.1mm. This spacer may be screen printed of a pressure sensitive adhesive, may be cut from a film pressure sensitive adhesive, or may be built up using any combination of materials that can both separate and adhere to the two substrates [30].

Both membranes are typically formed on flexible polymer sheets such as PET, polyimide, or any other film material. In custom force sensors, the top substrate could be made with a slightly less flexible material, such as polycarbonate, thin metal, or very thin circuit board material, as long as it is sufficiently deformable to allow a reasonable force to push the top substrate against the bottom substrate to activate the sensor [31].

The inside surface of one substrate is coated with FSR carbon-based ink (Figure 5-17) shows FSR ink under a microscope. When the two substrates are pressed together, the microscopic

protrusions on the FSR ink surface short across the interdigitated fingers of the facing surface. At low forces only the tallest protrusions make contact. At higher forces more and more points make contact.

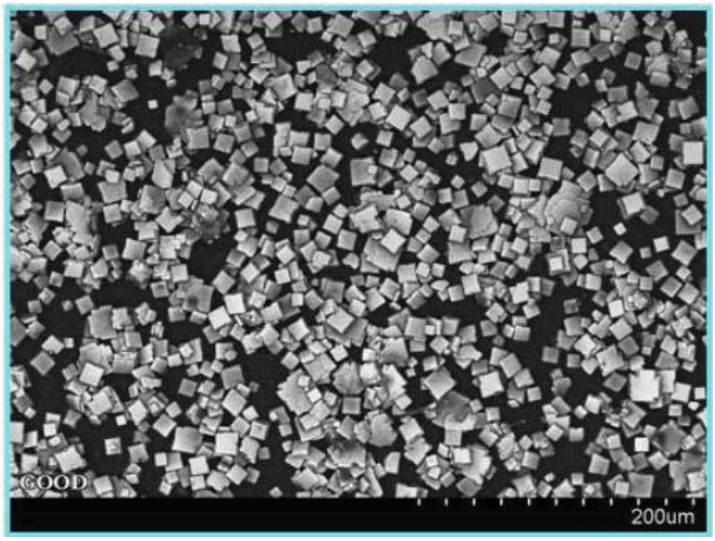


Figure 5-17: FSR carbon based ink under a microscope. [31]

The conductive traces are typically screen printed from silver polymer thick film ink. However, these traces may also be formed out of gold plated copper as on flexible or standard circuit boards (FPC or PCB) [30]. Force may be applied to either substrate. One of the exterior surfaces typically includes a mounting adhesive layer to allow mounting to a clean, smooth, rigid surface.

5.2.3.2 Implementation of FSR

5.2.3.2.1 *Communication with Pressure Sensor Board*

In order to use the FSRs a Bioloid foot pressure sensor board was implemented. This board has an onboard ATmega168, a low power CMOS 8-bit controller chip along with inputs for 4 FSR's by interlink. The board runs off 7-12V on the power line, and has an onboard regulator to bring the voltage down to 5V. The power is supplied to the Pressure sensor board through Pin 1 and Pin 2 of the connector. The board has one bus data line, half duplex Asynchronous Serial Communication, for both transmit (Tx) and receive (Rx) at TTL level.

This board is an accessory for the Bioloid which runs off the CM-5 controller, half duplex configuration. Since the communication is done on a single pin an additional board must be used. The pressure sensor board uses the same control tables as the AX12 servos, which are the servos used in the Bioloid device. So the communication protocol used for the AX-12 servos had to be implemented to communicate with the pressure sensor board. (Figure 5-18) shows the internal circuit in the CM-5 controller, which enables the controller to communicate with the pressure sensor board. Replicating this would enable us to then communicate with the FSR's.

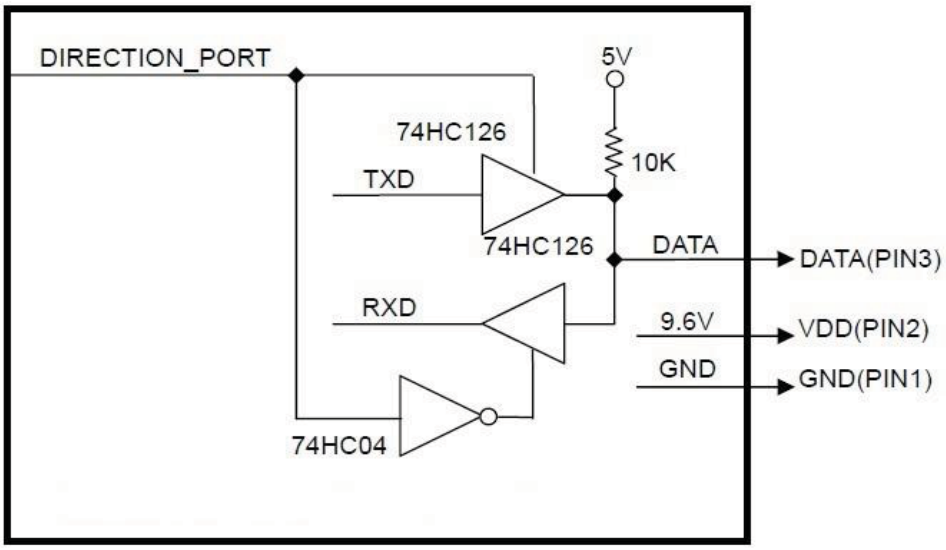


Figure 5-18: Half duplex UART circuit found within the Bioloid controller (www.robosavvy.com)

The circuit shown above is presented only to explain the use of half duplex UART. The direction of data signals on the TTL level TxD and RxD depends on the DIRECTION_PORT level as the following.

• When the DIRECTION_PORT level is High: the signal TxD is output as Data

• When the DIRECTION_PORT level is Low: the signal Data is input as RxD

To set the Direction port high or low, initially it was set up having the Tx and Rx pins on the 74HC126 (a tristate quad buffer) along with the 74HC04 (an inverter) (like in the picture above) connected to an ST232 chip which switches the signal from RS232 to TTL or vice

versa, which was then connected to a computer using a SERIAL-USB cable. This method was unsuccessful as it was not possible to achieve communication on a half duplex line using the ST232 chip. The reason for this being that the ST232 chip did not have the feature to change the DIRECTION_PORT high and low. To do this an additional chip, an FT232, was required as the FT232 has a direction pin which switches high and low automatically for direction making it possible to communicate with the pressure sensor board.

A USB-INTERFACE board was then bought which had all the necessary chips, i.e. the FT232, 74HC126 & 74HC04. The board has an FT232 which hooks up the direction port line to one of the tristates (74HC126) directly and through an inverter to the other tristate.

5.2.3.2.2 *Protocol, Testing and Results*

A typical resistance vs. force curve is shown in (Figure 5-20). This particular force-resistance curve was measured from a model 402 sensor (12.7 mm diameter). For testing purposes the FSR was placed on digital scales. The force was gradually increased and the resulting resistance value was recorded.

The “actuation force” or turn-on threshold is typically defined as the force required to bring the sensor from open circuit to below 100kΩ resistance.[3] This force is influenced by the substrate and overlay thickness and flexibility, size and shape of the actuator, and spacer-adhesive thickness. In this case the force applied was just under 100 grams.[7]

Figure 5-19: Test rig for FSR pressure sensing

(Figure 5-19) above demonstrates how the test was performed. A bolt pushed down on the FSR resulting in a weight displayed on the scales and a resulting resistance produced.

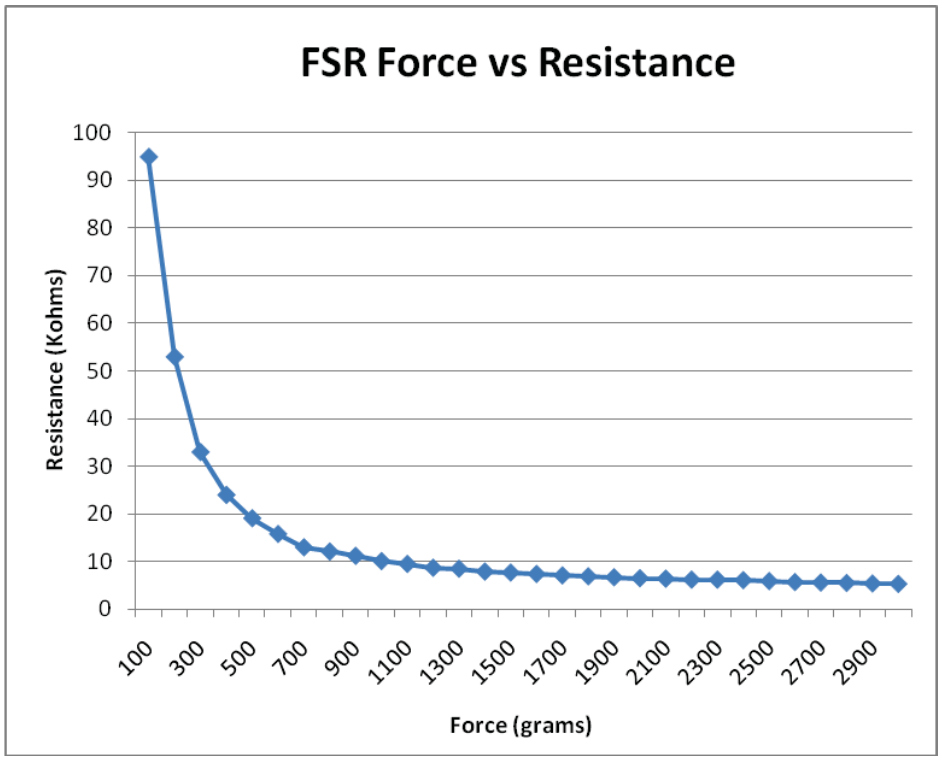


Figure 5-20: Results from FSR force testing

The computer communicates with the pressure sensor boards by sending and receiving data packets. There are two types of packets; the “Instruction Packet” (sent from the computer to the PSB) and the “Status Packet” (sent from the PSB to the computer.)

Protocol The pressure sensor boards communicate through asynchronous serial communication with 8 bit, 1 stop bit and no parity at a baudrate of 1000000bps.

As the highest baudrates on a computer is 256000bps achieving a baudrate of 1000000bps was not possible and hence made it impossible to communicate with PSB. The FTDI chip made it possible as this chip creates a virtual port which can communicate in speeds of over 1Mbps.

Unique ID – since two pressure sensor boards are being used, one for each leg of the humanoid, both PSB’s had to have their own unique ID’s. If the boards had the same ID value, multiple packets sent simultaneously collide, resulting in communication problems. Thus, it is imperative that the boards have their own unique ID. 22

Instruction Packet - The Instruction Packet is the packet sent by the main controller to the PSB units to send commands. The structure of the Instruction Packet is as the following.

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xFF	ID	LENGTH	INSTRUCTION	PARAMETER 1	PARAMETER N	CHECK SUM

Status Packet (Return Packet) - The Status Packet is the response packet from the PSB units to the Main Controller after receiving an instruction packet. The structure of the status packet is as the following.

1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	1 Byte
0xFF	0xFF	ID	ERROR	PARAMETER 1	PARAMETER 2	PARAMETER N	CHECK SUM

A terminal program was used at the initial stages to test the FSR's. This program allowed the user to send and receive packets as string, hex or ascii. Later Visual Basic.NET (VB) was used to make a program where sending and receiving was done at the click of a button.
The protocols were set as mentioned in the communication protocol and the hardware was setup as shown in (Figure 5-21).

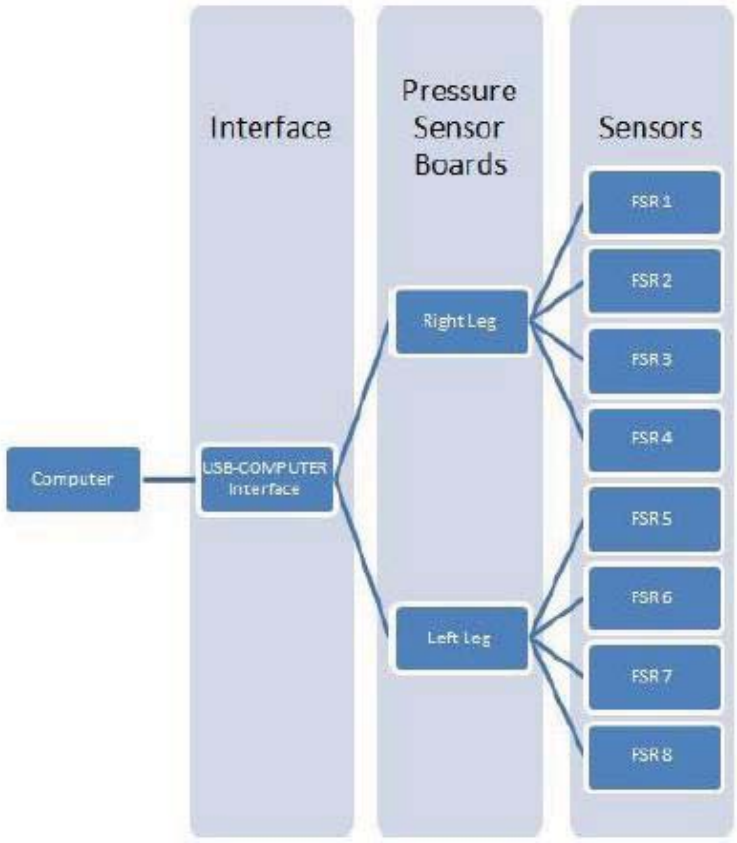


Figure 5-21: Block diagram of pressure sensor layout

An instruction packet was sent to the board via serial terminal program.

The instruction packet sent to check all four sensors connected to one of the PSB was

Data Transmitted - FF FF 01 04 02 1A 08 D6, where

0xFF – first header byte

0xFF – second header byte

0x01 – ID (set as 1)

0x04 – length of command (number of bytes past this point)

0x02 – READ_DATA command

0x1A – start address in control table

0x08 – number of bytes to read

0xD6 – checksum 24

Once this packet is received by the PSB a status packet is sent back to the computer. The status packet was

Data Received - FF FF 01 0A 00 D6 02 92 02 69 03 32 03 E7, where

0xFF – First header byte

0xFF – Second header byte

0x01 – ID (set as 1)

0x0A – Length of command (number of bytes past this point)

0x00 – Error if any (00 states that there is no error)

0xD6 – Sensor 1 low byte

0x02 – Sensor 1 high byte

0x92 – Sensor 2 low byte

0x02 – Sensor 2 high byte

0x69 – Sensor 3 low byte

0x03 – Sensor 3 high byte

0x32 – Sensor 4 low byte

0x03 – Sensor 4 high byte

0xE7 – Checksum

As the instruction in the instruction packet was to READ all four sensors, the status packet has returned with 2 bytes for each sensor, a low byte and a high byte. The values received can then be interpreted to find out how much force was applied to each sensor. This is done by using an equation to calculate sensor values.

Sensor Value = Low Byte + (High Byte * 256)

So for the bytes received in the status packet for each sensor, the sensor values are

Sensor 1 Value = D6 + (02 * 256)
= 214 + (2 *256)
= 726

Sensor 2 Value = 92 + (02 * 256)
= 146 + (2 * 256)
= 658

Sensor 3 Value = 69 + (03 * 256)
= 105 + (3 * 256)
= 873

Sensor 4 Value = 32 + (03 * 256)
= 50 + (03 * 256)
= 818

The value from each sensor is a number between 0 and 1023, proportional to the amount pressure being applied to the sensor.

After testing the sensors with the Terminal program, a program in Visual Basic.Net (VB) was written. This program, on the click of a button, sends out an instruction packet which reads all four sensors of the respective button clicked. Once sent it receives a status packet and displays the error and the sensor values of the 4 sensors in the GUI. This is shown in (Figure 5-22).

Figure 5-22: Visual basic GUI for force sensing resistors

The FSR program is then implemented into the main control program for the Humanoid robot. The sensor values from the FSR program and logic written in the main program, allow the robot to make decisions based on the values received from the sensors. So while walking, each step is completed only when all four sensors are touching the ground. Only when the ZMP is achieved will the robot carry on to its next state.

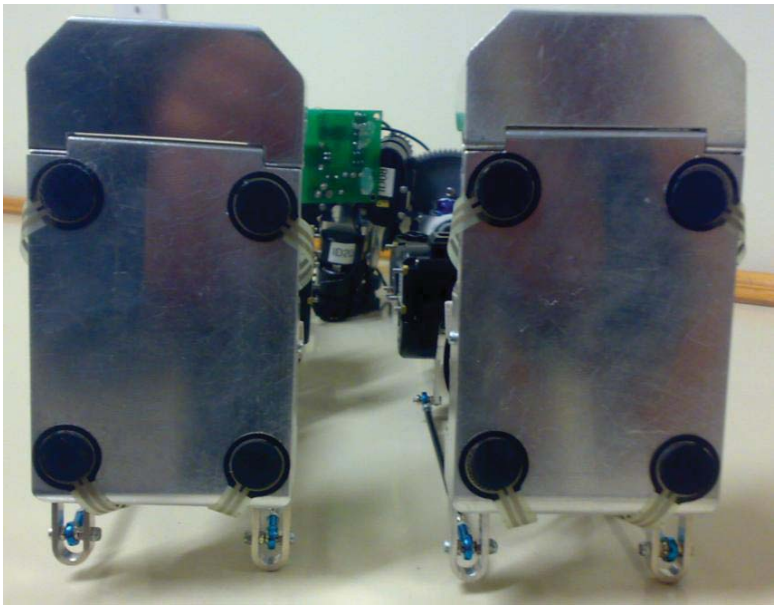


Figure 5-23: Base of humanoid feet showing pressure sensor positioning

Figure 5-23 shows the FSR sensors positioned on the corners of the feet. These sensors have additional rubber pads attached to the sensor to ensure the full load of the humanoid is placed on the sensors.

5.3 *Controller*

5.3.1 RB 110 Micro Computer

The amount of processing that will be done is so large and complex that a standard micro controller will be insufficient. The RB 110 comprises of a 1GHz processor, 256Mb DDR2 ram, 10Bit ADC, 4 regular Com ports, I2C, 2 High Speed Com ports, Windows XP operating system, and open source C++ library with unique I/O functions. This is basically a miniature computer dedicated for robotics use and has been tailored to suit humanoid robots such as Bioloid and Robobuilder which use serial controlled servos and the Kondo which uses Hitech PWM servos. (Figure 5-24) shows the OS based robot controller with all of its ports exposed.

Figure 5-24: RB110 x86 Operating system based controller board (www.roboard.com)

Specifications:

- Processor: DM&P Vortex86DX
- BIOS: AMI BIOS
- Memory: 256MB DDR 2 onboard
- Hi-Speed UART: FTDI FT2232HL Hi-Speed UART
- ADCs: Analog Devices AD-7918 10-bit
- I /O Interface:
    - Micro SD slot x1 (support class 2,4,6 SDHC with any capacity)
    - USB port x 1 (USB 2.0 version)
- Connectors:
    - 2.54 mm 3-pin box header for PWM x 16

    - 2.54 mm 10-pin box header for RS-232 x1
    - 2.54 mm 10 pin box header for High speed (COM 6) x 1
    - 2.0 mm 4 pin header for High speed (COM 5) x 1
    - 2.0 mm 4-pin header for RS-485 x 1
    - 2.0 mm 4-pin header for TTL serial x 2
    - 1.25mm 6-pin wafer for I2C x1
    - 2.54 mm 16-pin header for A/D x1
    - 2.54 mm 10-pin box header for USB x1
    - 1.25 mm 4-pin wafer for LAN x 1
    - 1.25mm 6-pin wafer for JTAG x1
    - 0.8mm 124-Pin Mini PCI Card connector
    - 3.96 mm 2 pin for Power x 2
- Power Consumption: +5V @ 400mA
- Power Input: DC-in 6V to 24V
- Dimension: 96 x 56 mm
- Resolution:
    - PWM: 20ns
    - Serial: 115200bps - 750Kbps (COM 1, 2, 3,and 4)
    - Hi-Speed Serial: up to 12Mbps (COM 5 & COM 6)
    - I2C: 1Kbps ~ 3.3Mbps
- Compatible O/S:
    - DOS, Windows 98/ME, Windows XP
    - Windows Embedded CE 6.0
    - Windows XP Embedded, Windows Embedded Standard
    - Linux distribution kernel 2.4.x, 2.6.x

5.4 *Discussion and conclusions*

The component that determines the success of a humanoid is the actuator module. Not only is the overall torque of the module important, the overall performance is equally important; how smooth the module can accelerate, its precision, its elastic feel, and the ability to tune it, and also its ease of control and programming are just a few examples. The WCK module has been adopted for this project as it has very nice functionalities which enable the module to be precisely tuned which appeals for this application.

The network protocol of the WCK servo is very simple once established, and has the ability to function high resolution of low resolution positioning. The servo requires a packet to be sent which contains a module identification, position and torque with a final byte as a checksum. A separate packet determines the modules speed and acceleration for every proceeding position request which reduces the need for continual speed and acceleration setting. The PID tuning capability is very unique and is not found in any other modules at present. This PID tuning allows the module to be tuned to act similar to a human muscle

which is more of an elastic positioning as opposed to a direct position. If over powered, these servos can nicely produce a torque of 14 kg per cm without overly straining them.

To monitor body position it is necessary for some form of gyroscopic and accelerometer sensor to be implemented. There are two commonly used forms of sensors used in low budget autonomous systems; IMU and inclinometer. IMU is an acronym for “inertia moment unit”, which essentially takes rotational and acceleration data to estimate the units position which is necessary for balancing a humanoid. An AHRS or “attitude heading reference system” is a unit that incorporates gyroscopes, accelerometers and megnetometers along with advanced estimation algorithms in order to produce a very accurate positioning unit. Basically, an AHRS is identical to regular IMUs’ and inclinometers with the ability to determine roll and pitch angles, but with the addition of a yaw axis due to its magnetometers. The addition of the yaw axis, or attitude heading, gives a complete axis representation of the position of the pelvis which is used in determining body position for the control system. The issue with an AHRS which was overlooked when investigating the system is that even though all axes can be calibrated, the yaw axis is greatly affected by metallic materials. This is due to the yaw axis relying on a magnetic sensor for its data which when surrounded by servo motors producing varying electric fields can produce undesired results. This means that the AHRS could not be mounted directly in the centre of the pelvis where first planned as this is surrounded by 8 servo motors. However since the pelvic area has been designed to be very compact it is possible to mount the AHRS just above this area in the carbon fibre backbone as it is not affected at this point.

The X3M inclinometer by USDigital is a very robust unit and is designed for low grade industrial systems. This unit only has a roll and pitch axis so, because of its lack of magnetometer, is not affected by magnetic interference which allows the unit to be placed within the pelvic area. This unit is very easy to work with as the transmit and receive protocol is similar to that of the WCK servo modules which made it very easy to interface with the software that had been produced for the servos. It is unknown how accurate these sensors must be as a walking algorithm has not been developed, even though, several static and dynamic tests were performed on the unit to determine its consistency and accuracy.

Bioloid pressure sensors have been implemented on the feet of the humanoid. The reason for this is that a commonly used control theory is to determine a point of zero torque, or equal pressure, known as the ZMP. This theory measures the pressure at each corner of the foot producing a combined pressure which is then monitored and manipulated appropriately by shifting body weight to result in a balanced robot. Basically if the pressure on the front of the foot is higher than the back, for example, then the robot is tending to topple forward.

These pressure sensors are called FSR sensors. These are a thin surface with an interdigitated surface which is then contacted with a conductive film varying the resistance depending on applied pressure. These sensors are connected to a small processing board mounted in the foot which converts the analog data to a digital serial packet. The on-going theme for sensors on this robot was to keep all processing onboard or as close to the sensor as possible. This greatly reduces transmission error and the amount of processing that is required by the controller. Once the sensor value has been converted at the foot it is then sent up to an interface board, then into the controller.

The processing required to solve the very complex walking algorithms is demanding. A regular micro controller is capable of driving all servo modules and reading all sensor values but would not be capable of the processing. The RB110 is a 32bit operating system based controller board. This board measures only 96mm by 50mm, consumes less than 1W, is 1GHz, has 256mb ram, and a full Windows XP operating system. The RB110 is unique in the mini pc area as it is dedicated to robotics control with high speed serial port, 16PWM ports, GPIOs and several other commonly used ports.

The RB110 was installed with a lightweight version of Windows XP with only the necessary applications and drivers installed. Since a controller like this is based on COM port control, a regular desktop computer can be used to develop and test code before it's transferred to the controller which greatly increases debugging and compiling time.

At present the RB110 controller has only been used for basic control of the system such as servo control and monitoring of sensors. It has yet to be used to its full potential as walking, and full inverse, kinematics algorithms have not been produced.

To conclude, WCK servos have been used for joint modulation due to their advanced PID motion control and simple to program features. Two position sensors have been implemented and tested to determine their accuracy and consistency which are used to monitor body and pelvis position. Bioloid pressure sensors are on each corner of both feet to monitor overall foot pressures so that the ZMP balance theory can be implemented. A Windows XP based controller dedicated to robotics is the brain and heart of the humanoid robot. This controller drives all actuator servos, monitors all sensors, and determines the position each joint should be at.

6 Software

Microsoft Robotics Developer Studio is a commonly used programming language for simple robotics control. Serial communication in this language is very simple, however more complex scripts are very time consuming and difficult to develop. Therefore for this project it has been decided that Visual Basic.NET will be used. The reason for this it that even though it is a reasonably old language it is still fully supported by Microsoft and therefore has a lot of sample code and support on the internet.

6.1 *Graphical User Interface (GUI)*

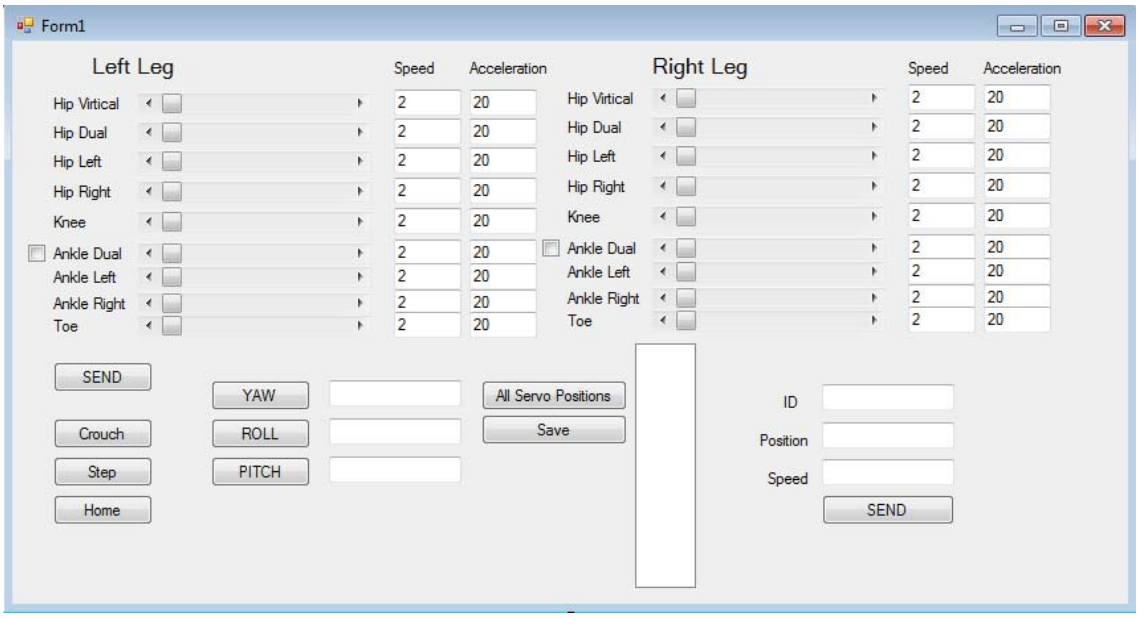


Figure 6-1: Visual basic GUI for humanoid control and monitoring

A GUI was developed in VB.NET (Figure 6-1) for monitor and control of sensors and modules. This interface is not intended to give the user the ability to modify code or apply walking algorithms for example. However what is was produced for is to test samples of code and establish a structured code setup to enable future developers to be able to simply use the established code for gait and balancing algorithms. Basically what was developed was the ability to; move all joints to any angle within a limited safety range, servo speed torque and acceleration can be set, yaw roll and pitch angles can be read, all servo positions can be read and saved into a text file, and the robot can home crouch and take a simple pre

program step forward. To read foot pressure sensors a separate GUI was developed which is described under the foot sensor section.

6.2 *Serial Servo Module*

6.2.1 Serial Port

For serial communication the software must have access to a communication port. In this case COM 4 on the controller has built in FTDI which is similar to a MAX232 chip allowing multiple servo driving. So for communication COM 4 is allocated with a baud rate of 115200bps, DataBit=8, ReadDataSize=4096.

There are some house-keeping issues that must be done when working with serial ports in Visual Basic. Below are some key issues and points that must be considered when programming serial communication in VB.

It is bad practice to leave the serial port open when not in use.

```
If Me.SerialPort1.IsOpen = True Then
            Me.SerialPort1.Close()
        End If
```

If the serial port is closed, open it as data cannot be sent otherwise.

```
If Me.SerialPort1.IsOpen = False Then
              Me.SerialPort1.Open()
        End If
```

Make sure the In and Out buffer of the serial port is clear before reading or writing.

```
      Me.SerialPort1.DiscardInBuffer()
      Me.SerialPort1.DiscardOutBuffer()
```

6.2.2 High Resolution Positioning

Low resolution positioning is still very accurate would possibly be acceptable for the development of a walking gate. However the high resolution capability is available and not much harder to implement and therefore is used in this application. (Figure 6-2) Section 1 splits the 16bit HighLowByte into a high and low byte of 8bits each performing the necessary bit shifting described in the protocol section. (Figure 6-2) Section 2 assigns all

associated data into an 8 bit array called temp which is then stored into the associated joint array which will then be send out once function SendOUTSerial is called.

```
Dim xHArray(7), xLArray(7) As Byte
        For n = 0 To 7
            xHArray(n) = HighLowByte(n + 1)
        Next
        For n = 8 To 15
            xLArray(n - 8) = HighLowByte(n)
        Next
        xLArray(0) = 0

        For n = 0 To 7
            H = H & xHArray(n).ToString
            L = L & xLArray(n).ToString
        Next
        xH = Convert.ToInt32(H, 2)
        xL = Convert.ToInt32(L, 2)


        temp(0) = 255
        temp(1) = 224
        temp(2) = 200
        temp(3) = ID
        temp(4) = 0
        temp(5) = xH
        temp(6) = xL
        Checksum = (ID Xor 0 Xor xH Xor xL Xor 224 Xor 200) And 127
        temp(7) = Checksum
```

1

2

Figure 6-2: High resolution positioning software layout

6.2.3 Speed and Acceleration

The speed and acceleration of a module once set will remain the same until altered. The way this code is structured requires the user to first assign Speed to a value within 0-15 and assign Acceleration to a value to within 20-100. Once the SpeedANDAcceleration command is called an array called SpeedAccArray is loaded with the associated values and the check sum is calculated. (Figure 6-3) Section 1 in red below demonstrates this. It is then checked to see whether the serial port is open where the SpeedAccArray is then sent out.

```
Public Sub SpeedANDAcceleration()
        Dim SpeedAccChecksum As Byte
        Dim ModeID As UInt16

        ModeID = ID + 224

        Dim SpeedAccArray(5) As Byte
        SpeedAccArray(0) = 255
        SpeedAccArray(1) = ModeID
        SpeedAccArray(2) = 13
        SpeedAccArray(3) = Speed
        SpeedAccArray(4) = Acceleration
        SpeedAccChecksum = (ModeID Xor 13 Xor Speed Xor Acceleration) And 127
        SpeedAccArray(5) = SpeedAccChecksum

        If Me.SerialPort1.IsOpen = False Then
            Me.SerialPort1.Open()
        End If

        SerialPort1.Write(SpeedAccArray, 0, 6)
    End Sub
```

1

Figure 6-3: Speed and acceleration software layout

6.2.4 Joint Positioning

There is a delay between sending serial data for one joint, calculating the next joint then sending that data out for the next joint. It became apparent that when a lot of servos were being moved there was a sufficient time delay which would become an issue when developing a walking algorithm.

To solve this issue the software performs as follows;

- The 'Initiation' state assigns every array to 0
- The user will define the required angle (eventually an algorithm with calculate this)
- The angle will then be set in the joint to be move. RightToe – PelvisVertical. These sub programs have all assigned ID information and then call another sub program called AngleSend
- AngleSend takes the ID and Angle of the servo and then calculates all byte information for the serial packet. This packet is then stored into an array with the same name as the desired joint to be moved.
- This can be repeated for all joints before the function SendOUTSerial is called. Once this is called all serial packet data stored in the arrays will be sent out the specified Com port.

Below in (Figure 6-4) shows the functional flow diagram of the software.

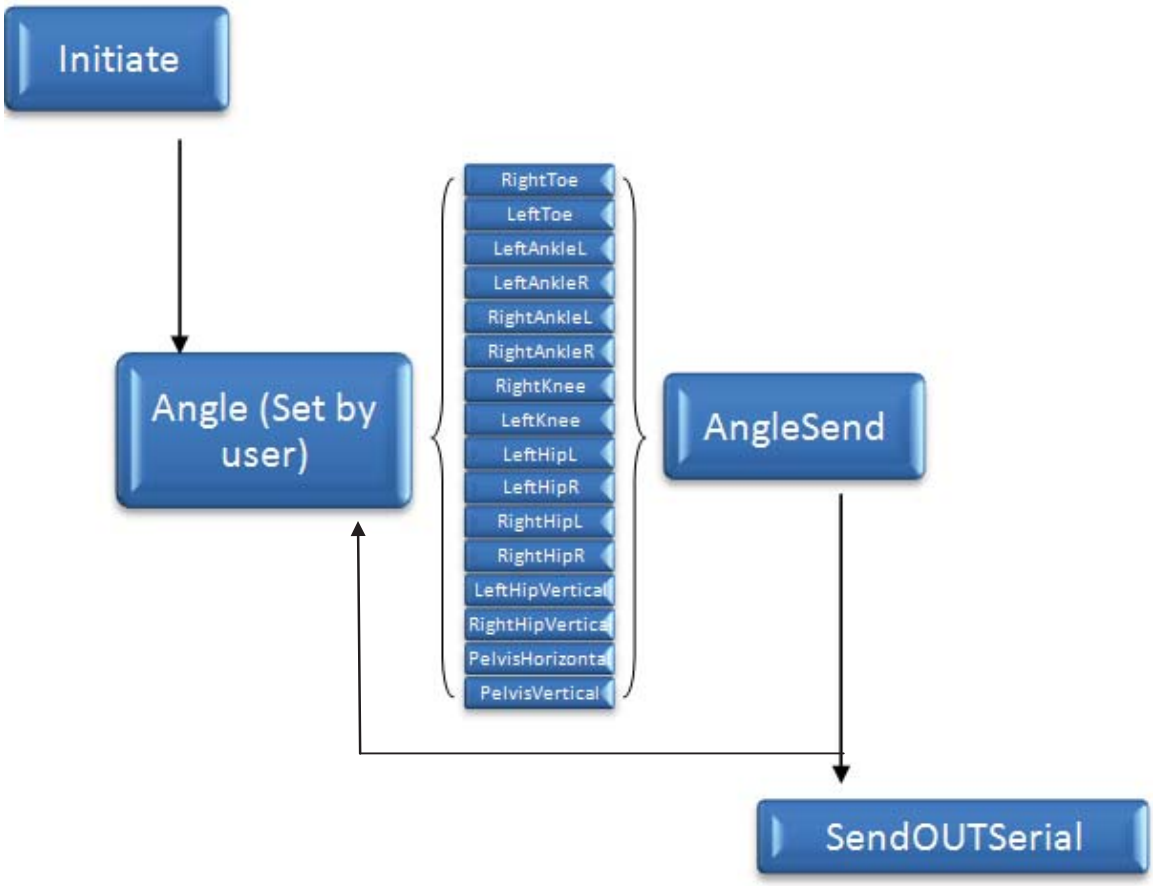


Figure 6-4: Block diagram of software layout for module actuation

6.3 *X3M Inclinometer*

The X3M, even though specified as 3 axis, is a little deceiving. The 3rd axis is not a yaw or horizontal axis as implied in the data sheet, instead it is another pitch but mounted 90 degrees around from the original pitch axis. Essential this means you have 2 options for pitch and roll, not an additional 3rd axis.

This is a very easy sensor to implement. Firstly the user must send a 3 byte request for an angle of a specific axis. Secondly the sensor will send back a 5 byte packet which is stored and can be converted into an angle.

6.3.1 Angle value acquisition

This portion of code is purely for acquiring the angle of a particular axis. The X3M has many other functions that can be acquired such as acceleration and temperature sensors but these were not used. (Figure 6-5) Section 1 shows the data packet that must be sent to retrieve the angle of axis 0. (Figure 6-5) Section 2 shows how the serial port is read and all data stored into and array called GyroAngle which is then converted into a usable angle in degrees.

```
Private Sub Gyro()
        Dim GyroAngle1, GyroAngle2, GyroAngle3 As Int64

        Dim GyroSend(2) As Byte
        GyroSend(0) = 0
        GyroSend(1) = 224
        GyroSend(2) = 0

        If Me.SerialPort2.IsOpen = False Then
            Me.SerialPort2.Open()
        End If

        Me.SerialPort2.Write(GyroSend, 0, 3)

        Dim GyroAngle(4) As Byte

        Me.SerialPort2.Read(GyroAngle, 0, 5)

        GyroAngle1 = GyroAngle(1)
        GyroAngle1 = GyroAngle1 * 65536
        GyroAngle2 = GyroAngle(2)
        GyroAngle2 = GyroAngle2 * 256
        GyroAngle3 = GyroAngle(3)

        GyroServoAng = GyroAngle1 + GyroAngle2 + GyroAngle3

        Me.TextBox1.Text = GyroServoAng
    End Sub
```

1

2

Figure 6-5: X3M Inclinometer software layout

6.4 *Pressure Sensor*

The pressure sensor software is separate from the main program at present. The reason for this is for debugging purposes as individual programs are easier to work with.

6.4.1 Pressure value acquisition

(Figure 6-6) Section 1 shows the 8bit packet that must be sent to the pressure sensor board to request all pressures of the sensors on the right foot. The left foot has the same coding except the ID byte in the packet is changed to the associated ID.

```
    Public Sub RInstructionPacket()
        Dim RightLegArray(7) As Byte
        RightLegArray(0) = 255
        RightLegArray(1) = 255
        RightLegArray(2) = 1
        RightLegArray(3) = 4
        RightLegArray(4) = 2
        RightLegArray(5) = 26
        RightLegArray(6) = 8
        RightLegArray(7) = 214
        If Me.SerialPort1.IsOpen = False Then
            Me.SerialPort1.Open()
        End If
        Me.SerialPort1.Write(RightLegArray, 0, 7)
    End Sub
    Public Sub RStatusPacket()

        'Public Sub RequestServo()
        Dim RxIncoming(14) As Byte
        Dim RxByte As Integer
        Dim Receive(4) As Byte
        If Me.SerialPort1.IsOpen = True Then
            Me.SerialPort1.Close()
        End If

        Receive(0) = 255
        Receive(1) = 255
        Receive(2) = 1
        Receive(3) = 10


    End Sub
End Class
```

1

Figure 6-6: Pressure value software layout

Please refer to appendix A for entire visual basic code.

6.5 *Discussion and conclusions*

VisualBasic.net was the programming language adopted for this project. The reasoning behind this is not because it is 'better' than other languages such as C# or C++ but because of previous experience with similar projects at Massey. VB.net has a very simple user interface and serial communication is produced by simply dragging and dropping the appropriate function into the GUI. Also VB.net is widely support on programming forums so simple codes are easy to come by which can greatly reduce development time.

A GUI was developed which enable the used to manipulate individual joints, request current positions of joints, pressure sensor values, and inclinometer values. To enable this GUI to work, code was developed which sent out appropriate packet information for requests such as position, acceleration and torque, current position and torque requests, and overload settings.

The overall system for positioning joints was set up as follows; firstly the ID of the servo is specified, next the acceleration and speed is set and sent out to the servo through the serial port, the angle of the joint is specified and sent to a function which calculated the appropriate packet information which is then stored in its own function. Finally, once the user or program has set all angles, a final function sends out all packets in one continuous stream. Doing this eliminated the small delay between servo calculations which resulted in undesirable effects.

To conclude, software has been developed so that there is a base for the next step which is to develop a walking algorithm. All sensor values can be read and stored in a function within the program ready to be used. The layout of the program allows a used to request a specific joint to move to any angle within its limits without the need to inquire what the servo ID numbers are or what packets need to be sent.

7 Inverse Kinematics

Kinematics is a process whereby, when given all joint angles of a manipulator, you can determine the position and orientation of the end effecter. Inverse kinematics is simply the opposite where a desired position and orientation is provided and from this all joint angles can be deduced. It is simple in theory but is far from simple in practice as there can be a resultant solution between zero and an infinite number of results which requires extremely complex algorithms to determine which result is appropriate.

Human beings have the ability to naturally determine which path and orientation of joint path is most efficient for a given situation. For example when reaching to turn on a tap, humans do this without considering the relative configuration of their shoulder and elbow required to reach the tap.

For the control of a humanoid it is necessary for the control algorithm to have some form of inverse kinematics [49] [37]. The kinematics control algorithm will determine all necessary servo angles using the end effecter position determined by the walking algorithm, which in this case is the humanoid foot.

There are several methods for solving IK problems, coming originally from robotics applications. These include cyclic coordinate descent methods [50], pseudoinverse methods [51], Jacobian transpose methods [53], the Levenberg-Marquardt damped least squares methods [52], quasi-Newton and conjugate gradient methods [50], and neural net and artificial intelligence methods [54] [55].

7.1 *6 Degrees of Freedom (IK)*

Once the number of degrees of freedom are more than 3 the computation becomes very complex [46] [49]. (Figure 7-2) shows a 3DOF manipulator, this shows how there are two solutions to give the same end effector position and orientation.

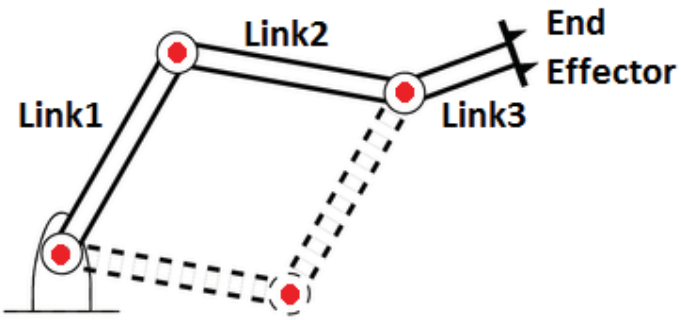


Figure 7-1: 3DOF manipulator

The humanoid that was constructed has 7 DOF per leg. 1 DOF however was the toe and for simplicity could be excluded when developing the inverse kinematics.

(Figure 7-2) shows the functional system layout of one leg showing dimensions and DOF. This information is used to deduce the Denavit Hartenberg parameters which are used for calculating the inverse kinematics.

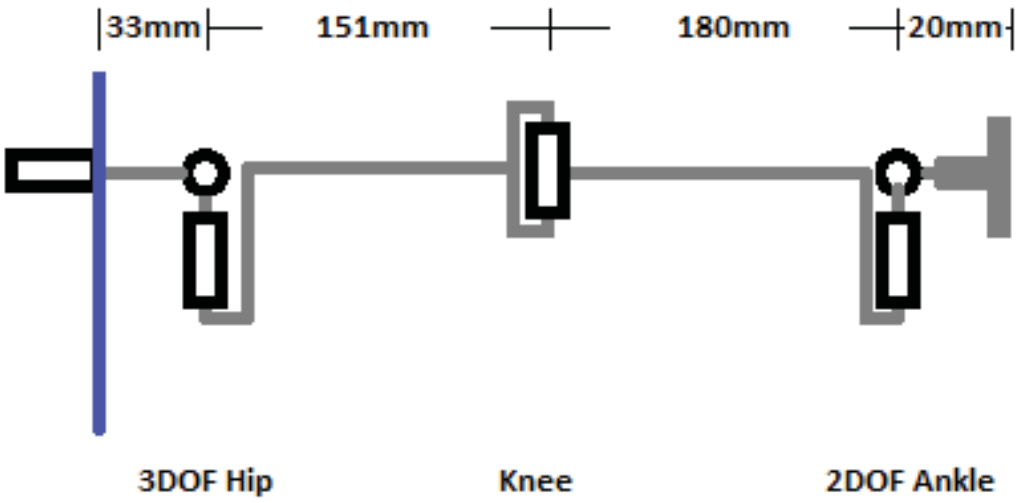


Figure 7-2: Schematic representation of the 6DOF humanoid leg

7.1.1 Denavit Hartenberg Convention

A commonly used convention for selecting frames of reference in robotics applications is the DH convention which was introduced by Jaques Denavit and Richard S. Hartenberg [55]. The transformation is represented as a product of four basic transformations.

1. the *z*-axis is in the direction of the joint axis
2. the *x*-axis is parallel to the common normal:

$$X_n = Z_n \times Z_{n-1} \quad \textbf{(7-1)}$$

3. the *y*-axis follows from the *x*- and *z*-axis by choosing it to be a right-handed coordinate system.

Each homogeneous transformation Ai is represented as a product of four basic transformations. The parameters of the transformation are as follows and are known as D-H Parameters:

- r_i: link length
- α_i: link twist
- d_i: link offset
- θ_i: link angle

Each transformation, A_i, is represented as a product of these previously stated parameters. This is shown in (7-2) which is the product of the θ and α rotational matrix and the transpose matrix of both r and d. The matrices of (7-2) are shown in (7-3).

$$A_i = Rot_{z,\theta i} \,.\, Trans_{z,di} . Trans_{x,ri} . Rot_{x,\alpha i} \quad \textbf{(7-2)}$$

$$\left\{\begin{array}{ccc|c} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{ccc|c} 1 & 0 & 0 & \alpha_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}$$

(7-3)

Following is the DH convention process with corresponding matrices.

Hip:

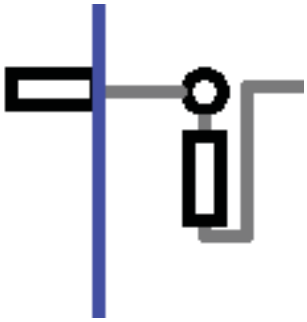


Figure 7-3: Schematic representation of hip joint

The DH convention process is as follows.

- Read from left to right.
- Moving from the blue line in (7-3) to the next joint axis the distance ‘d’ is 33mm which is shown in Figure 7-2.
- This same axis is also rotated 90 degrees about the vertical plane α

These two values are then filled into the top row in Table 7-1 below.

- The second row of the table is the final axis of the hip. This is the pitch axis which is orientated 90 degrees from the initial position in both θ and α plane.

Matrix Number	d	r	θ	α
1 Hip	33	0	0	90
2 Hip	0	0	90	90

Table 7-1: DH convention of the humanoid hip.

$$A_1=\left\{\begin{array}{cccc} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 33 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}4)$$

A_1 is the first DH convention of row '1 Hip' in Table 7-1. It can be seen that the corresponding rotational or transpose matrix is only implemented in the formula if there is a value in the table. 'd' equals 33mm so the appropriate transpose matrix is used.

$$A_1=\left\{\begin{array}{cccc} C\theta_1 & -C\alpha_1 S\theta_1 & S\alpha_1 S\theta_1 & 33 S\alpha_1 S\theta_1 \\ S\theta_1 & C\alpha_1 C\theta_1 & -C\theta_1 C\alpha_1 & -33 S\theta_1 S\alpha_1 \\ 0 & S\alpha_1 & C\alpha_1 & 33 C\alpha_1 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}5)$$

After these matrices are expanded using the dot product method the resultant matrix is (7-5).

$$A_2=\left\{\begin{array}{cccc} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}6)$$

A_2 is completed in the same manner as A_1 using the values and appropriate matrices for '2 Hip'. This is shown in (7-6) and is reduced in (7-7).

$$A_2 = \left\{ \begin{array}{cccc|c} C\theta_2 & -C\alpha_2 S\theta_2 & S\alpha_2 S\theta_2 & 0 \\ S\theta_2 & C\alpha_2 C\theta_2 & -C\theta_2 C\alpha_2 & 0 \\ 0 & S\alpha_2 & C\alpha_2 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array} \right\} \quad \textbf{(7-7)}$$

Hip to Knee:

Figure 7-4: Schematic representation of hip to knee area

Figure 7-4 is a schematic representation of the section running from the hip to the knee. Following, is the method used to determine the DH parameters of this section of the leg. Table 7-2, matrix (7-8) and matrix (7-9) are derived using the same methods described previously.

Matrix Number	d	r	θ	α
3 Hip to Knee	0	150	0	0

Table 7-2: DH convention of the humanoid knee

$$A_3 = \left\{ \begin{array}{ccc|c} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array} \right\} \left\{ \begin{array}{ccc|c} 1 & 0 & 0 & 150 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array} \right\} \quad \textbf{(7-8)}$$

$$A_3=\left\{\begin{array}{ccc|c} C\theta_3 & -S\theta_3 & 0 & 151\,S\theta_3 \\ S\theta_3 & C\theta_3 & 0 & 151\,S\theta_3 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}9)$$

Knee to ankle:

Figure 7-5: Schematic representation of knee to ankle area

Figure 7-4 is a schematic representation of the section running from the knee to the ankle. Following, is the method used to determine the DH parameters of this section of the leg. Table 7-3, matrix (7-8) and matrix (7-9) are derived using the same methods described previously.

Matrix Number	d	r	θ	α
4 Knee to Ankle	0	180	0	0

Table 7-3: DH convention of the humanoid knee to ankle

$$A_4=\left\{\begin{array}{ccc|c} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{ccc|c} 1 & 0 & 0 & 180 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}10)$$

$$A_4=\left\{\begin{array}{cccc} C\theta_4 & -S\theta_4 & 0 & 180\ S\theta_4 \\ S\theta_4 & C\theta_4 & 0 & 180\ S\theta_4 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}11)$$

Ankle and foot:

Figure 7-6: Schematic representation of foot area

The final part of the DH parameters are in the ankle. This is shown in Figure 7-6 and Table 7-4 which is a 2 DOF joint. The foot area however is actually 3 DOF as it has the additional toe but for simplification of the initial system the toe can be ignored.

Matrix Number	d	r	θ	α
5 Ankle	0	0	0	90
6 Foot	0	20	0	0

Table 7-4: DH convention of the humanoid ankle

$$A_5=\left\{\begin{array}{ccc|c} C\theta_i & -S\theta_i & 0 & 0 \\ S\theta_i & C\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\}\left\{\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & C\alpha_i & -S\alpha_i & 0 \\ 0 & S\alpha_i & C\alpha_i & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad (7\text{-}12)$$

$$A_5 = \left\{\begin{array}{cccc|c} C\theta_5 & -C\alpha_5\, S\theta_5 & S\alpha_5\, S\theta_5 & & 0 \\ S\theta_5 & C\alpha_2\, C\theta_5 & -C\theta_5\, C\alpha_5 & & 0 \\ 0 & S\alpha_5 & C\alpha_5 & & 0 \\ \hline 0 & 0 & 0 & & 1 \end{array}\right\} \quad \textbf{(7-13)}$$

$$A_6 = \left\{\begin{array}{cccc|c} C\theta_i & -S\theta_i & 0 & & 0 \\ S\theta_i & C\theta_i & 0 & & 0 \\ 0 & 0 & 1 & & 0 \\ \hline 0 & 0 & 0 & & 1 \end{array}\right\} \left\{\begin{array}{cccc|c} 1 & 0 & 0 & & 20 \\ 0 & 1 & 0 & & 0 \\ 0 & 0 & 1 & & 0 \\ \hline 0 & 0 & 0 & & 1 \end{array}\right\} \quad \textbf{(7-14)}$$

$$A_6 = \left\{\begin{array}{ccc|c} C\theta_6 & -S\theta_6 & 0 & 20\, S\theta_6 \\ S\theta_6 & C\theta_6 & 0 & 20\, S\theta_6 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right\} \quad \textbf{(7-15)}$$

Matrix (7-12) through to (7-15) are derived in the same manner as the hip was.

The transform matrix (7-16) of all of these results determines the set of equations that need to be solved.

$$T_6 = A_1\ A_2\ A_3\ A_4\ A_5\ A_6 \tag{7-16}$$

$$T_6 = \left\{\begin{array}{cccc|c} R_{11} & R_{12} & R_{13} & & d_x \\ R_{21} & R_{22} & R_{23} & & d_y \\ R_{31} & R_{32} & R_{33} & & d_z \\ \hline 0 & 0 & 0 & & 1 \end{array}\right\} \tag{7-17}$$

$$R_{11} = (C_1 \times C_2 \times C_{345} + S_1 \times S_{345}) \times C_6 + C_1 \times S_2 \times S_6$$

$$R_{21} = (S_1 \times C_2 \times C_{345} - C_1 \times S_{345}) \times C_6 + S_1 \times S_6$$

$$R_{31} = S_2 \times C_{345} \times C_6 - C_2 \times S_6$$

$$R_{12} = (C_1 \times C_2 \times C_{345} + S_1 \times S_{345}) \times -S_6 + C_1 \times S_2 \times C_6$$

$$R_{22} = (S_1 \times C_2 \times C_{345} + C_1 \times S_{345}) \times -S_6 + S_1 \times S_2 \times C_6$$

$$R_{32} = S_2 \times C_{345} \times -S_6 - C_2 \times C_6$$

$$R_{13} = (C_1 \times C_2 \times S_{345} + S_1 - C_{345})$$

$$R_{23} = (S_1 \times C_2 \times S_{345} - C_1 \times -C_{345})$$

$$R_{33} = S_s \times S_{345}) \tag{7-18}$$

(7-18) are the set of equations that must be solved to determine the rotation angle of each joint. There can be many or even infinite solutions to these equations.

7.1.2 Inverse Jacobian Method

A Jacobian is a vector derivative with respect to another vector. Basically the Jacobian method takes the error in desired position and estimates a small change in joint angles in order to move the end effecter closer to the desired position. The Jacobian is one of the most important quantities in the analysis and control of robot motion [56] [36].

Forward kinematics is a non linear function as it involves sin's and cos's of the input variables [56] Therefore we can only use the Jacobian as an approximation that is valid near the desired end effector position

The Jacobian matrix J(**e**,**Φ**) shows how each component of **e** (end effector position) varies with respect to each joint angle.

$$J(\mathbf{e},\mathbf{\Phi}) = \begin{Bmatrix} \frac{\partial e_x}{\partial \phi_1} & \frac{\partial e_x}{\partial \phi_2} \\ \frac{\partial e_y}{\partial \phi_1} & \frac{\partial e_y}{\partial \phi_2} \end{Bmatrix}$$

(7-19)

θ represents the current joint angles and e represents the current end effector position, g is used to represent the desired end effector position.

To choose a value for Δe that will move e closer to g (7-20) is used. β is a value between 0 and 1 in order to reduce the size of the step.

$$\mathbf{\Delta} e = \beta(g - e)$$

(7-20)

A simple Jacobian algorithm for solving inverse kinematics is as follows

Compute J(e,θ) for the current pose θ

1. Invert the Jacobian matrix

$$J^{-1}$$

(7-21)

2. Approximate step

$$\Delta e = \beta(g - e)$$

(7-22)

3. Compute change in joint degrees of freedom

$$\Delta\theta = J^{-1}.\Delta e$$

(7-23)

4. Apply changes to angles

$$\theta = \theta + \Delta\theta$$

(7-24)

5. Compute new e vector and apply forward kinematics to see where the end effector is positioned

7.2 *A 3-dimensional trigonometric approach*

The inverse kinematics problem is extremely demanding once the number of DOF in a manipulator is greater than 3. The DH parameters and the jacobian matrix become so complex that if there is a solution to the problem in most cases there will be so many solutions to the same problem that an appropriate one cannot be chosen.

Therefore a new approach to solving the inverse kinematics is proposed. The proposed approach is the 3D trigonometry approach using basic trigonometric rules. The manipulator will be viewed both from the front plane and the sagittal plane where appropriate angles are calculated depending on the desired Z and X foot position. Figure 7-7 below shows the sagittal plane of the proposed trigonometry approach, this takes the centre of the hip horizontally and vertically as a zero point. This means that the end effector, the centre of

the foot, has a desired position a Z distance from the horizontal hip and an X distance from the vertical hip.

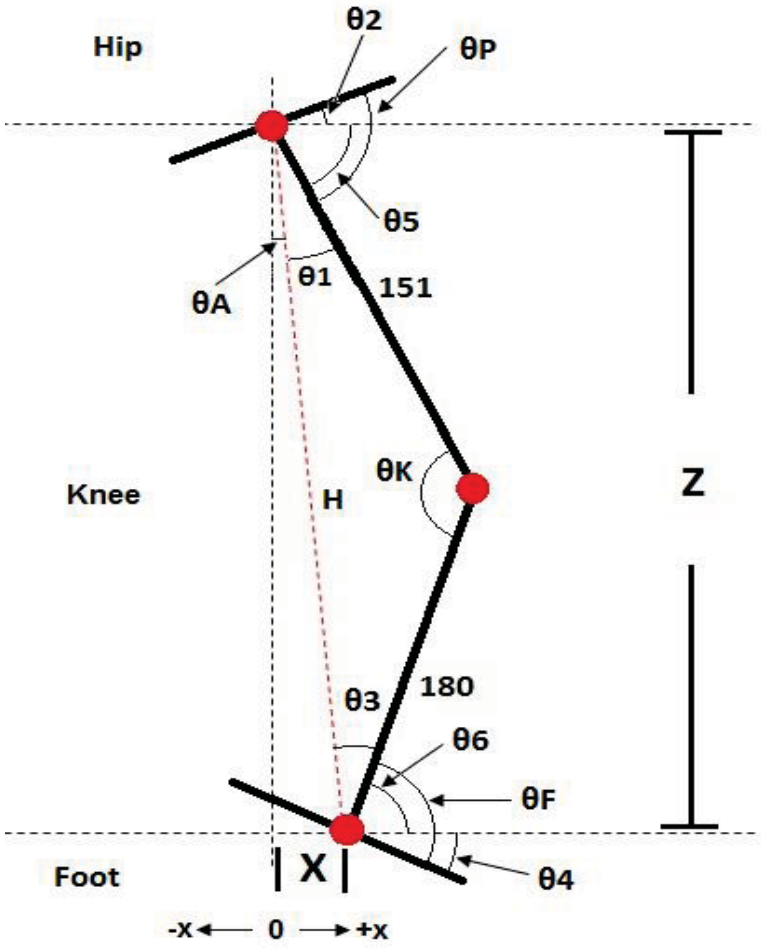


Figure 7-7: Trigonometric representation of humanoid leg from sagittal plane

Calculate length of H

$$H = \sqrt{Z^2 + X^2} \tag{7-25}$$

Calculate θK

$$A = \frac{b^2 + c^2 - a^2}{2ac} \tag{7-26}$$

$$\frac{180}{\sin A} = \frac{H}{\sin \theta K}$$

(7-27)

Calculate θA

$$\theta A = \tan^{-1}\frac{X}{Z}$$

(7-28)

$$\theta 5 = 90 - \theta A - \theta 1$$

(7-29)

Calculate θP

If x is positive

$$\theta P = \theta 2 + \theta 5$$

(7-30)

If x is negative

$$\theta P = \theta 5 - \theta 2$$

(7-31)

Calculate θF

If x is positive

$$\theta F = \theta 6 + \theta 4$$

(7-32)

If x is negative

$$\theta F = \theta 6 - \theta 4$$

(7-33)

Basically using these sets of equations θP, θK, θF which represent pelvis, knee and foot angles respectively can be calculated.

The front plane of this manipulator is the same minus the knee joint. Basically it turns into a 1 link manipulator which is far simpler than the sagittal plane.

7.3 *Discussion and conclusions*

It is essential in manipulator control to produce a means to calculate the inverse kinematics (IK). Inverse kinematics allows the user to determine an end point and orientation and the IK will calculate the appropriate joint angles. For a humanoid that would mean the control system would only have to specify where it would like to place the feet, it would not have to worry about every single joint and servo angle.

The concept seems simple enough but in fact once a manipulator exceeds 3 degrees of freedom the IK algorithm complexity grows exponentially. For a 3D, 7degree of freedom, double manipulator system the IK can potentially have an infinite number of solutions which means determining the best suited solution is basically not realistically possible in the short time spaces available.

A lot of different inverse kinematic techniques were investigated and several of the more common were investigated in depth. The DH parameters were determined and the resulting matrices were produced which inevitably led to a set of complex formulae that would need to be solved. The inverse Jacobian method is slightly more simple but requires a large number of iterations as it works more like a guess and check method.

It was decided that a simpler method could be developed purely for manipulators such as humanoid legs which are high in DOF but low in their number of links. A basic trigonometric approach is proposed which views both sagittal plan and frontal plan then combines the two to produce a 3D model. At present only the sagittal and frontal plane have been represented. A technique like this seems more suited to a humanoid as leg movement is mechanically limited and will move in a repetitive trajectory.

8 Conclusions and Future work

Humans have an obsession with producing mechanical counterparts of themselves. Humanoid robots could be used to perform tasks in un-safe environments that would put a human life at risk; they could also be used as a companion or helper for disabled or old people.

The work developed in this thesis outlines common flaws and restraints that plague the successful design of humanoid robots. The number one restraint that determines how agile and human-like the robot will be is the power to weight ratio of the servo actuator modules used. High power actuators are too expensive for a low budget research project, and in order to be able to use low power modules it was necessary to implement a unique form of joint actuation. With the joint actuation proposed and implemented, instead of using a single servo to actuate each axis, two servos are used at the same time to actuate both axes. This system results in a maximum joint torque twice as large as that of a classical layout. Also the characteristics of this system allow the servo actuator modules to be placed further up in the body which reduces leg inertia further increasing the overall torque the modules can supply.

Various aspects of biomechanics were investigated and their effect on the operation of a humanoid. Aspects such as the implementation of a knee cap which increases ease of height control and the resulting effect a toe joint has on the stride characteristics of a humanoid.

Several different designs and iterations were produced before a final full 3D model was produced. This model was used to test the flexibility, strength of components, and inertia characteristics of the proposed humanoid before a mechanical structure was manufactured.

Using the 3D model, computer aided machining software was used to produce machine code for the CNC machine in order to manufacture appropriate components. Most components on this humanoid were very complex as they have been designed for optimal strength to weight ratio which inevitably results in a thin walled, awkward to make shape. An additional 4^{th} axis was added to the CNC machine solely for this project which made producing these parts significantly easier. The robot is made from high tensile aluminium,

carbon fibre composite tubing and carbon impregnated plastics which resulted in a robot weighing only 2.4kg.

To be able to develop a walking algorithm it is necessary for the system to monitor all aspects of the humanoid. An attitude heading reference system (AHRS) is implemented in the body to monitor body position and movement. Bioloid FSR sensors are implemented on the feet to monitor corner pressures which are used for balance and stance monitoring. All sensors were tested and various dynamic and static tests were performed to determine real world characteristics of the sensors.

WCK servo modules were used for joint actuation. These are originally found on the Robobuilder humanoid which is a hobby robot. These modules are extremely intelligent and unique as they have full tuneable PID control and functions that allow simple user control. The appropriate network protocol was established which was different to what was specified in the data sheet provided. These modules where also subjected to physical tests to determine their maximum torque output.

Visual Basic.net was used to develop appropriate software for monitoring sensors and actuating servo modules. A graphical user interface allows the user to control the position of all joints, request current servo positions, monitor AHRS and inclinometer values, and also monitor values of the FSR pressure sensors. Basically the software that has been developed is a backbone for the walking algorithm development stage. All servo modules can be positioned in either high or low resolution mode, their acceleration and max speed can be set, and their current position and load can be read easily. Protocol for all sensors has been established and their current values can be requested at any time by sending a simple packet.

Forward kinematics is a technique used to determine the resulting end effector position of a manipulator depending on known joint angles. In most cases it is more common to have a known end point but unknown joint angles. To solve this problem Inverse Kinematics is used. After reviewing many articles and various techniques for solving the inverse kinematics problem, it became apparent that a humanoid with 3D, 7 degrees of freedom and 2 manipulators/legs, the algorithm would be so complex it would take too long to solve. Also with current techniques, the number results can be infinite and therefore very hard to

determine a suitable solution. A new approach to solving the inverse kinematics using basic trigonometry was proposed. This solution can be calculate extremely quickly as it is only basic mathematics which reduces processing time. The concept has not been completely implemented however so it cannot be determined how accurate or fast the system is.

In summery; a humanoid has be designed and manufactured from scratch using unique techniques which allow the use of low cost servo actuator modules, various dynamics have been exploited to allow easier and more efficient control and to monitor body position, and foot pressure appropriate sensors were implemented. The result is a humanoid platform ready for development of a walking algorithm.

8.1 *Future work*

The upper half of the humanoid has yet to be manufactured. The arms of a human are used for balance so are necessary for developing a realistic human walking gait. Mechanically the structure is very sound except for the pushrods as they are too sloppy and also tend to break easily if abused.

Electronically the system is complete. The foot pressure sensors have been mounted and wired in, but placement of the AHRS and the inclinometer remains to be done. Once the upper body has been manufactured the controller can be mounted, at present the controller is off-board and the humanoid is running from an umbilical cord.

The control of the humanoid is the next major part of the development of this humanoid. An advanced dynamic gate is far from easy to develop as they are regarded as the ‘hardest’ control system to produce. Honda and other high budget research teams have managed to develop successful systems, but usually not without the help of a team consisting of thirty plus engineers and a multimillion dollar budget. In order to be successful in developing a dynamic walking algorithm a unique approach must also be applied to this section.

9 References

1. P. W. E. Barlow, G. Sen Gupta, S. David. (2010). *Humanoid Service Robot*. Proceedings of Electronics New Zealand Conference (ENZCon 2010), Hamilton, New Zealand, November 22-23, pp.97-102
2. P. W. E. Barlow, G. Sen Gupta, S. David. (2011). *Review of Sensors and Sensor Integration for the control of a Humanoid Robot*. IEEE International Instrumentation and Measurement Technology Conference (I2MTC 2011), Hangzhou, China, May 10-12, (To be published)
3. Chestnutt, J., M. Lau, et al. (2005). *Footstep Planning for the Honda ASIMO Humanoid*. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2005), April 18-22, pp. 629-634
4. Dede, M. I. C., S. Nasser, et al. (2005). *Cerberus the Humanoid Robot: Part I – Design.* 2005 Florida Conference on Recent Advances in Robotics, Gainsville, USA, pp. 1-8
5. Hirai, K., M. Hirose, et al. (1998). *The development of Honda humanoid robot*. Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 1998), May 16-20, Leuven, Vol. 2, pp. 1321-1326
6. Hyungjik, L., C. Ho Jin, et al. (2009). *Center of gravity based control of a humanoid balancing robot for boxing games : BalBot V*. ICCAS-SICE, Aug 18-21, Fukuoka, pp. 124-128
7. Ill-Woo, P., K. Jung-Yup, et al. (2005). *Mechanical design of humanoid robot platform KHR-3 (KAIST Humanoid Robot 3: HUBO).* 5th IEEE-RAS International Conference on Humanoid Robots, Dec 5, Tsukuba, pp. 321-326
8. Ill-Woo, P., K. Jung-Yup, et al. (2004). *Development of humanoid robot platform KHR-2 (KAIST humanoid robot-2).* 4th IEEE/RAS International Conference on Humanoid Robots, Nov 10-12, Vol. 1, pp. 292-310
9. Jun-Ho, O., H. David, et al. (2006). *Design of Android type Humanoid Robot Albert HUBO*. IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, pp. 1428-1433

10. Kagami, S., Y. Takahashi, et al. (2004). *High-speed matrix pressure sensor for humanoid robot by using thin force sensing resistance rubber sheet*. Proceedings of IEEE Sensors 2004, Oct 24-27, Vol. 3, pp. 1534-1537

11. Takahashi, Y., K. Nishiwaki, et al. (2005). *High-speed pressure sensor grid for humanoid robot foot*. 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), Aug 2-5, pp. 3909-3914

12. Kaneko, K., S. Kajita, et al. (2002). *Design of advanced leg module for humanoid robotics project of METI.* IEEE International Conference on Robotics and Automation (ICRA '02), Vol. 1, pp. 38-45

13. Kim, J.-Y., I.-W. Park, et al. (2005). *Walking Control Algorithm of Biped Humanoid Robot on Uneven and Inclined Floor*. Journal of Intelligent & Robotic Systems, Springer Verlag, Vol. 48, No. 4, pp. 457-484

14. Liquan, W., Y. Zhiwei, et al. (2006). *Influence Analysis of Toe-joint on Biped Gaits*. Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation, June 25-28, Henan, pp. 1631-1635

15. Lohmeier, S., T. Buschmann, et al. (2006). *Leg Design for a Humanoid Walking Robot*. 2006 6th IEEE-RAS International Conference on Humanoid Robots, Dec 4-6, Genova, pp. 536-541

16. Yanzhi, Z., Z. Tieshi, et al. (2007). *Performance Analysis and Optimization of Sizable 6-axis Force Sensor Based on Stewart Platform*. International Conference on Mechatronics and Automation (ICMA 2007), Aug 5-8, Harbin, pp. 2189-2193

17. Lukac, D., T. Siedel, et al. (2009). *Designing the test feet of the humanoid robot M-Series*. XXII International Symposium on Information, Communication and Automation Technologies (ICAT 2009), Oct 29-31, Bosnia, pp. 1-6

18. S.H.Park (2009). *Walking Robots.* Presented at BHR Seminar (https://research.cc.gatech.edu/humanoids/sites/edu.humanoids/files/walking_robots_sung_hyun_park.ppt)

19. Madadi, V. V. and S. Tosunoglu (2007). *Design and Development of a Biped Robot*. International Symposium on Computational Intelligence in Robotics and Automation (CIRA 2007), June 20-23, Jacksonville, USA, pp. 243-247

20. Masayuki, K., M. Takahiro, et al. (2006). *How does a Balancing Motion of a Humanoid Robots Affect a Human Motion and Impression?*, 2006 6th IEEE-RAS International Conference on Humanoid Robots, Dec 4-6, Genova, pp. 252-257

21. Minakata, H., R. Ito, et al. (2002). *An experimental study of pseudo passive walking robot-discussions about control parameters and expansion to flexible shoe system*. 7th International Workshop on Advanced Motion Control, pp. 449-454

22. Okada, M., T. Shinohara, et al. (2003). *Double spherical joint and backlash clutch for lower limbs of humanoids*. IEEE International Conference on Robotics and Automation (ICRA '03), Sept 14-19, Vol. 1, pp. 491-496

23. J. DeLope, T. Zarraonandia, R.Gonzalez-Careaga, D.Maravall (2003). *Solving the Inverse Kinematics in Humanoid Robots: A Neural Approach,* Artificial Neural Nets Problem Solving Methods, Lecture Notes in Computer Science, Springer Verlag, 2003, Vol. 2687/2003, pp. 1051-1052

24. S. Kajita (2008). *Overview of ZMP-based biped walking,* Presented in Dynamic Walking Seminar; Netherlands, 2008 (http://dynamicwalking.org/dw2008/files/presentations/DW2008_keynotepresentation_Shuuji_Kajita.pdf)

25. Osuka, K. and K. Kirihara (2000). *Motion analysis and experiments of passive walking robot QUARTET II*. IEEE International Conference on Robotics and Automation (ICRA '00), April 24-28, San Francisco, Vol. 3, pp. 3052-3056

26. Owaki, D., M. Koyama, et al. (2010). *A two-dimensional passive dynamic running biped with knees*. IEEE International Conference on Robotics and Automation (ICRA 2010), May 3-7, Anchorage, USA, pp. 5237-5242

27. Vukobratovic M., Brovac, B., Surla, D., Sotkic, D. (1990) *Biped Locomotion,* Springer-Verlag, Berlin, ISBN: 0387174567

28. Pratt, J. E. (2000). *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots*. Doctoral Thesis, Department of Electrical Engineering and Computer Science. Massachusetts Institute of Technology, June 2000

29. Santos, V. M. F. and F. M. T. Suva (2005). *Engineering solutions to build an inexpensive humanoid robot based on a distributed control architecture*., 5th IEEE-RAS International Conference on Humanoid Robots, Dec 5, Tsukuba, pp. 86-91

30. Interlink Electronics, *FSR – Interlink Electronics FSR Force Sensing Resistors.* REV.A 09/23/10

31. Interlink Electronics. *FSR - Force Sensing Resistor Integration Guide and Evaluation* Parts Catalog. Version 1.0, Revision D.

32. Sias, F. R., Jr. and Y. F. Zheng (1990). *How many degrees-of-freedom does a biped need?* IEEE International Workshop on Intelligent Robots and Systems (IROS '90), Jul 3-6, Ibaraki, Vol. 1, pp. 297-302

33. Sung-Kyun, K., H. Seokmin, et al. (2009). *A walking motion imitation framework of a humanoid robot by human walking recognition from IMU motion data*. 9th IEEE-RAS International Conference on Humanoid Robots, Dec 7-10, Paris, pp. 343-348

34. Parmiggiani, A., M. Randazzo, et al. (2009). *Joint torque sensing for the upper-body of the iCub humanoid robot.* 9th IEEE-RAS International Conference on Humanoid Robots, Dec 7-10, Paris, pp. 15-20

35. G. Welch, G. Bishop (2006), *An Introduction to the Kalman Filter,* Department of Computer Science, University of North Carolina (http://www.cs.unc.edu/~welch/media/pdf/kalman_intro.pdf)

36. Tabaczynski, M. (2005). *Jacobian Solutions to the Inverse Kinematics Problem*. Tufts University (http://www.eecs.tufts.edu/~mtabac0a/IK/Project.pdf)

37. Taira, T., N. Kamata, et al. (2005). *Design and implementation of reconfigurable modular humanoid robot architecture*. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005), Aug 2-6, pp. 3566-3571

38. Takanishi, A., Y. Ogura, et al. (2007). *Some Issues in Humanoid Robot Design*. Robotics Research. S. Thrun, R. Brooks and H. Durrant-Whyte, Springer Berlin/Heidelberg. 28: 357-372.

39. Tsay, T. I. J. and C. H. Lai (2006). *Design and Control of a Humanoid Robot*. IEEE/RSJ International Conference on Intelligent Robots and Systems, Oct 9-15, Beijing, pp. 2006-2007

40. Wollherr, D., M. Hardt, et al. (2002). *Actuator selection and hardware realization of a small and fast-moving, autonomous humanoid robot*. IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 3, pp. 2491-2496

41. Zhong, Q., Q. Pan, et al. (2009). *Design and implementation of humanoid robot HIT-2*. IEEE International Conference on Robotics and Biomimetics (ROBIO 2009), Feb 22-25, Bangkok, pp. 967-970

42. RoboBuilder. (2010). *Intelligent Modular Robot*. WCK series User Manual ver 1.07, User Manual

43. J. H. Kim, S. W. Park, I. W. Park, and J. H. Oh (2002) *Development of a Humanoid Biped Walking Robot Platform KHR-1 –Initial Design and Its Performance Evaluation-*, 3rd IARP International Workshop on Humanoid and Human Friendly Robotics, pp.14–21

44. *Workshop on Mechanical Design of Humanoids.* Workshop of the 9th IEEE RAS International Conference on Humanoid Robots (Humanoids09), Paris, December 7-10, 2009.

45. D. G. Caldwell, N. Tsagarakis, D. Badihi and G. A. Medrano-Cerda (1998). *Pneumatic Muscle Actuation Technology a light weight power system for a Humanoid Robot,* IEEE International Conference on Robotics and Automation (ICRA 1998), May 16-20, Leuven, Vol. 4, pp. 3053-3058

46. F. Vecchi, C. Freschi, S. Micera, A. Sabatini, P. Dario, and R. Sacchetti (2000). *Experimental Evaluation of Two Commercial Force Sensors for Applications in Biomechanics and Motor Control*. 5th Annual Conference of the International Functional Electrical Stimulation Society, Aalborg, Danemark, pp. 44–44

47. A. Hollinger, M.M. Wanderley (2006). *Evaluation of Commercial ForceSensing Resistors***.** 2006 International Conference on New Interfaces for Musical Expression (NIME06), June 4-8, Paris.

48. Aydemir, D. and H. Iba (2006). *Evolutionary Behavior Acquisition for Humanoid Robots.* Parallel Problem Solving from Nature - T. Runarsson, H.-G. Beyer, E. Burkeet al, Springer Berlin / Heidelberg. 4193: 651-660

49. Tevatia, G. and S. Schaal (2000). *Inverse kinematics for humanoid robots.* IEEE International Conference on Robotics and Automation (ICRA '00), Apr 24-28, San Francisco, Vol. 1, pp. 294-299

50. L. C. T. Wang and C. C. Chen (1991). *A combined optimization method for solving the inverse kinematics problem of mechanical manipulators.* IEEE Transactions on Robotics and Automation, Vol. 7, pp. 489-499

51. D. E. Whitney (1969). *Resolved motion rate control of manipulators and human prostheses.* IEEE Transactions on Man-Machine Systems, Vol. 10, Issue 2, pp. 47-53

52. C. W. Wampler (1986). *Manipulator inverse kinematic solutions based on vector formulations and damped least squares methods.* IEEE Transactions on Systems, Man, and Cybernetics, Vol. 16, Issue 1, pp. 93-101

53. A. Balestrino, G. De Maria, and L. Sciavicco (1984). *Robust control of robotic manipulators*, 9th IFAC World Congress, Budapest, pp. 1359-1363

54. R. Grzeszczuk and D. Terzopoulos (1995). *Automated learning of muscle actuated locomotion through control abstraction*, 22nd annual conference on Computer graphics and interactive techniques (SIG GRAPH'95), New York, ACM Press, pp. 63-70

55. A. D'Souza, S. Vijayakumar, and S. Schaal (2001). *Learning inverse kinematics*. IEEE/RSJ International Conference on Intelligent Robots and Systems, Vol. 1, pp. 298-303

56. Cheah, C. C., C. Liu, et al. (2004). *Stability of inverse Jacobian control for robot manipulator.* IEEE International Conference on Control Applications, Vol 1, pp. 321-326

Glossary

AHRS- Attitude Heading Reference System, as sensors for measuring and monitoring orientation.

Axis - The line about which a rotating body (such as a tool) turns.

Computer-Aided Design - (CAD) Computer software is used to develop and/or alter all aspects of design such as the end product, the system, machinery used etc.

Computer-Aided Manufacturing - (CAM) Computer software is used to design and/or alter the manufacturing process.

Degrees of Freedom - (DOF) The number of independent variables in the system. Each joint in a serial robot represents a degree of freedom.

End-effector - The application tools located at the end of the robot's arm and are typically connected to robot flanges.

FSR- Force sensing resistor

Jacobian - End-effector velocity is related to joint speeds by this matrix of first-order partial derivatives.

Inclinometer- Sensor for measuring angle, or incline.

Kinematics - Analysis of motion by leaving out information of forces.

Pseudoinverse - The inversion of a non square matrix used with joint speeds to minimize the magnitude of a vector.

Pitch – If viewing it from a planes point of view, pitch is the angle of attach. Ie climbing or descending

Roll - If viewing it from a planes point of view, roll is literally rolling or banking angle.

Yaw - If viewing it from a planes point of view, yaw is the vertical angle or drift. Also known as heading.

Appendices

9.1 *Appendix A*

9.1.1 Software:

```
Imports System
Public Class Form1
    Public ID, Speed, Acceleration As UInt16
    Public Angle, RID, RKneeAng, LKneeAng As Integer
    Private temp(8) As Byte
    Private RightToeArray(8), RightAnkleLArray(8), RightAnkleRArray(8),
RightKneeArray(8), RightHipLArray(8), RightHipRArray(8),
RightHipSimArray(8), RightHipVerticalArray(8) As Byte
    Private LeftToeArray(8), LeftAnkleLArray(8), LeftAnkleRArray(8),
LeftKneeArray(8), LeftHipLArray(8), LeftHipRArray(8), LeftHipSimArray(8),
LeftHipVerticalArray(8) As Byte
    Private PelvisHorizontalArray(8), PelvisVerticalArray(8) As Byte


    Private Declare Sub Sleep Lib "kernel32" (ByVal dwMilliseconds As Long)

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim n As Integer

        For n = n To 3

            Homing()

            If SerialPort1.IsOpen = False Then
                Me.SerialPort1.Open()
            End If
            SendOUTSerial()
        Next n


    End Sub
    Public Sub Homing()
        Speed = 2
        Acceleration = 20
        Angle = 512
        RightToe()
        LeftToe()
        Angle = 512
        LeftAnkleL()
        Angle = 512
        LeftAnkleR()
        Angle = 512
        RightAnkleL()
        Angle = 512
        RightAnkleR()
        Speed = 4
        Angle = 375
        RightKnee()
        Angle = 375
        LeftKnee()
        Speed = 3
        Acceleration = 40
```

```
        Angle = 512
        RightHipL()
        RightHipR()
        LeftHipL()
        LeftHipR()
        RightHipVertical()
        LeftHipVertical()
        PelvisHorizontal()
        PelvisVertical()
    End Sub

    '________RIGHT
LEG_____________________________________________________________________
___
    Private Sub RightToe()
        ID = 11
        SpeedANDAcceleration()
        AngleSend()
        RightToeArray = temp.Clone()
    End Sub
    Private Sub RightAnkleL()
        ID = 13
        SpeedANDAcceleration()
        AngleSend()
        RightAnkleLArray = temp.Clone()
    End Sub
    Private Sub RightAnkleR()
        ID = 12
        SpeedANDAcceleration()
        AngleSend()
        RightAnkleRArray = temp.Clone()
    End Sub

    Public Sub RightKnee()

        ID = 14
        SpeedANDAcceleration()
        AngleSend()
        RightKneeArray = temp.Clone()

    End Sub
    Public Sub RightHipL()
        ID = 17
        SpeedANDAcceleration()
        AngleSend()
        RightHipLArray = temp.Clone()
    End Sub
    Public Sub RightHipR()
        ID = 16
        SpeedANDAcceleration()
        AngleSend()
        RightHipRArray = temp.Clone()
    End Sub

    Public Sub RightHipVertical()
        ID = 18
        SpeedANDAcceleration()
        AngleSend()
        RightHipVerticalArray = temp.Clone()
    End Sub
```

```
    '________Left
LEG_____________________________________________________________________
___
    Public Sub LeftToe()
        ID = 1
        SpeedANDAcceleration()
        AngleSend()
        LeftToeArray = temp.Clone()
    End Sub
    Public Sub LeftAnkleL()
        ID = 2
        SpeedANDAcceleration()
        AngleSend()
        LeftAnkleLArray = temp.Clone()
    End Sub
    Public Sub LeftAnkleR()
        ID = 3
        SpeedANDAcceleration()
        AngleSend()
        LeftAnkleRArray = temp.Clone()
    End Sub

    Public Sub LeftKnee()
        ID = 4
        SpeedANDAcceleration()
        AngleSend()
        LeftKneeArray = temp.Clone()
    End Sub
    Public Sub LeftHipL()
        ID = 6
        SpeedANDAcceleration()
        AngleSend()
        LeftHipLArray = temp.Clone()
    End Sub
    Public Sub LeftHipR()
        ID = 7
        SpeedANDAcceleration()
        AngleSend()
        LeftHipRArray = temp.Clone()
    End Sub

    Public Sub LeftHipVertical()
        ID = 8
        SpeedANDAcceleration()
        AngleSend()
        LeftHipVerticalArray = temp.Clone()
    End Sub

'________Pelvis_____________________________________________________________
_____________
    Public Sub PelvisHorizontal()
        ID = 20
        SpeedANDAcceleration()
        AngleSend()
        PelvisHorizontalArray = temp.Clone()
    End Sub
    Public Sub PelvisVertical()
        ID = 21
        SpeedANDAcceleration()
        AngleSend()
        PelvisVerticalArray = temp.Clone()
```

```
    End Sub

    '________MAIN-SPEED & ANGLE
SUBS_____________________________________________________________________
____

    Public Sub SpeedANDAcceleration()
        Dim SpeedAccChecksum As Byte
        Dim ModeID As UInt16

        ModeID = ID + 224

        Dim SpeedAccArray(5) As Byte
        SpeedAccArray(0) = 255
        SpeedAccArray(1) = ModeID
        SpeedAccArray(2) = 13
        SpeedAccArray(3) = Speed
        SpeedAccArray(4) = Acceleration
        SpeedAccChecksum = (ModeID Xor 13 Xor Speed Xor Acceleration) And
127
        SpeedAccArray(5) = SpeedAccChecksum

        If Me.SerialPort1.IsOpen = False Then
            Me.SerialPort1.Open()
        End If

        SerialPort1.Write(SpeedAccArray, 0, 6)
    End Sub

    Public Sub AngleSend()
        Dim Checksum As Byte
        Dim L, H As String
        Dim xL, xH As String

        Dim ByteAngle As Byte() = BitConverter.GetBytes(Angle)
        Dim str As String = BitConverter.ToString(ByteAngle)

        Dim myBinaryString As String = Convert.ToString(Angle, 2)
        Dim myBitArray As New BitArray(myBinaryString.Length)

        While myBinaryString.Length < 16
            myBinaryString = "0" & myBinaryString
        End While
        Dim HighLowByte(16) As Byte

        Dim n As Integer

        For n = 0 To 15
            HighLowByte(n) = myBinaryString.Substring(n, 1)
        Next

        Dim xHArray(7), xLArray(7) As Byte
        For n = 0 To 7
            xHArray(n) = HighLowByte(n + 1)
        Next
        For n = 8 To 15
            xLArray(n - 8) = HighLowByte(n)
        Next
        xLArray(0) = 0

        For n = 0 To 7
```

```
            H = H & xHArray(n).ToString
            L = L & xLArray(n).ToString
        Next
        xH = Convert.ToInt32(H, 2)
        xL = Convert.ToInt32(L, 2)


        temp(0) = 255
        temp(1) = 224
        temp(2) = 200
        temp(3) = ID
        temp(4) = 0
        temp(5) = xH
        temp(6) = xL
        Checksum = (ID Xor 0 Xor xH Xor xL Xor 224 Xor 200) And 127
        temp(7) = Checksum


    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click

        Dim ListArray As ArrayList = New ArrayList(1)
        Dim n As Integer

        For n = n To 3

            Speed = 2
            Acceleration = 30

            Angle = 370
            RightHipL()
            RightHipR()
            LeftHipL()
            LeftHipR()
            Speed = 2
            Acceleration = 20
            Angle = 560
            LeftKnee()
            Angle = 560
            RightKnee()
            Angle = 350
            LeftAnkleL()
            LeftAnkleR()
            RightAnkleL()
            RightAnkleR()


            If SerialPort1.IsOpen = False Then
                Me.SerialPort1.Open()
            End If

            Me.SerialPort1.Write(LeftAnkleLArray, 0, 8)
            Me.SerialPort1.Write(LeftAnkleRArray, 0, 8)
            Me.SerialPort1.Write(RightAnkleLArray, 0, 8)
            Me.SerialPort1.Write(RightAnkleRArray, 0, 8)
            Me.SerialPort1.Write(RightKneeArray, 0, 8)
            Me.SerialPort1.Write(LeftKneeArray, 0, 8)
            Me.SerialPort1.Write(LeftHipLArray, 0, 8)
            Me.SerialPort1.Write(LeftHipRArray, 0, 8)
```

```
            Me.SerialPort1.Write(RightHipLArray, 0, 8)
            Me.SerialPort1.Write(RightHipRArray, 0, 8)
        Next n

    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
        ListBox1.Items.Clear()
        RID = 1
        For RID = RID To 4
            RequestServo()
        Next
        RID = 6
        For RID = RID To 8
            RequestServo()
        Next
        RID = 11
        For RID = RID To 14
            RequestServo()
        Next
        RID = 16
        For RID = RID To 18
            RequestServo()
        Next
        RID = 20
        RequestServo()
        RID = 21
        RequestServo()
    End Sub

    Public Sub RequestServo()

        Dim RChecksum As Byte
        Dim RxIncoming(1) As Byte
        Dim RxByte As Integer
        Dim Receive(3) As Byte

        If Me.SerialPort1.IsOpen = True Then
            Me.SerialPort1.Close()
        End If

        RxByte = 0
        While RxByte = 0

            Receive(0) = 255
            Receive(1) = 160 + RID
            Receive(2) = 0
            RChecksum = ((160 + RID) Xor 0) And 127
            Receive(3) = RChecksum


            If Me.SerialPort1.IsOpen = False Then
                Me.SerialPort1.Open()
            End If
            Me.SerialPort1.DiscardInBuffer()
```

```
            Me.SerialPort1.DiscardOutBuffer()

            Me.SerialPort1.Write(Receive, 0, 4)
            Sleep(100)
            Try
                Me.SerialPort1.Read(RxIncoming, 0, 2)

            Catch ex As ArgumentException

                Me.ListBox1.Items.Add("No Servo")
            Finally
                RxByte = RxIncoming(1).ToString
                RxIncoming(0) = 0
                RxIncoming(1) = 0

            End Try

        End While

        RxByte = RxByte * 4
        Me.ListBox1.Items.Add("ID " & RID & " ---- " & RxByte)
        Me.SerialPort1.DiscardOutBuffer()

    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
        Using SW As New IO.StreamWriter("C:\Documents and Settings\New
user\Desktop\listBox1.txt", True)
            For Each itm As String In Me.ListBox1.Items
                SW.WriteLine(itm)
            Next
        End Using
    End Sub
    Private Sub SendOUTSerial()
        Me.SerialPort1.Write(RightToeArray, 0, 8)
        Me.SerialPort1.Write(LeftToeArray, 0, 8)
        Me.SerialPort1.Write(LeftAnkleLArray, 0, 8)
        Me.SerialPort1.Write(LeftAnkleRArray, 0, 8)
        Me.SerialPort1.Write(RightAnkleLArray, 0, 8)
        Me.SerialPort1.Write(RightAnkleRArray, 0, 8)
        Me.SerialPort1.Write(RightKneeArray, 0, 8)
        Me.SerialPort1.Write(LeftKneeArray, 0, 8)
        Me.SerialPort1.Write(LeftHipLArray, 0, 8)
        Me.SerialPort1.Write(LeftHipRArray, 0, 8)
        Me.SerialPort1.Write(RightHipLArray, 0, 8)
        Me.SerialPort1.Write(RightHipRArray, 0, 8)
        Me.SerialPort1.Write(LeftHipVerticalArray, 0, 8)
        Me.SerialPort1.Write(RightHipVerticalArray, 0, 8)
        Me.SerialPort1.Write(PelvisHorizontalArray, 0, 8)
        Me.SerialPort1.Write(PelvisVerticalArray, 0, 8)
    End Sub

    Public Sub Init()
        RightToeArray(0) = 0
        LeftToeArray(0) = 0
        LeftAnkleLArray(0) = 0
        LeftAnkleRArray(0) = 0
        RightAnkleLArray(0) = 0
        RightAnkleRArray(0) = 0
        RightKneeArray(0) = 0
```

```
        LeftKneeArray(0) = 0
        LeftHipLArray(0) = 0
        LeftHipRArray(0) = 0
        RightHipLArray(0) = 0
        RightHipRArray(0) = 0
        LeftHipVerticalArray(0) = 0
        RightHipVerticalArray(0) = 0
        PelvisHorizontalArray(0) = 0
        PelvisVerticalArray(0) = 0
    End Sub
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        'LeftSide()

        RightSide()
        'Sleep(1000)
        'FirstStepLeftLeg()
        'Sleep(1000)
        'COGtoLeftLeg()
    End Sub
    Private Sub LeftSide()
        Dim n As Byte
        Init()
        Acceleration = 20
        Speed = 1
        For n = n To 3
            Angle = 534
            LeftAnkleL()
            Angle = 475
            LeftAnkleR()
            Angle = 400
            LeftKnee()
            Angle = 436
            LeftHipL()
            Angle = 480
            LeftHipR()

            Angle = 532
            RightAnkleL()
            Angle = 475
            RightAnkleR()
            Angle = 400
            RightKnee()
            Angle = 480
            RightHipR()
            Angle = 436
            RightHipL()
            Angle = 492
            RightHipVertical()
            Speed = 6
            Angle = 712
            PelvisHorizontal()
            Angle = 512
            PelvisVertical()
            Speed = 1
            SendOUTSerial()
        Next
    End Sub
    Private Sub RightSide()
        Dim n As Byte
        Init()
```

```
    Acceleration = 20
    Speed = 1
    For n = n To 3
        Angle = 436
        LeftAnkleL()
        Angle = 552
        LeftAnkleR()
        Angle = 375
        LeftKnee()
        Angle = 556
        LeftHipL()
        Angle = 484
        LeftHipR()

        Angle = 456
        RightAnkleL()
        Angle = 542
        RightAnkleR()
        Angle = 375
        RightKnee()
        Angle = 456
        RightHipR()
        Angle = 484
        RightHipL()


        Speed = 6
        Angle = 300
        PelvisHorizontal()
        Angle = 512
        PelvisVertical()
        Speed = 1
        SendOUTSerial()
    Next
End Sub
Public Sub FirstStepLeftLeg()
    Dim n As Byte
    Init()
    Acceleration = 20
    Speed = 2
    For n = n To 3
        Angle = 514
        LeftAnkleL()
        Angle = 510
        LeftAnkleR()
        Angle = 400
        LeftKnee()
        Angle = 420
        LeftHipL()
        Angle = 420
        LeftHipR()

        Angle = 452
        RightAnkleL()
        Angle = 462
        RightAnkleR()
        Angle = 380
        RightKnee()
        Angle = 570
        RightHipR()
        Angle = 570
```

```
        RightHipL()
        Angle = 492
        RightHipVertical()

        Angle = 500
        RightToe()
        Speed = 6
        Angle = 300
        PelvisHorizontal()
        Angle = 512
        PelvisVertical()
        SendOUTSerial()
    Next
End Sub
Public Sub COGtoLeftLeg()
    Dim n As Byte
    Init()
    Acceleration = 20
    Speed = 2

    Angle = 514
    LeftAnkleL()
    Angle = 510
    LeftAnkleR()
    Angle = 400
    LeftKnee()
    Angle = 420
    LeftHipL()
    Angle = 420
    LeftHipR()

    Angle = 552
    RightAnkleL()
    Angle = 562
    RightAnkleR()
    Sleep(500)
    Angle = 410
    RightKnee()
    Angle = 570
    RightHipR()
    Angle = 570
    RightHipL()
    Angle = 492
    RightHipVertical()

    Angle = 450
    RightToe()
    Speed = 6
    Angle = 750
    PelvisHorizontal()
    Angle = 512
    PelvisVertical()
    SendOUTSerial()

End Sub
Private Sub RightLegForward()
    Dim n As Byte
    Init()
    Acceleration = 20
    Speed = 1
    For n = n To 3
```

```
            Angle = 475
            LeftAnkleL()
            Angle = 532
            LeftAnkleR()
            Angle = 400
            LeftKnee()
            Angle = 480
            LeftHipL()
            Angle = 436
            LeftHipR()

            Angle = 475
            RightAnkleL()
            Angle = 532
            RightAnkleR()
            Angle = 400
            RightKnee()
            Angle = 436
            RightHipR()
            Angle = 480
            RightHipL()


            Speed = 6
            Angle = 300
            PelvisHorizontal()
            Angle = 512
            PelvisVertical()
            Speed = 1
            SendOUTSerial()

        Next
    End Sub

    Private Sub Button6_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button6.Click
        RightLegForward()

    End Sub

    Private Sub Button7_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button7.Click

        FirstStepLeftLeg()

    End Sub

    Private Sub Button8_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button8.Click

        COGtoLeftLeg()
    End Sub

    Private Sub Button9_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button9.Click
        ID = 3
        Angle = 1016
        AngleSend()

    End Sub
End Class
```

9.2 *Appendix B*

9.2.1 Machine code for the Pelvis:

```
%
O5000 /* PELVIS-1.CNC */
/* 30-JAN-2011*/
G90 G17
G80 G49 G40
G54
G91 G28 Z0
G90
M01
N1 M6 T16
T3
/* TOOL -16- MILL DIA 8.0 R0. MM
*/
G90 G00 G40 G54
G43 H16 D16 G0 X79.439 Y-5. Z70.
S3000 M3
M8
/*-----------------------*/
/*F-CONTOUR-T16 - PROFILE*/
/*-----------------------*/
  X79.439 Y-5. Z50.
  Z0.
G1 Z-5. F100
  X126.473 F450
G3 X167.314 Y3.268 R105.
  X178.056 Y9.911 R34.324
G2 X178.891 Y10.159 R1.
G3 X189.419 Y31.263 R12.344
G2 X189.115 Y32.079 R1.
G3 X165.185 Y68.045 R34.324
G2 X164.551 Y68.638 R1.
G3 X141.351 Y67.224 R12.198
G2 X140.794 Y66.558 R1.
G3 X127.654 Y56.108 R34.324
G1 X122.177 Y48.995
G2 X120.389 Y49.51 R1.
G1 X117.818 Y76.372
G3 X109.854 Y83.61 R8.
G1 X80.38
G3 X72.417 Y76.372 R8.
G1 X69.845 Y49.51
G2 X68.057 Y48.995 R1.
G1 X62.581 Y56.108
G3 X49.441 Y66.558 R34.324
G2 X48.884 Y67.224 R1.
G3 X25.684 Y68.638 R12.198
G2 X25.05 Y68.045 R1.
G3 X1.12 Y32.079 R34.324
G2 X0.815 Y31.263 R1.
G3 X11.343 Y10.159 R12.344
G2 X12.178 Y9.911 R1.
G3 X22.921 Y3.268 R34.324
  X63.761 Y-5. R105.
G1 X79.439
  Z-8. F100
  X126.473 F450
G3 X167.314 Y3.268 R105.
  X178.056 Y9.911 R34.324
G2 X178.891 Y10.159 R1.
G3 X189.419 Y31.263 R12.344
G2 X189.115 Y32.079 R1.
G3 X165.185 Y68.045 R34.324
G2 X164.551 Y68.638 R1.
G3 X141.351 Y67.224 R12.198
G2 X140.794 Y66.558 R1.
G3 X127.654 Y56.108 R34.324
G1 X122.177 Y48.995
G2 X120.389 Y49.51 R1.
G1 X117.818 Y76.372
G3 X109.854 Y83.61 R8.
G1 X80.38
G3 X72.417 Y76.372 R8.
G1 X69.845 Y49.51
G2 X68.057 Y48.995 R1.
G1 X62.581 Y56.108
G3 X49.441 Y66.558 R34.324
G2 X48.884 Y67.224 R1.
G3 X25.684 Y68.638 R12.198
G2 X25.05 Y68.045 R1.
G3 X1.12 Y32.079 R34.324
G2 X0.815 Y31.263 R1.
G3 X11.343 Y10.159 R12.344
G2 X12.178 Y9.911 R1.
G3 X22.921 Y3.268 R34.324
  X63.761 Y-5. R105.
G1 X79.439
  Z-11. F100
  X126.473 F450
G3 X167.314 Y3.268 R105.
  X178.056 Y9.911 R34.324
G2 X178.891 Y10.159 R1.
G3 X189.419 Y31.263 R12.344
G2 X189.115 Y32.079 R1.
G3 X165.185 Y68.045 R34.324
G2 X164.551 Y68.638 R1.
G3 X141.351 Y67.224 R12.198
G2 X140.794 Y66.558 R1.
G3 X127.654 Y56.108 R34.324
G1 X122.177 Y48.995
G2 X120.389 Y49.51 R1.
G1 X117.818 Y76.372
G3 X109.854 Y83.61 R8.
G1 X80.38
G3 X72.417 Y76.372 R8.
G1 X69.845 Y49.51
G2 X68.057 Y48.995 R1.
G1 X62.581 Y56.108
G3 X49.441 Y66.558 R34.324
G2 X48.884 Y67.224 R1.
G3 X25.684 Y68.638 R12.198
G2 X25.05 Y68.045 R1.
G3 X1.12 Y32.079 R34.324
G2 X0.815 Y31.263 R1.
G3 X11.343 Y10.159 R12.344
G2 X12.178 Y9.911 R1.
G3 X22.921 Y3.268 R34.324
  X63.761 Y-5. R105.
G1 X79.439
  Z-12. F100
  X126.473 F450
G3 X167.314 Y3.268 R105.
  X178.056 Y9.911 R34.324
G2 X178.891 Y10.159 R1.
G3 X189.419 Y31.263 R12.344
G2 X189.115 Y32.079 R1.
G3 X165.185 Y68.045 R34.324
G2 X164.551 Y68.638 R1.
G3 X141.351 Y67.224 R12.198
G2 X140.794 Y66.558 R1.
G3 X127.654 Y56.108 R34.324
G1 X122.177 Y48.995
G2 X120.389 Y49.51 R1.
G1 X117.818 Y76.372
G3 X109.854 Y83.61 R8.
G1 X80.38
G3 X72.417 Y76.372 R8.
G1 X69.845 Y49.51
G2 X68.057 Y48.995 R1.
G1 X62.581 Y56.108
G3 X49.441 Y66.558 R34.324
G2 X48.884 Y67.224 R1.
G3 X25.684 Y68.638 R12.198
G2 X25.05 Y68.045 R1.
G3 X1.12 Y32.079 R34.324
G2 X0.815 Y31.263 R1.
G3 X11.343 Y10.159 R12.344
G2 X12.178 Y9.911 R1.
G3 X22.921 Y3.268 R34.324
  X63.761 Y-5. R105.
G1 X79.439
  Y-4. F450
  X126.473
G3 X166.939 Y4.195 R104.
  X177.383 Y10.65 R33.324
G2 X179.053 Y11.145 R2.
G3 X188.728 Y30.54 R11.344
G2 X188.119 Y32.173 R2.
G3 X164.886 Y67.09 R33.324
G2 X163.618 Y68.276 R2.
G3 X142.32 Y66.978 R11.198
G2 X141.206 Y65.647 R2.
G3 X128.447 Y55.499 R33.324
G1 X122.97 Y48.385
G2 X119.394 Y49.414 R2.
G1 X116.822 Y76.277
G3 X109.854 Y82.61 R7.
G1 X80.38
G3 X73.412 Y76.277 R7.
G1 X70.84 Y49.414
G2 X67.265 Y48.385 R2.
G1 X61.787 Y55.499
G3 X49.029 Y65.647 R33.324
G2 X47.914 Y66.978 R2.
G3 X26.616 Y68.276 R11.198
G2 X25.348 Y67.09 R2.
G3 X2.115 Y32.173 R33.324
G2 X1.506 Y30.54 R2.
G3 X11.181 Y11.145 R11.344
G2 X12.852 Y10.65 R2.
G3 X23.295 Y4.195 R33.324
  X63.761 Y-4. R104.
G1 X79.439
  Y-5.
G0 Z50.
M9
G91 G28 Z0
G90
M01
N2 M6 T3
T13
/* TOOL -3- MILL DIA 6.0 R0. MM
*/
G90 G00 G40 G54
G43 H3 D3 G0 X99.755 Y51.879 Z70.
S1500 M3
M8
/*-----------------------*/
/*P-CONTOUR1-T3 - POCKET*/
/*-----------------------*/
  X99.755 Y51.879 Z50.
  Z0.
G1 Z-5. F80
G2 X96.355 Y51.453 R13.8 F170
G1 X93.88
```

G2 X90.48 Y51.879 R13.8
G1 Y51.268
X99.755
Y51.879
X99.237 Y53.914
G2 X96.355 Y53.553 R11.7
G1 X93.88
G2 X88.38 Y54.927 R11.7
G1 Y49.168
X101.855
Y54.927
G2 X99.237 Y53.914 R11.7
G1 X98.72 Y55.949
G2 X96.355 Y55.653 R9.6
G1 X93.88
G2 X86.28 Y59.388 R9.6
G1 Y47.068
X103.955
Y59.388
G2 X98.72 Y55.949 R9.6
G1 X98.202 Y57.984
G2 X96.355 Y57.753 R7.5
G1 X93.88
G2 X86.38 Y65.253 R7.5
G1 Y71.056
G3 X84.18 Y71.056 R-1.1
G1 Y46.068
G3 X85.28 Y44.968 R1.1
G1 X104.955
G3 X106.055 Y46.068 R1.1
G1 Y71.056
G3 X103.855 Y71.056 R-1.1
G1 Y65.253
G2 X98.202 Y57.984 R7.5
G0 Z50.
X99.755 Y51.879
Z-3.
G1 Z-6.5 F80
G2 X96.355 Y51.453 R13.8 F170
G1 X93.88
G2 X90.48 Y51.879 R13.8
G1 Y51.268
X99.755
Y51.879
X99.237 Y53.914
G2 X96.355 Y53.553 R11.7
G1 X93.88
G2 X88.38 Y54.927 R11.7
G1 Y49.168
X101.855
Y54.927
G2 X99.237 Y53.914 R11.7
G1 X98.72 Y55.949
G2 X96.355 Y55.653 R9.6
G1 X93.88
G2 X86.28 Y59.388 R9.6
G1 Y47.068
X103.955
Y59.388
G2 X98.72 Y55.949 R9.6
G1 X98.202 Y57.984
G2 X96.355 Y57.753 R7.5
G1 X93.88
G2 X86.38 Y65.253 R7.5
G1 Y71.056
G3 X84.18 Y71.056 R-1.1
G1 Y46.068
G3 X85.28 Y44.968 R1.1
G1 X104.955
G3 X106.055 Y46.068 R1.1
G1 Y71.056
G3 X103.855 Y71.056 R-1.1
G1 Y65.253
G2 X98.202 Y57.984 R7.5
G0 Z50.
X99.755 Y51.879
Z-4.5
G1 Z-7. F80
G2 X96.355 Y51.453 R13.8 F170
G1 X93.88
G2 X90.48 Y51.879 R13.8
G1 Y51.268
X99.755
Y51.879
X99.237 Y53.914
G2 X96.355 Y53.553 R11.7
G1 X93.88
G2 X88.38 Y54.927 R11.7
G1 Y49.168
X101.855
Y54.927
G2 X99.237 Y53.914 R11.7
G1 X98.72 Y55.949
G2 X96.355 Y55.653 R9.6
G1 X93.88
G2 X86.28 Y59.388 R9.6
G1 Y47.068
X103.955
Y59.388
G2 X98.72 Y55.949 R9.6
G1 X98.202 Y57.984
G2 X96.355 Y57.753 R7.5
G1 X93.88
G2 X86.38 Y65.253 R7.5
G1 Y71.056
G3 X84.18 Y71.056 R-1.1
G1 Y46.068
G3 X85.28 Y44.968 R1.1
G1 X104.955
G3 X106.055 Y46.068 R1.1
G1 Y71.056
G3 X103.855 Y71.056 R-1.1
G1 Y65.253
G2 X98.202 Y57.984 R7.5
G1 X98.079 Y58.469 F170
G2 X96.355 Y58.253 R7.
G1 X93.88
G2 X86.88 Y65.253 R7.
G1 Y71.056
G3 X83.68 Y71.056 R-1.6
G1 Y46.068
G3 X85.28 Y44.468 R1.6
G1 X104.955
G3 X106.555 Y46.068 R1.6
G1 Y71.056
G3 X103.355 Y71.056 R-1.6
G1 Y65.253
G2 X98.079 Y58.469 R7.
G0 Z50.
X99.755 Y28.29
Z0.
G1 Z-5. F80
Y29.59 F170
X90.48
Y28.29
G2 X93.88 Y28.715 R13.8
G1 X96.355
G2 X99.755 Y28.29 R13.8
G1 X99.237 Y26.255
G2 X101.855 Y25.242 R11.7
G1 Y31.69
X88.38
Y25.242
G2 X93.88 Y26.615 R11.7
G1 X96.355
G2 X99.237 Y26.255 R11.7
G1 X98.72 Y24.219
G2 X103.955 Y20.781 R9.6
G1 Y33.79
X86.28
Y20.781
G2 X93.88 Y24.515 R9.6
G1 X96.355
G2 X98.72 Y24.219 R9.6
G1 X98.202 Y22.184
G2 X103.855 Y14.915 R7.5
G1 Y8.79
G3 X106.055 Y8.79 R-1.1
G1 Y34.79
G3 X104.955 Y35.89 R1.1
G1 X85.28
G3 X84.18 Y34.79 R1.1
G1 Y8.79
G3 X86.38 Y8.79 R-1.1
G1 Y14.915
G2 X93.88 Y22.415 R7.5
G1 X96.355
G2 X98.202 Y22.184 R7.5
G0 Z50.
X99.755 Y28.29
Z-3.
G1 Z-6.5 F80
Y29.59 F170
X90.48
Y28.29
G2 X93.88 Y28.715 R13.8
G1 X96.355
G2 X99.755 Y28.29 R13.8
G1 X99.237 Y26.255
G2 X101.855 Y25.242 R11.7
G1 Y31.69
X88.38
Y25.242
G2 X93.88 Y26.615 R11.7
G1 X96.355
G2 X99.237 Y26.255 R11.7
G1 X98.72 Y24.219
G2 X103.955 Y20.781 R9.6
G1 Y33.79
X86.28
Y20.781
G2 X93.88 Y24.515 R9.6
G1 X96.355
G2 X98.72 Y24.219 R9.6
G1 X98.202 Y22.184
G2 X103.855 Y14.915 R7.5
G1 Y8.79
G3 X106.055 Y8.79 R-1.1
G1 Y34.79
G3 X104.955 Y35.89 R1.1
G1 X85.28
G3 X84.18 Y34.79 R1.1
G1 Y8.79
G3 X86.38 Y8.79 R-1.1
G1 Y14.915
G2 X93.88 Y22.415 R7.5
G1 X96.355
G2 X98.202 Y22.184 R7.5
G0 Z50.
X99.755 Y28.29
Z-4.5
G1 Z-7. F80
Y29.59 F170
X90.48
Y28.29
G2 X93.88 Y28.715 R13.8
G1 X96.355
G2 X99.755 Y28.29 R13.8
G1 X99.237 Y26.255
G2 X101.855 Y25.242 R11.7
G1 Y31.69
X88.38
Y25.242
G2 X93.88 Y26.615 R11.7
G1 X96.355

```
G2 X99.237 Y26.255 R11.7
G1 X98.72 Y24.219
G2 X103.955 Y20.781 R9.6
G1 Y33.79
  X86.28
  Y20.781
G2 X93.88 Y24.515 R9.6
G1 X96.355
G2 X98.72 Y24.219 R9.6
G1 X98.202 Y22.184
G2 X103.855 Y14.915 R7.5
G1 Y8.79
G3 X106.055 Y8.79 R-1.1
G1 Y34.79
G3 X104.955 Y35.89 R1.1
G1 X85.28
G3 X84.18 Y34.79 R1.1
G1 Y8.79
G3 X86.38 Y8.79 R-1.1
G1 Y14.915
G2 X93.88 Y22.415 R7.5
G1 X96.355
G2 X98.202 Y22.184 R7.5
G1 X98.079 Y21.7 F170
G2 X103.355 Y14.915 R7.
G1 Y8.79
G3 X106.555 Y8.79 R-1.6
G1 Y34.79
G3 X104.955 Y36.39 R1.6
G1 X85.28
G3 X83.68 Y34.79 R1.6
G1 Y8.79
G3 X86.88 Y8.79 R-1.6
G1 Y14.915
G2 X93.88 Y21.915 R7.
G1 X96.355
G2 X98.079 Y21.7 R7.
G0 Z50.
M9
G91 G28 Z0
G90
M01
N3 M6 T13
T3
/* TOOL -13- MILL DIA 5.0 R0. MM
*/
G90 G00 G40 G54
G43 H13 D13 G0 X85.28 Y71.056
Z70. S1000 M3
M8
/*-----------------*/
/*S-SLOT-T13 - SLOT*/
/*-----------------*/
  X85.28 Y71.056 Z50.
  Z0.
G1 Z-12. F33
  Y46.068 F100
G0 Z50.
  Y34.79
  Z0.
G1 Z-12. F33
  Y8.79 F100
G0 Z50.
  X104.955 Y71.056
  Z0.
G1 Z-12. F33
  Y46.068 F100
G0 Z50.
  Y34.79
  Z0.
G1 Z-12. F33
  Y8.79 F100
G0 Z50.
M9
G91 G28 Z0
G90
M01
N4 M6 T3
T1
/* TOOL -3- MILL DIA 6.0 R0. MM
*/
G90 G00 G40 G54
G43 H3 D3 G0 X71.48 Y23.127 Z70.
S1500 M3
M8
/*----------------------*/
/*P-CONTOUR2-T3 - POCKET*/
/*----------------------*/
  X71.48 Y23.127 Z50.
  Z0.
G1 Z-5. F80
  Y32.264 F170
  X69.123
G2 X68.488 Y28.094 R33.964
  X71.48 Y23.127 R12.6
G1 Y15.441
G2 X66.9 Y9.1 R12.6
G1 X71.48
  Y15.441
  X73.58
  Y33.864
G3 X73.08 Y34.364 R0.5
G1 X67.626
G3 X67.127 Y33.886 R0.5
G2 X66.237 Y27.682 R31.864
G3 X66.394 Y27.186 R0.5
G2 X60.166 Y8.806 R10.5
  X56.117 Y9.148 R13.622
G3 X55.825 Y7.339 R-0.919
  X63.761 Y7. R93.
G1 X73.08
G3 X73.58 Y7.5 R0.5
G1 Y15.441
G0 Z50.
  X71.48 Y23.127
  Z-3.
G1 Z-8. F80
  Y32.264 F170
  X69.123
G2 X68.488 Y28.094 R33.964
  X71.48 Y23.127 R12.6
G1 Y15.441
G2 X66.9 Y9.1 R12.6
G1 X71.48
  Y15.441
  X73.58
  Y33.864
G3 X73.08 Y34.364 R0.5
G1 X67.626
G3 X67.127 Y33.886 R0.5
G2 X66.237 Y27.682 R31.864
G3 X66.394 Y27.186 R0.5
G2 X60.166 Y8.806 R10.5
  X56.117 Y9.148 R13.622
G3 X55.825 Y7.339 R-0.919
  X63.761 Y7. R93.
G1 X73.08
G3 X73.58 Y7.5 R0.5
G1 Y15.441
G0 Z50.
  X71.48 Y23.127
  Z-6.
G1 Z-11. F80
  Y32.264 F170
  X69.123
G2 X68.488 Y28.094 R33.964
  X71.48 Y23.127 R12.6
G1 Y15.441
G2 X66.9 Y9.1 R12.6
G1 X71.48
  Y15.441
  X73.58
  Y33.864
G3 X73.08 Y34.364 R0.5
G1 X67.626
G3 X67.127 Y33.886 R0.5
G2 X66.237 Y27.682 R31.864
G3 X66.394 Y27.186 R0.5
G2 X60.166 Y8.806 R10.5
  X56.117 Y9.148 R13.622
G3 X55.825 Y7.339 R-0.919
  X63.761 Y7. R93.
G1 X73.08
G3 X73.58 Y7.5 R0.5
G1 Y15.441
  X74.08 F170
  Y33.864
G3 X73.08 Y34.864 R1.
G1 X67.626
G3 X66.627 Y33.908 R1.
G2 X65.752 Y27.801 R31.364
G3 X66.064 Y26.81 R1.
G2 X60.133 Y9.305 R10.
  X56.233 Y9.635 R13.122
G3 X55.783 Y6.841 R-1.419
  X63.761 Y6.5 R93.5
G1 X73.08
G3 X74.08 Y7.5 R1.
G1 Y15.441
G0 Z50.
  X118.755
  Z0.
G1 Z-5. F80
  Y9.1 F170
  X123.334
G2 X118.755 Y15.441 R12.6
G1 Y23.127
G2 X121.747 Y28.094 R12.6
  X121.111 Y32.264 R33.964
G1 X118.755
  Y23.127
  X120.755 Y22.486
G2 X123.841 Y27.186 R10.5
G3 X123.997 Y27.682 R0.5
G2 X123.107 Y33.886 R31.864
G3 X122.608 Y34.364 R0.5
G1 X117.155
G3 X116.655 Y33.864 R0.5
G1 Y7.5
```

```
G3 X117.155 Y7. R0.5
G1 X126.473
G3 X134.409 Y7.339 R93.
  X134.118 Y9.148 R-0.919
G2 X130.068 Y8.806 R13.622
  X120.755 Y22.486 R10.5
G0 Z50.
  X118.755 Y15.441
  Z-3.
G1 Z-8. F80
  Y9.1 F170
  X123.334
G2 X118.755 Y15.441 R12.6
G1 Y23.127
G2 X121.747 Y28.094 R12.6
  X121.111 Y32.264 R33.964
G1 X118.755
  Y23.127
  X120.755 Y22.486
G2 X123.841 Y27.186 R10.5
G3 X123.997 Y27.682 R0.5
G2 X123.107 Y33.886 R31.864
G3 X122.608 Y34.364 R0.5
G1 X117.155
G3 X116.655 Y33.864 R0.5
G1 Y7.5
G3 X117.155 Y7. R0.5
G1 X126.473
G3 X134.409 Y7.339 R93.
  X134.118 Y9.148 R-0.919
G2 X130.068 Y8.806 R13.622
  X120.755 Y22.486 R10.5
G0 Z50.
  X118.755 Y15.441
  Z-6.
G1 Z-11. F80
  Y9.1 F170
  X123.334
G2 X118.755 Y15.441 R12.6
G1 Y23.127
G2 X121.747 Y28.094 R12.6
  X121.111 Y32.264 R33.964
G1 X118.755
  Y23.127
  X120.755 Y22.486
G2 X123.841 Y27.186 R10.5
G3 X123.997 Y27.682 R0.5
G2 X123.107 Y33.886 R31.864
G3 X122.608 Y34.364 R0.5
G1 X117.155
G3 X116.655 Y33.864 R0.5
G1 Y7.5
G3 X117.155 Y7. R0.5
G1 X126.473
G3 X134.409 Y7.339 R93.
  X134.118 Y9.148 R-0.919
G2 X130.068 Y8.806 R13.622
  X120.755 Y22.486 R10.5
G0 Z50.
  X118.755 Y15.441
  Z-9.
G1 Z-12. F80
  Y9.1 F170
  X123.334
G2 X118.755 Y15.441 R12.6
G1 Y23.127
G2 X121.747 Y28.094 R12.6
  X121.111 Y32.264 R33.964
G1 X118.755
  Y23.127
  X120.755 Y22.486
G2 X123.841 Y27.186 R10.5
G3 X123.997 Y27.682 R0.5
G2 X123.107 Y33.886 R31.864
G3 X122.608 Y34.364 R0.5
G1 X117.155
G3 X116.655 Y33.864 R0.5
G1 Y7.5
G3 X117.155 Y7. R0.5
G1 X126.473
G3 X134.409 Y7.339 R93.
  X134.118 Y9.148 R-0.919
G2 X130.068 Y8.806 R13.622
  X120.755 Y22.486 R10.5
G1 X121.231 Y22.334 F170
G2 X124.17 Y26.81 R10.
G3 X124.483 Y27.801 R1.
G2 X123.607 Y33.908 R31.364
G3 X122.608 Y34.864 R1.
G1 X117.155
G3 X116.155 Y33.864 R1.
G1 Y7.5
G3 X117.155 Y6.5 R1.
G1 X126.473
G3 X134.452 Y6.841 R93.5
  X134.002 Y9.635 R-1.419
G2 X130.101 Y9.305 R13.122
  X121.231 Y22.334 R10.
G0 Z50.
/*----------------------*/
/*F-CONTOUR3-T3 - PROFILE*/
/*----------------------*/
  X31.612 Y56.974 Z50.
  Z0.
G1 Z-3. F80
G3 X35.464 Y63.64 R5. F170
G2 X38.569 Y63.451 R-1.7
G3 X41.583 Y56.367 R5.
G0 Z50.
  X31.612 Y56.974
  Z-1.
G1 Z-4. F80
G3 X35.464 Y63.64 R5. F170
G2 X38.569 Y63.451 R-1.7
G3 X41.583 Y56.367 R5.
G0 Z50.
  X31.612 Y56.974
  Z-2.
G1 Z-4.5 F80
G3 X35.464 Y63.64 R5. F170
G2 X38.569 Y63.451 R-1.7
G3 X41.583 Y56.367 R5.
G0 Z50.
  X31.612 Y56.974
  Z0.
G1 Z-4.5 F80
  X31.445 Y57.96 F170
G3 X34.526 Y63.293 R4.
G2 X39.457 Y62.993 R-2.7
G3 X41.869 Y57.325 R4.
G1 X41.583 Y56.367
G0 Z50.
  X55.919 Y27.628
  Z0.
G1 Z-3. F80
G3 X59.766 Y20.959 R5. F170
G2 X58.05 Y18.365 R-1.7
G3 X50.407 Y19.297 R5.
G0 Z50.
  X55.919 Y27.628
  Z-1.
G1 Z-4. F80
G3 X59.766 Y20.959 R5. F170
G2 X58.05 Y18.365 R-1.7
G3 X50.407 Y19.297 R5.
G0 Z50.
  X55.919 Y27.628
  Z-2.
G1 Z-4.5 F80
G3 X59.766 Y20.959 R5. F170
G2 X58.05 Y18.365 R-1.7
G3 X50.407 Y19.297 R5.
G0 Z50.
  X55.919 Y27.628
  Z0.
G1 Z-4.5 F80
  X56.856 Y27.28 F170
G3 X59.934 Y21.945 R4.
G2 X57.208 Y17.825 R-2.7
G3 X51.094 Y18.571 R4.
G1 X50.407 Y19.297
G0 Z50.
  X18.351 Y21.251
  Z0.
G1 Z-3. F80
G3 X10.652 Y21.254 R5. F170
G2 X9.264 Y24.037 R-1.7
G3 X13.892 Y30.19 R5.
G0 Z50.
  X18.351 Y21.251
  Z-1.
G1 Z-4. F80
G3 X10.652 Y21.254 R5. F170
G2 X9.264 Y24.037 R-1.7
G3 X13.892 Y30.19 R5.
G0 Z50.
  X18.351 Y21.251
  Z-2.
G1 Z-4.5 F80
G3 X10.652 Y21.254 R5. F170
G2 X9.264 Y24.037 R-1.7
G3 X13.892 Y30.19 R5.
G0 Z50.
  X18.351 Y21.251
  Z0.
G1 Z-4.5 F80
  X17.581 Y20.613 F170
G3 X11.422 Y20.616 R4.
G2 X9.216 Y25.036 R-2.7
G3 X12.919 Y29.958 R4.
G1 X13.892 Y30.19
G0 Z50.
  X139.827 Y19.297
  Z0.
G1 Z-3. F80
G3 X132.185 Y18.365 R5. F170
G2 X130.469 Y20.959 R-1.7
G3 X134.316 Y27.628 R5.
G0 Z50.
  X139.827 Y19.297
  Z-1.
G1 Z-4. F80
G3 X132.185 Y18.365 R5. F170
G2 X130.469 Y20.959 R-1.7
G3 X134.316 Y27.628 R5.
G0 Z50.
  X139.827 Y19.297
  Z-2.
G1 Z-4.5 F80
G3 X132.185 Y18.365 R5. F170
G2 X130.469 Y20.959 R-1.7
G3 X134.316 Y27.628 R5.
G0 Z50.
  X139.827 Y19.297
  Z0.
G1 Z-4.5 F80
  X139.14 Y18.571 F170
G3 X133.026 Y17.825 R4.
G2 X130.3 Y21.945 R-2.7
G3 X133.378 Y27.28 R4.
G1 X134.316 Y27.628
G0 Z50.
  X176.342 Y30.19
  Z0.
G1 Z-3. F80
```

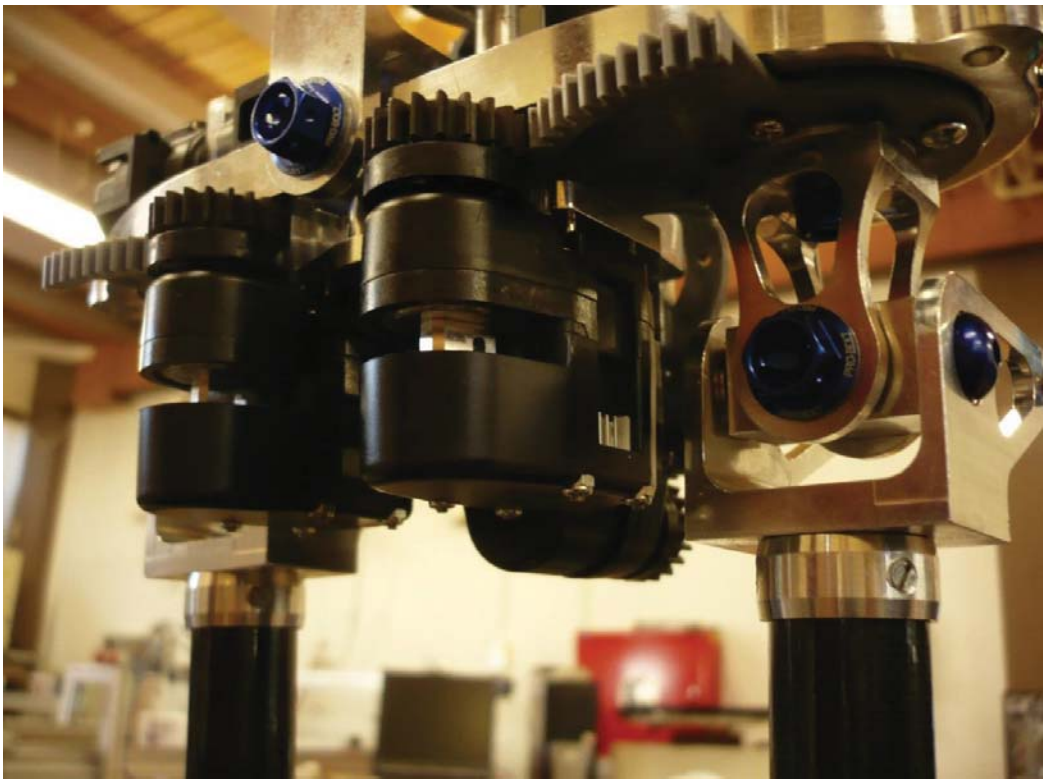
```
G3 X180.971 Y24.037 R5. F170
G2 X179.582 Y21.254 R-1.7
G3 X171.883 Y21.251 R5.
G0 Z50.
  X176.342 Y30.19
  Z-1.
G1 Z-4. F80
G3 X180.971 Y24.037 R5. F170
G2 X179.582 Y21.254 R-1.7
G3 X171.883 Y21.251 R5.
G0 Z50.
  X176.342 Y30.19
  Z-2.
G1 Z-4.5 F80
G3 X180.971 Y24.037 R5. F170
G2 X179.582 Y21.254 R-1.7
G3 X171.883 Y21.251 R5.
G0 Z50.
  X176.342 Y30.19
  Z0.
G1 Z-4.5 F80
  X177.315 Y29.958 F170
G3 X181.018 Y25.036 R4.
G2 X178.813 Y20.616 R-2.7
G3 X172.653 Y20.613 R4.
G1 X171.883 Y21.251
G0 Z50.
  X148.652 Y56.367
  Z0.
G1 Z-3. F80
G3 X151.665 Y63.451 R5. F170
G2 X154.77 Y63.64 R-1.7
G3 X158.622 Y56.974 R5.
G0 Z50.
  X148.652 Y56.367
  Z-1.
G1 Z-4. F80
G3 X151.665 Y63.451 R5. F170
G2 X154.77 Y63.64 R-1.7
G3 X158.622 Y56.974 R5.
G0 Z50.
  X148.652 Y56.367
  Z-2.
G1 Z-4.5 F80
G3 X151.665 Y63.451 R5. F170
G2 X154.77 Y63.64 R-1.7
G3 X158.622 Y56.974 R5.
G0 Z50.
  X148.652 Y56.367
  Z0.
G1 Z-4.5 F80
  X148.366 Y57.325 F170
G3 X150.777 Y62.993 R4.
G2 X155.708 Y63.293 R-2.7
G3 X158.79 Y57.96 R4.
G1 X158.622 Y56.974
G0 Z50.
M9
G91 G28 Z0
G90
M01
N5 M6 T1
T8
/* TOOL -1- ROUGH DIA 4.0 MM */
G90 G00 G40 G54
G43 H1 D1 G0 X37.058 Y64.231 Z70.
S2500 M3
M8
/*-------------------*/
/*D-DRILL-T1 - DRILL*/
/*-------------------*/
  X37.058 Y64.231 Z50.
G98 G81 Z-5.5 R-2.5 F100
   X9.344 Y22.339
   X59.48 Y19.284
   X130.754
   X180.891 Y22.339
   X153.176 Y64.231
G80
M9
G91 G28 Z0
G90
M01
N6 M6 T8
T16
/* TOOL -8- DRILL DIA 4.2 MM */
G90 G00 G40 G54
G43 H8 D8 G0 X37.058 Y64.231 Z70.
S2000 M3
M8
/*-------------------*/
/*D-DRILL1-T8 - DRILL*/
/*-------------------*/
  X37.058 Y64.231 Z50.
G98 G83 Z-13. R0. Q2.5 F80
   X9.344 Y22.339
   X59.48 Y19.284
   X130.754
   X180.891 Y22.339
   X153.176 Y64.231
G80
M9
G91 G28 Z0
G90
M01
N7 M6 T16
T16
/* TOOL -16- MILL DIA 8.0 R0. MM
*/
G90 G00 G40 G54
G43 H16 D16 G0 X35.294 Y37.985
Z70. S3000 M3
M8
/*-----------------------*/
/*P-CONTOUR4-T16 - POCKET*/
/*-----------------------*/
  X35.294 Y37.985 Z50.
  Z0.
G1 Z-6. F100
G3 X35.294 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X35.294 Y40.785 I0. J-5.5
G1 Y43.585
G3 X35.294 Y43.585 I0. J-8.3
G1 Y46.385
G3 X35.294 Y46.385 I0. J-11.1
G1 Y49.185
G3 X35.294 Y49.185 I0. J-13.9
G1 Y51.985
G3 X35.294 Y51.985 I0. J-16.7
G1 Y54.785
G3 X35.294 Y54.785 I0. J-19.5
G0 Z50.
  Y37.985
  Z-4.
G1 Z-10. F100
G3 X35.294 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X35.294 Y40.785 I0. J-5.5
G1 Y43.585
G3 X35.294 Y43.585 I0. J-8.3
G1 Y46.385
G3 X35.294 Y46.385 I0. J-11.1
G1 Y49.185
G3 X35.294 Y49.185 I0. J-13.9
G1 Y51.985
G3 X35.294 Y51.985 I0. J-16.7
G1 Y54.785
G3 X35.294 Y54.785 I0. J-19.5
G0 Z50.
  Y37.985
  Z-8.
G1 Z-13. F100
G3 X35.294 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X35.294 Y40.785 I0. J-5.5
G1 Y43.585
G3 X35.294 Y43.585 I0. J-8.3
G1 Y46.385
G3 X35.294 Y46.385 I0. J-11.1
G1 Y49.185
G3 X35.294 Y49.185 I0. J-13.9
G1 Y51.985
G3 X35.294 Y51.985 I0. J-16.7
G1 Y54.785
G3 X35.294 Y54.785 I0. J-19.5
G0 Z50.
  X154.94 Y37.985
  Z0.
G1 Z-6. F100
G3 X154.94 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X154.94 Y40.785 I0. J-5.5
G1 Y43.585
G3 X154.94 Y43.585 I0. J-8.3
G1 Y46.385
G3 X154.94 Y46.385 I0. J-11.1
G1 Y49.185
G3 X154.94 Y49.185 I0. J-13.9
G1 Y51.985
G3 X154.94 Y51.985 I0. J-16.7
G1 Y54.785
G3 X154.94 Y54.785 I0. J-19.5
G0 Z50.
  Y37.985
  Z-4.
G1 Z-10. F100
G3 X154.94 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X154.94 Y40.785 I0. J-5.5
G1 Y43.585
G3 X154.94 Y43.585 I0. J-8.3
G1 Y46.385
G3 X154.94 Y46.385 I0. J-11.1
G1 Y49.185
G3 X154.94 Y49.185 I0. J-13.9
G1 Y51.985
G3 X154.94 Y51.985 I0. J-16.7
G1 Y54.785
G3 X154.94 Y54.785 I0. J-19.5
G0 Z50.
  Y37.985
  Z-8.
G1 Z-13. F100
G3 X154.94 Y37.985 I0. J-2.7 F450
G1 Y40.785
G3 X154.94 Y40.785 I0. J-5.5
G1 Y43.585
G3 X154.94 Y43.585 I0. J-8.3
G1 Y46.385
G3 X154.94 Y46.385 I0. J-11.1
G1 Y49.185
G3 X154.94 Y49.185 I0. J-13.9
G1 Y51.985
G3 X154.94 Y51.985 I0. J-16.7
G1 Y54.785
G3 X154.94 Y54.785 I0. J-19.5
G0 Z50.
/*-----------------------*/
/*F-CONTOUR5-T16 - PROFILE*/
/*-----------------------*/
  X35.294 Y56.284 Z50.
  Z0.
G1 Z-5. F100
G41 G1 Y60.285 F450
```

```
G3 X35.294 Y60.285 I0. J-25.
G40 G1 Y56.284
  Z-8. F100
G41 G1 Y60.285 F450
G3 X35.294 Y60.285 I0. J-25.
G40 G1 Y56.284
  Z-10.5 F100
G41 G1 Y60.285 F450
G3 X35.294 Y60.285 I0. J-25.
G40 G1 Y56.284
G0 Z50.
  Z0.
G1 Z-10.5 F100
G41 G1 Y61.285 F450
G3 X35.294 Y61.285 I0. J-26.
G40 G1 Y56.284
G0 Z50.
  X154.94
  Z0.
G1 Z-5. F100
G41 G1 Y60.285 F450
G3 X154.94 Y60.285 I0. J-25.
G40 G1 Y56.284
  Z-8. F100
G41 G1 Y60.285 F450
G3 X154.94 Y60.285 I0. J-25.
G40 G1 Y56.284
  Z-10.5 F100
G41 G1 Y60.285 F450
G3 X154.94 Y60.285 I0. J-25.
G40 G1 Y56.284
G0 Z50.
  Z0.
G1 Z-10.5 F100
G41 G1 Y61.285 F450
G3 X154.94 Y61.285 I0. J-26.
G40 G1 Y56.284
G0 Z50.
M9
G91 G28 Y0 Z0
M30
%
```

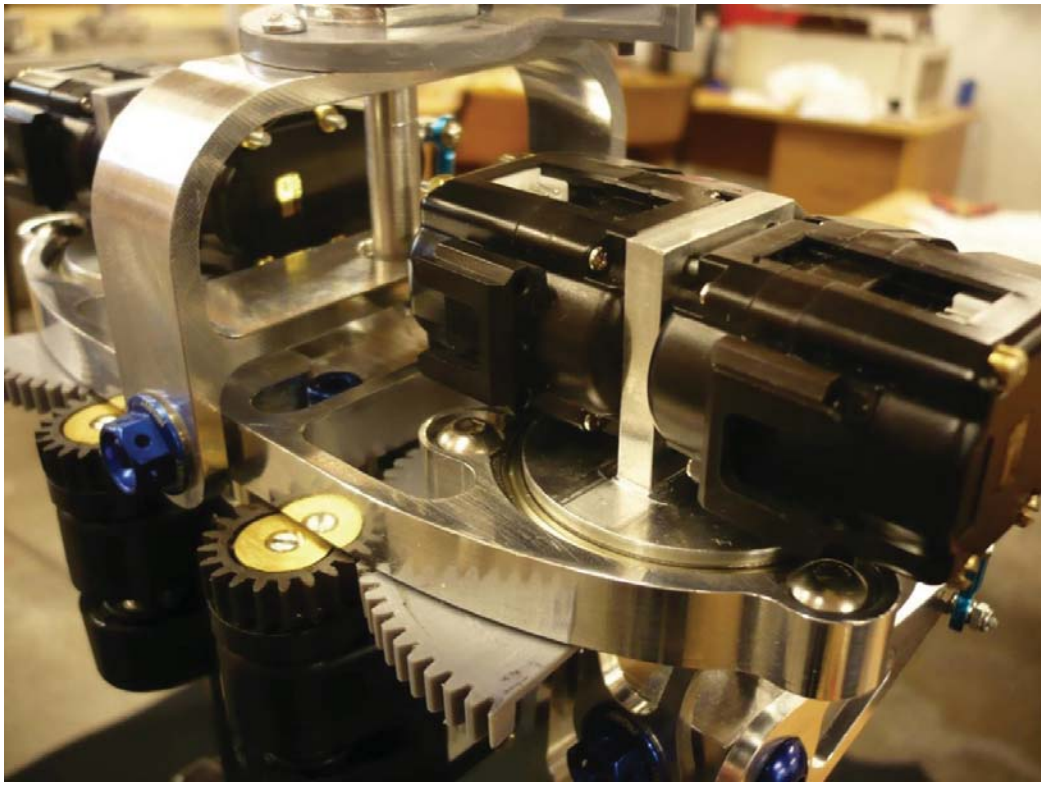
9.3 *Appendix C*

9.3.1 Images of completed humanoid; MURPH.

Completed lower half of the humanoid



Pelvis and hip area of humanoid



Upper hip area showing the yaw axis

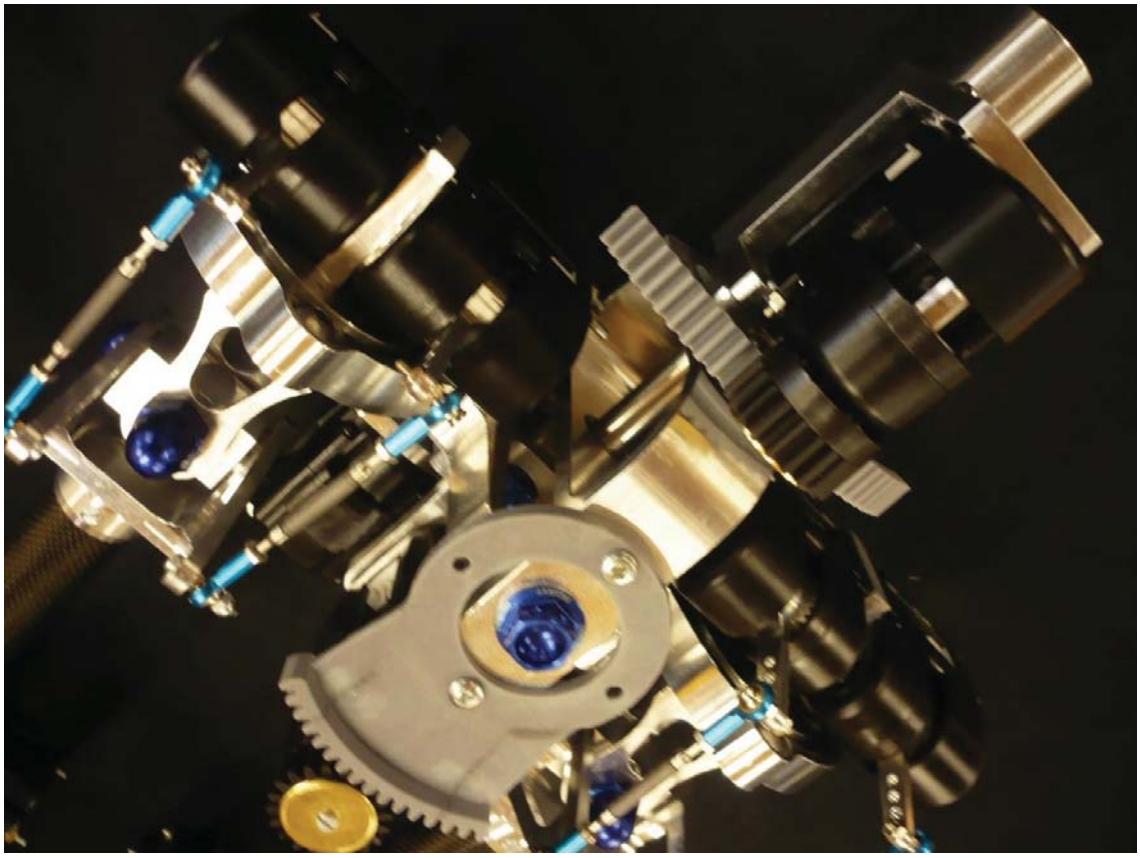
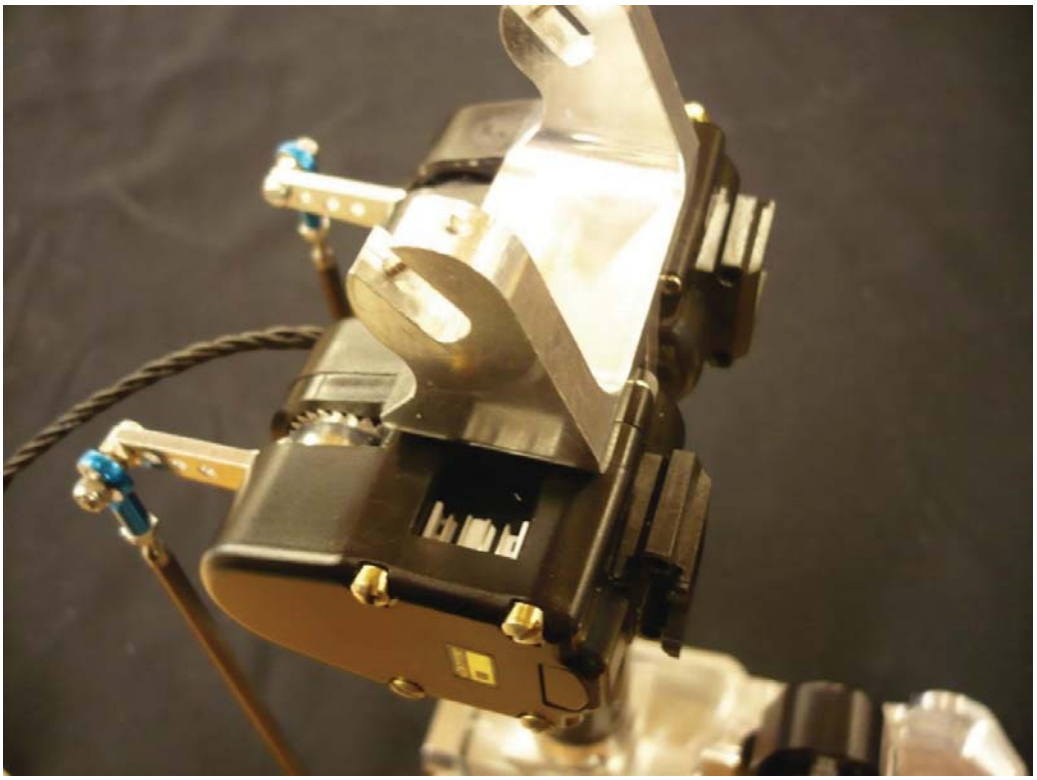


Image from behind the humanoid showing the pelvis pivot and twist/yaw axis

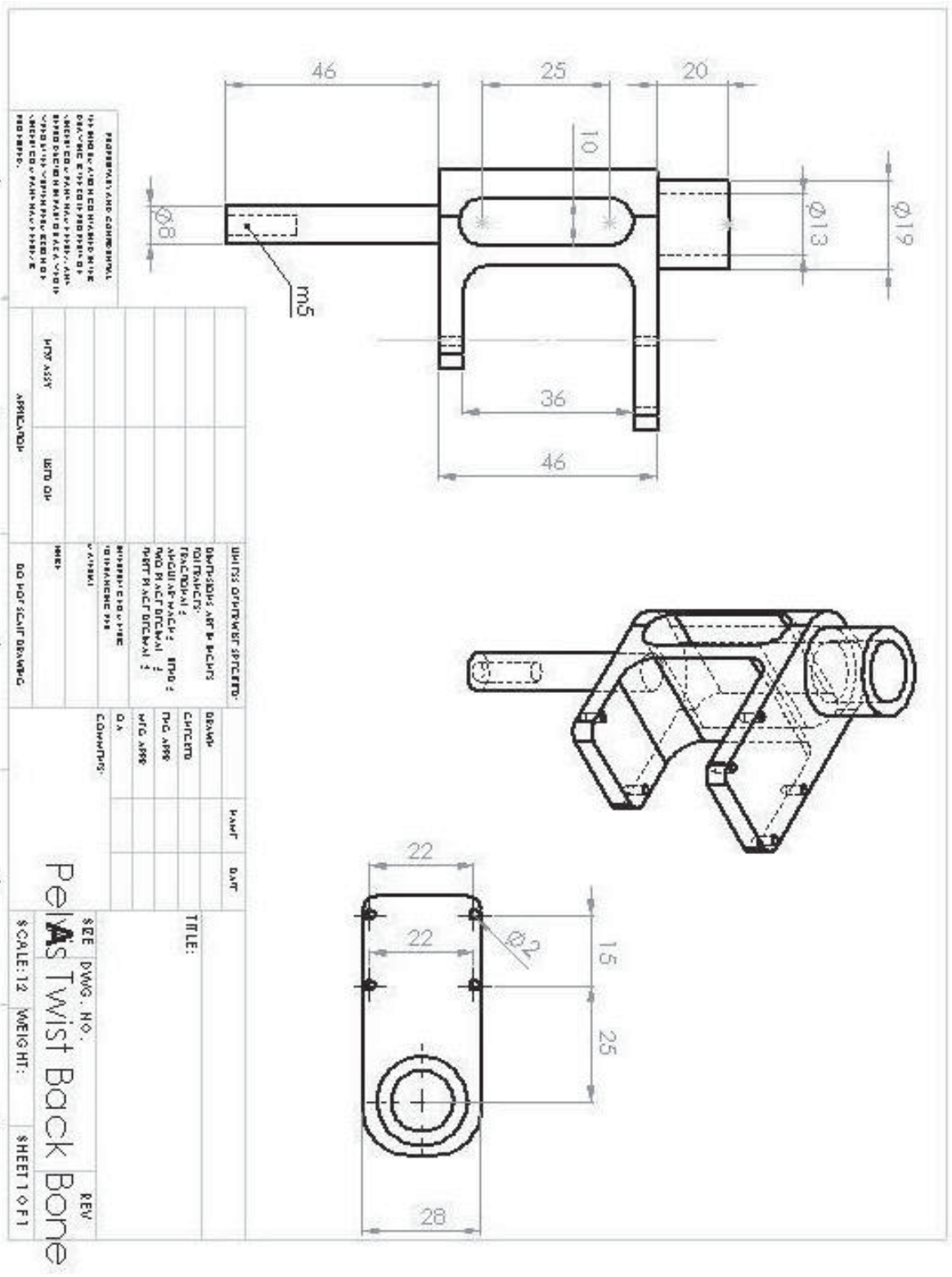


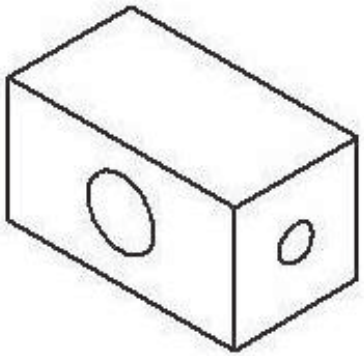
Lower half of the leg displayed from the knee joint. The two ankle actuator modules can be seen

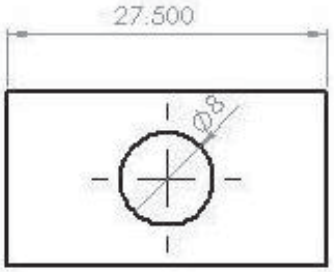
MURPH executing a stepping stance

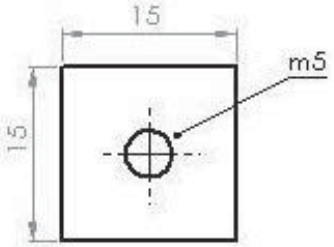
9.4 *Appendix D*

Workshop drawings of components that were machined by hand on a regular milling machine.









		UNLESS OTHERWISE SPECIFIED:		NAME	DATE
		DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ANGULAR: MACH BEND TWO PLACE DECIMAL THREE PLACE DECIMAL	DRAWN		
			CHECKED		
			ENG APPR.		
			MFG APPR.		
		INTERPRET GEOMETRIC TOLERANCING PER:	Q.A.		
		MATERIAL	COMMENTS:		
NEXT ASSY	USED ON	FINISH			
APPLICATION		DO NOT SCALE DRAWING			

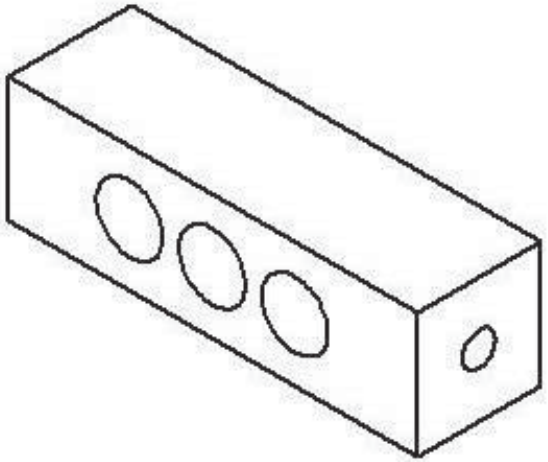
TITLE:

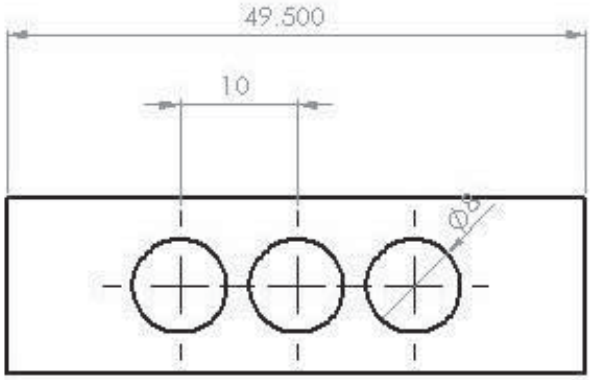
SIZE: A | DWG. NO. | REV

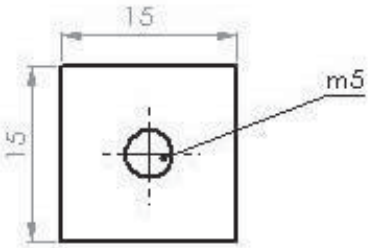
Hip 15x15 joint pivot

SCALE: 2:1 | WEIGHT: | SHEET 1 OF 1

5 4 3 2 1







		UNLESS OTHERWISE SPECIFIED:		NAME	DATE
		DIMENSIONS ARE IN INCHES TOLERANCES: FRACTIONAL ANGULAR: MACH BEND TWO PLACE DECIMAL THREE PLACE DECIMAL	DRAWN		
			CHECKED		
			ENG APPR		
			MFG APPR		
		INTERPRET GEOMETRIC TOLERANCING PER:	Q A		
		MATERIAL	COMMENTS:		
		FINISH			
NEXT ASSY	USED ON				
APPLICATION		DO NOT SCALE DRAWING			

TITLE:

SIZE	DWG. NO.	REV
A	Ankle 15x15 joint pivot	
SCALE: 2:1	WEIGHT:	SHEET 1 OF 1

5 4 3 2 1