DESIGN AND CONTROL OF A SOCCER-PLAYING HUMANOID ROBOT

HO HOAN NGHIA

NATIONAL UNIVERSITY OF SINGAPORE
2004

DESIGN AND CONTROL OF A SOCCER-PLAYING HUMANOID ROBOT

HO HOAN NGHIA
(B.Eng, (Hons.), HCMC University of Technology)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF ENGINEERING
DEPARTMENT OF MECHANICAL ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2004

Acknowledgements

I would like to thank my supervisors, Prof. Hong Geok Soon and Dr. Chew Chee Meng, for all their help and support during the past two years. Despite of their tight schedules, they were willing to discuss with me for hours to solve any problem that was blocking my way. Their helpful advices helped me clarify many issues, and lead me out of wrong direction. I am also very grateful for their kindness and patience whenever I made mistake during research.

I also want to take this opportunity to express my thanks to the lab's technicians, Ms. Tshin, Ms. Ooi, Ms. Hamilda, and Mr. Zhang. Without their help and support, I may not finish my works. Their unconditional help and kindness has surprised me so much. All their paperwork has invaluable contributions to my project.

Thanks to members of legged locomotion group: Sateesh, Wai Yong, Feng Kai, Zhou Wei, Hu Jiay, who contributed valuable discussion and correction that lead to the completion of my thesis. I also acknowledge helpful advices from Eddie Choong during my first days of the project. His explanation about several issues made me familiar with the project faster and with confidence. I also thank Alan and Tay Lian Sen, the FYP students, who made excellent contributions related to vision processing and mechanical design. Thanks to my family and friends who have great support and encouragement to help me finish this thesis.

Table of Contents

Summary

This thesis presents the design and control of a small-sized humanoid robot called ROPE. The name stands for **RO**bot **PE**le, or **RO**bot for **P**ersonal **E**ntertainment. ROPE was designed firstly to participate in international robotics games like RoboCup and FIRA. These are two popular robotics games whose purpose is to promote research and development of robot. Besides, ROPE can also be used as a test-bed for research on bipedal walking control.

The design of ROPE consists of two aspects: mechanical and control system design. The mechanical system shapes the structure and appearance of the robot to that of human, and the control system will decide its human-like behaviors. Mechanical structure of ROPE is mainly made of sheet metal to reduce cost. The control system uses PC-104, a highly compact PC board, as central processor. This PC system processes sensory information and generates control signals accordingly to control the actuators. The sensors include a vision camera on the head, attitude sensors in the body and force sensors on the feet sole. RC servomotors are used as actuators to drive the legs, the arms and the head.

The control algorithm used to control the walking of ROPE is planned in Cartesian space instead of joint space. The planned Cartesian trajectories are converted to joints' trajectories using inverse kinematics transformation. This transformation is carried out online during the walking of the robot. Besides, learning algorithm is also studied in simulation to examine the feasibility of learning to improve the performance of the robot. In this study, genetic algorithm is used to optimize the stance ankle gain such that smooth motion can be obtained. The framework to control the simulated biped is Virtual Model Control.

List of Figures

List of Tables

Chapter 1

Introduction

Humanoid robots have been attracting much attention from public in recent years due to introduction of several impressive commercial robots such as ASIMO, Qrio, HOAP, HRP, and H7. However, it is still far away from the final goal that aims at making humanoid robot a companion of human in factory, outdoor, at home or even outer space. Despite of that fact, research on humanoid robot greatly benefits the development of several technologies. The challenges posed by the development of a humanoid robot will accelerate the invention of more advanced actuators, sensors and will promote the understanding of bipedal locomotion and issues related to artificial intelligence.

To make a common ground for researchers in this field to exchange their ideas, there are two big robotic competitions taking place annually: RoboCup and FIRA. A project was started two years ago to design a biped, called ROPE, to participate in such competitions, and more importantly, to make it a test-bed for research on several issues of humanoid robot. At the time this thesis is finalized, ROPE has just come back from RoboCup 2004 in Lisbon, Portugal. It was ranked fifth among thirteen teams participating in that competition. It was also among a few robots that were fully autonomous, i.e. all the

controllers are installed in the robot and no remote control is necessary. Rules of this competition are included in Appendix E.

Due to broad nature of humanoid robotics research, ROPE project is limited to creating a small-sized humanoid robot, which is capable of walking stably and performing some basic soccer playing skills such as approaching a ball and kicking it toward a goal, defending a goal, avoiding obstacles on its way, etc. The design strategy for this robot is using off-the-shelf components to save cost.

The mechanical design of ROPE was done in SolidWork®. The bipedal walking simulation was done in Yobotics®, a dynamic simulation package. Matlab® was used to analyze simulation and experimental results. The control program was developed on RT-Linux operating system and in C programming language.

This thesis is organized into 7 chapters:

Chapter 1: Introduction. In this chapter, the scope of research is introduced. The organization of this thesis is also described.

Chapter 2: Literature review. Several issues related to current status of humanoid robotics research will be reviewed in this chapter. This survey helps determine research direction for the project.

Chapter 3: Sensors, Actuators and Control System. This chapter presents the selection of three most important elements of humanoid robots: sensors, actuators and controllers. A set of criteria is applied to select each component out of a wide range of commercial products available in the market. This chapter also discusses the signal processing for the sensors and the operation of the control system.

Chapter 4: Mechanical Design. Design philosophy and technical specifications of the robot can be found in the first part of this chapter. The rest of the chapter is the detailed design of the robot, in which legs, body, arms and head design are presented.

Chapter 5: Bipedal Walking Control. This chapter deals with the development of algorithms used to control the robot. The main strategy is to plan the trajectories in Cartesian space using Linear Inverted Model or polynomial curve fitting. Simulation and experimental results are also discussed in this chapter.

Chapter 6: Learning Algorithm. This chapter discusses the feasibility of applying learning algorithm to optimize the walking of the biped. In this study, genetic algorithm is used to optimize the ankle torque of the stance leg. Simulation results will be discussed.

Chapter 7: Conclusion and Future Works. Conclusions will be made in this chapter and possible works are recommended to improve this robot in future.

Chapter 2

Literature Review

2.1. Latest Achievements in Realizing Humanoid Robot

The concept of humanoid robot is not new. Its history is as long as the history of our civilization. Effort of making a machine with the shape of human that can do repetitive and hazardous tasks can be dated back to the era of ancient Greek and Egyptian with their automata. More recently, Leonardo da Vinci, a talented and famous Italian artist and engineer in the Renaissance, has also designed some mechanical men [45]. However, it must be waited until the last three decades to see humanoid robot being studied systematically. Beginning with the project at Waseda University, which resulted in WABOT-1 in 1972 [3], the new era of humanoid robot has started and been moving forward at a faster and faster pace. Impressive commercial humanoid robots introduced consecutively by several Japanese industrial giant companies during the past ten years have affirmed this trend. P2, P3, ASIMO from Honda [5][51]; HOAP from Fujitsu [50]; SDR-4X, Qrio from Sony [6][53]; HRP-2P, H7 from Kawada Industries Inc. [4] are among the most impressive commercial humanoid robots. Their capabilities can surprise anyone who has ever seen them. Besides, there are countless projects all over the world carried out by universities and institutes without any less interesting [49]. Those are the

signals signifying that 21st century will be the century of personal robots who work in domestic scene rather than in industrial factories as we saw in later half of last century.

It seems that making a humanoid robot able to walk stably is not a too difficult task nowadays. Presently, most humanoid robots can walk stably; however, their gaits are different from human gait. The above-mentioned commercial robots can easily perform most types of bipedal locomotion as human: walking forward, backward, turning around, walking upstairs and downstairs, up a slope and down a slope, dancing, standing on one leg, balancing on a moving ground, etc. Qrio of Sony can even run. However, compared with the average walking speed of human, robot's walking speed is slower. ASIMO of Honda, which is considered the most advanced commercial humanoid robot nowadays, can walk at nearly 0.44m/s on average and a maximum of 1.3m/s [51] compared with 1.33m/s and 4.6m/s respectively for humans [41].

Beside the basic walking skills realized in most humanoid robots, several advanced features are implemented. One of which is the capability of turning a corner sharply without stopping as we can see in ASIMO's walking gait. The ability of balancing a container filled with water held in hands while walking is another remarkable behavior of ASIMO. Recently, the capability of standing up from lying position has attracted more and more attention. This behavior has been implemented in HRP-2P [4] and HOAP-2 [50], and it is useful in case the robot falls down.

However, bipedal walking capabilities are just basic features of a robot. The features that make big difference are artificial intelligence capabilities. ASIMO, Qrio, HRP-2P have been equipped with image and speech recognition, speech synthesis, decision making. They are capable of recognizing some simple objects and responding to simple questions posed by the audiences or obeying voice commands from the users. The DB robot of ATR

Human Information Processing Research Laboratories can juggle three balls or play ping-pong using visual information [14]. However, this project only focuses on the behavior of upper part of body.

Even though human beings have made substantial achievements during the past two decades, so much work is still needed until humanoid robots can co-exist with human beings in normal life. Currently, most of the behaviors of the robots are performed in somewhat well-defined environments. In highly unstructured environments, it is too dangerous for the robot to work without guidance from human. Its robustness and decision making capabilities must be improved much further.

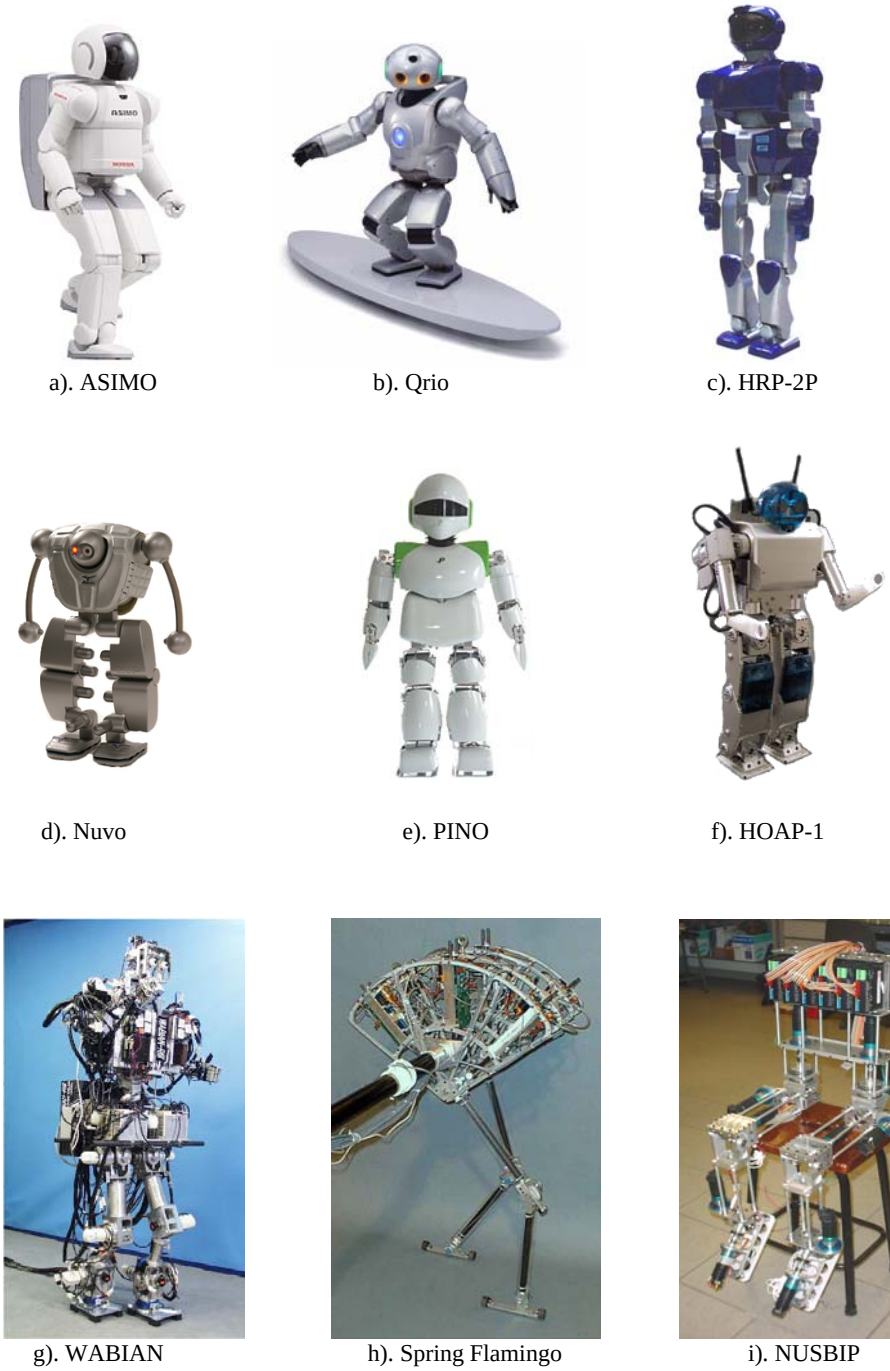
2.2. Hardware System

2.2.1. Mechanical design

Designers of humanoid robots face some difficulties due to the many degrees of freedom and the anthropomorphic requirements. These constraints determine the actuators and material selection as well as mechanical structure design. They pose significant challenges to the designers to overcome the problem of gear backlash, ranges of joints' movement, length ratio between parts of the limbs, etc. Most of the commercial humanoid robots use customized components which are produced only for use in those humanoid robots. The cost is therefore too high to be affordable. For our robot, off-the-shelf components will be more suitable; however, design freedom will be somehow compromised.

Mechanical design of a humanoid robot poses four basic problems to the designers, namely, size of the robot, transmission system, material selection and mechanical structure design. There are several approaches to solve these problems. Strategy of Honda [52] is to

design a humanoid robot with height of normal people while Sony [6] is aiming at small-sized robot. Others' robots have a variety of height in between [3][4][5][8][9].



a). ASIMO b). Qrio c). HRP-2P

d). Nuvo e). PINO f). HOAP-1

g). WABIAN h). Spring Flamingo i). NUSBIP

Fig. 2.1. Some realized humanoid robot. From a) to f) are commercial humanoid robots. The last three pictures are bipeds built for research purpose.

For power transmission, HOAP of Fujitsu uses timing belt transmission [50], Asimo and HRP use harmonic gear transmission [4][11], while most other robots use normal gearbox [6][7][8][9][41]. Gearbox is the most common type of power transmission in humanoid robot because it is a built-in feature of many types of motors. The shortcoming of using gearbox is backlash problem which is more serious with higher gear reduction ratio. Timing belt and harmonic gear transmission are two alternatives to reduce the backlash problem.

Various types of material have been used to make all kinds of components for humanoid robot, for example, aluminum alloy[41], sheet metal [7] and magnesium alloy [4]. Being light, strong and easy to fabricate are a few features that make aluminum alloy the favorite material in many humanoid robots. To make the structure even lighter and stronger, magnesium alloy is used. It was first used in HRP robot. However, this material is difficult to fabricate. Sheet metal is a good alternative for low-cost robot because it can be machined using hand tools; however, this affects accuracy.

A major factor affecting robot performance is the joint design. An effective design may result in a compact structure and a smooth motion. Honda and Sony designed a special type of spherical joint for the hip, and normal pin joints elsewhere [6][11]. METI project designed a cantilever type joint for the hip [4], while others just arranged three pin joints appropriately to form a three-DOF hip joint.

2.2.2. Control system architecture

Figures from 2.2 to 2.5 show typical control systems of humanoid robots for commercial and research purposes. Even though different designers use different components to construct their robots, the control systems of most humanoid robots are quite similar. A typical control system consists of three subsystems: sensory system,

central processor and motion controllers. Selection of those components depends much on how flexible the system is required. Types of actuators in use also affect the designing of the control system.

Figure 2.2 and 2.4 are control systems used to control bipeds driven by RC servomotors. These systems must have several channels of PMW to control the motors. They have a host computer for motion planning and behavior control. Output from host computer will be sent to low-level controller to control motion of the joint appropriately. In PINO, the controller consists of a computer SH2 and a CPLD while in MBR-3 a DSP and FPGA are used. In these systems, the low-level controller is usually able to store all the motion data and playback the motion without the host computer.

Figure 2.3. and 2.5 are control system used to control bipeds driven by DC motors. It must have DC motor drivers to drive the motors. The PID controllers for DC motors can be a motion controller card as in Figure 2.3 or can be done by a real-time program as in Figure 2.5. The systems in Figure 2.3 and 2.5 can actually control the robot by themselves because they are complete computer system. However, they have another option: they can be controlled by a remote computer using wireless connection, for remote control or online operation monitoring.

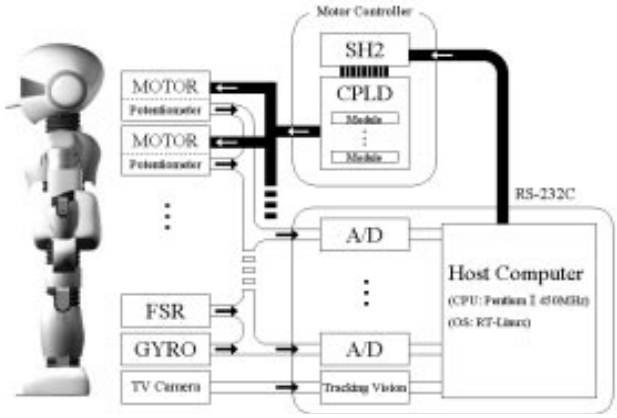


Fig. 2.2. PINO's control system

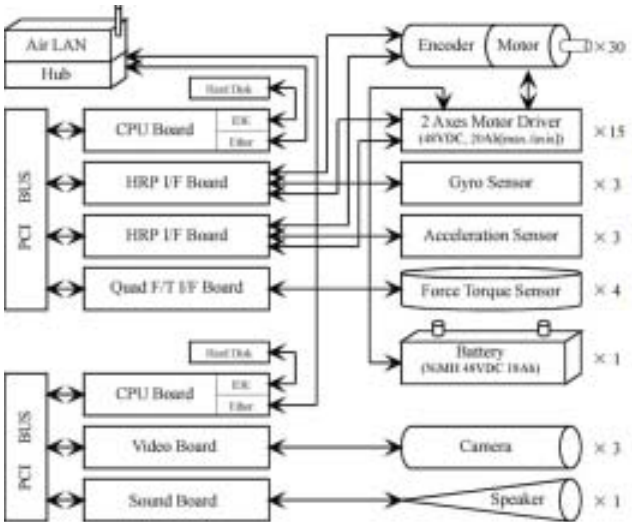


Fig. 2.3. HRP-2P's control system

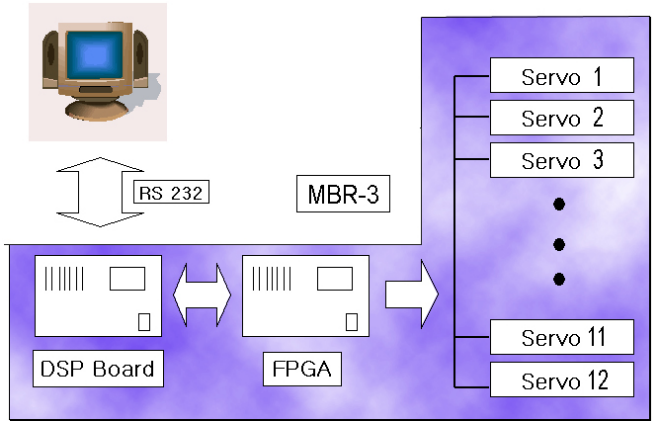


Fig. 2.4. MBR-3's control system

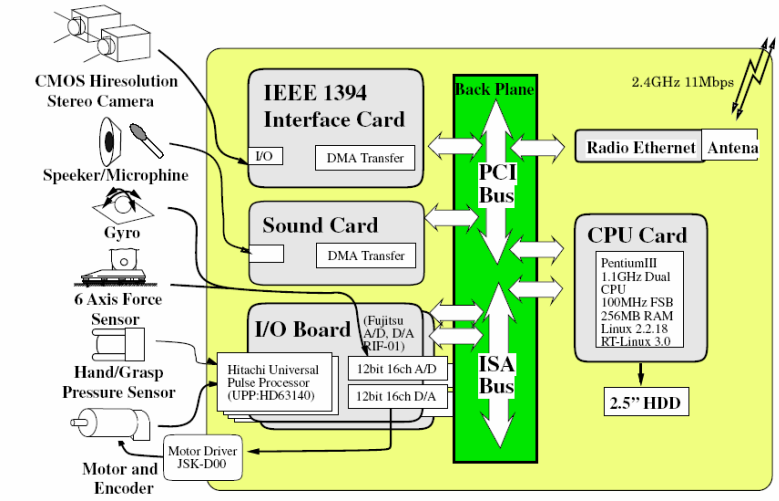


Fig. 2.5. H7's control system

2.3. Bipedal Walking Control

In controlling a humanoid robot, the effect of the under-actuated degree of freedom between the feet sole and the ground become dominant compared with traditional robot (usually referred to as manipulators and which have a fixed base). Due to this problem, a stable walking gait cannot be accomplished simply by planning the joint trajectories as we usually do for a normal manipulator. Generally speaking, strategies to control a biped can be decomposed into four methods: model-based, biologically-inspired, learning and divide-and-conquer.

In model-based methods, a mathematical model is derived based on laws of physics, and a control strategy is developed from this mathematical model. Biological-inspired approaches are trying to replicate the control mechanism of walking in humans. Learning is also an approach inspired by nature. During learning process, the agent will make a try and learn from its experiences until final goals are achieved. Divide-and-conquer is a common approach we usually use to solve complex problem. In this approach, a difficult problem is divided to several sub-problems which are easier to solve. For example, a 3D walking gait can be decomposed into motion on three orthogonal planes; or a walking cycle is divided into several phases.

Those methods are not mutually exclusive but used as composition to form a unified algorithm to control a biped. Followed are a summary of typical algorithms to control humanoid robots.

2.3.1. Virtual Model Control

Virtual Model Control is a control framework in robotics that utilizes virtual components placed at strategic positions to achieve certain effects [17]. The effects of these components are not realized through the use of physical components but by other

means such as the actuation of the joints. This is actually a framework to facilitate the calculation of joints' torques.

The advantage of this control framework lies on its intuitiveness. We can set the parameters of the virtual components intuitively. This may facilitate the tuning process. However, this control framework needs force actuators to realize the control. They are not always available in suitable form for humanoid robots.

2.3.2. Offline kinematics planning

One of the simplest forms of controlling a biped is to plan joint trajectories offline. The control system will control the joints to track these desired trajectories during the playback phase. The desired trajectories can be obtained in a number of ways. The simplest way is to record actual human walking data and perform certain data processing before implementing it to real biped. During the earliest phase of its development of humanoid robot, Honda followed this approach. To ensure a stable gait, a tuning process based on ZMP criterion is done in simulation before any implementation in real robot. ZMP stands for Zero Moment Point, which is a special point on the foot sole used to evaluate the stability of a humanoid robot during walking.

This class of control algorithm is ineffective. The tuning of the trajectories requires great effort and is very tedious. Furthermore, the planned trajectories may yield a stable gait in simulation but they may not work on real robots. This is because the modeling usually does not capture the real dynamics of the robot. In addition, this method will not bring us much understanding about the control of bipedal walking even if we can make the robot walk stably.

2.3.3. Linear inverted pendulum

Kajita et al [20] has found that if an inverted pendulum is constrained to move along a straight line, its motion is governed by a linear differential equation. This equation has a closed-form solution. This property can be utilized to control a biped.

Using this method, the biped is modeled as an inverted pendulum which rotates about the ankle joint, and its mass is lumped at the center of gravity. The joints of the biped are controlled such that its center of mass moves along a constraint line. The motion of the center of mass should follow the motion equation of the linear inverted pendulum. Ankle joint torque will compensate for any error in tracking of the model if it exists.

The advantage of this method is its simple. However, using this algorithm, the ground profile must be well defined because the constraint line is defined with respect to it. Another disadvantage is that the swing leg is ignored in the model. The dynamic effect of the swing leg is considered as disturbance, and will be compensated by the ankle torque. This is sometimes not effective because the ankle torque is limited due to under-actuation problem, and the dynamics of the swing leg changes when walking speed changes.

2.3.4. Natural dynamic exploitation

This algorithm directly exploits the natural dynamics of the controlled biped. The swing leg is allowed to swing freely. Its motion will be limited by the knee cap. The stance ankle uses a compliant mechanism to transfer the center of pressure smoothly along the stance foot during the walking cycle. This algorithm has been proven to work but requires extensive trial-and-error tuning before the biped can walk [18].

2.3.5. ZMP-trajectory planning

The term ZMP, zero moment point, is one of the most well-known terminologies in humanoid robotics. This concept has been widely adopted among researchers, although it is highly controversial.

ZMP was first introduced by Vukobratovic. This term refers to a point on the ground where the total ground reaction force acts upon the foot sole. Therefore the total moment acted about this point is zero, thus the name zero moment point. Based on this concept, a class of walking gait planning has been introduced [21][22][25]. Because it is believed that ZMP reflects the stability of the biped, the ZMP should be planned beforehand. From the desired trajectory of the ZMP, trajectory of the CoM can be derived and therefore trajectories of the joints can be obtained. During the walking period, the controller just needs to make the joints track the reference trajectories. The better the joints track the reference trajectories, the more stable the biped walks.

2.3.6. Central pattern generator (CPG)

It is observed in neurologically simple animals that the rhythmic movement of animal is controlled by rhythm-generating networks in the nervous system [27]. In an attempt to imitate such systems in controlling bipedal robot, several researches have been done [26][27][28]. In these researches, CPGs, which are created using coupled Van der Pol non-linear oscillator equations, will generate periodic signals. These signals are used as references to control joints' motions, which are able to constitute stable walking gaits if they are appropriately generated.

CPGs approach has been implemented successfully in quadruped robot. The resultant control system is simple and robust. However, for biped system, CPG approach becomes extremely hard to implement to real robot due to large number of unknown parameters in

Van der Pol set of oscillator equations. Thus most of research on applying CPG to control bipedal robot is so far limited to 4-dof biped.

Chapter 3

Sensors, Actuators and Control System

3.1. Sensors

Bipedal walking is the most advanced and developed type of locomotion. This erect walking style gives us tremendous advantages by freeing our hands from walking task and engaging them to manipulation. Walking on two feet may be a major reason why human-beings are more evolved than any other species on this planet. However, bipedal walking requires an extremely sophisticated sensory system to balance the body during walking. To replicate bipedal locomotion artificially, a wise way may be to learn from the biological system of human.

Human relies on three sub-systems to balance themselves during movement: vision, proprioception and vestibular. Vision system collects images of surrounding objects to help the brain compute relationship of our body to the environment. Proprioception uses sensory receptors distributed all over the body, especially in muscles and joints, to sense the stretch and pressure of the tissues surrounding them. The vestibular system, located in

the inner ear, consists of three semicircular canals connected to the otolith organs (Figure 3.1). Vestibular system is particularly important for sensing motion of our body.

Among the three systems, vestibular is the most important organ for balance feeling. For example, even though vestibular-disordered people can keep balance through learning and training, they may still feel dizziness and disorientation. The vestibular system consists of three semicircular canals and the otolith organs which includes the saccule and utricle. It is located in the inner ear. The three semicircular canals are arranged perpendicular to each other along the three orthogonal planes to sense angular velocities of the roll, pitch and yaw motions. The otolith organs sense linear acceleration along the three orthogonal axes. Figure 3.1. illustrates the location and structure of vestibular system.

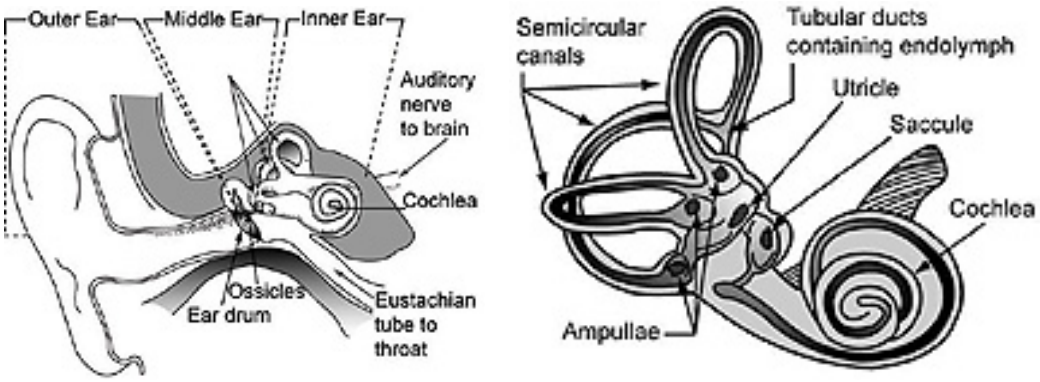


Fig. 3.1. a). Diagram of the ear. b). Vestibular System. *(Adapted from [55])*

The brain will integrate all these sensory signals (vision, proprioception, vestibular) and generate motor control commands accordingly to keep our body in balance. A missing or a disturbing of any of these signals may instantly cause problem in balancing. For example, people can experience a sense of unbalance when they close their eyes and stand on one leg, or inability to walk by 15 months in infants is an indication of hearing disorder, especially the vestibular system. Another example is the feel of dizziness after we spin our body for a while. This is because the fluid in semicircular canals keeps

moving even after our body is stopped due to its momentum. The sensing hair cells in those organs sense the motion of the fluid, and the brain will misinterpret it as a motion of body. That disturbed signal makes us feel like we are spinning, but in fact we have stopped.

Having identified several sensory organs that assist us in keeping balance, we can reproduce a similar sensory system to aid the walking of a humanoid robot. That system consists of a vision sensor on the head, an attitude and motion sensor in the body and force sensors on the feet sole. Vision sensor is actually a camera used to collect surrounding images. Attitude and motion sensors play the same role as that of vestibular system in humans. It consists of three rate gyroscopes and 3-axis accelerometer. Force sensors' function is to sense the pressure on the feet sole.

3.1.1. Force sensor

Force sensors are used to sense the center of pressure on the feet sole. Four force sensors are placed at four corners of each foot from which the center of pressure will be derived. There are a few types of force sensors available commercially, among which are resistive force sensors and load-cells. Resistive force sensors make a change in resistance once a load is applied. Load-cells measure applied force by measuring the deformation of the sensory element. The ranges of measurement of load-cells are usually large; therefore they are not suitable for applications with small range of measurement.

There is a suitable type of resistive force sensor available, a picture of which is shown in Figure 3.2. This sensor has a small active round area (10mm in diameter), which is suitable for small feet, and a flexible cover suitable for mounting. Its resistance is almost infinite under no-load condition and will decrease linearly with applied load. An amplifier circuit is required to convert a change in resistance to a change in output voltage so that

DAQ system can detect the force inputs. A simple inverting amplifier based on 741 op-amp can accomplish this task.

Fig. 3.2. FlexiForce force sensor.

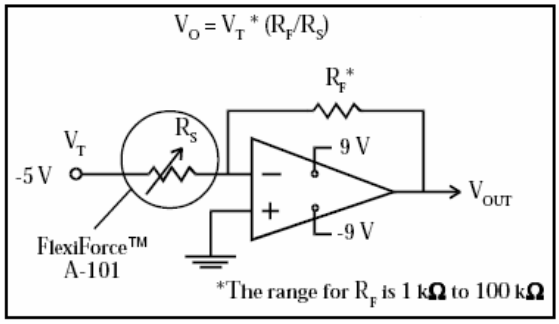


Fig. 3.3. A simple inverting amplifier to make a force-to-voltage conversion.

A calibration process is needed to derive input-output relationship. An apparatus has been set up for this calibration process. The apparatus consists of a power supply, a voltmeter, a weight balance, a device used to concentrate applied forces to the active area of the sensor, and some weights. The experiment setup is shown is Figure 3.4.

Calibration data for the case R_F =62K is shown in Figure 3.5. In this experiment, the signal is relatively noisy and few data have been collected. As a result, the interpolated line does not go through the origin as expected. Another possible reason is that the round tip that concentrate the applied force does not have good contact with the sensor. Another experiment has been done with R_F=40K to reduce the noise level; however, it also reduces the sensitivity of the amplifier. More data has been collected, and the readings were done with an oscilloscope for more reliable results. The experimental data of Figure 3.6 shows the improved result.

Fig. 3.4. The experiment setup for the force calibration.

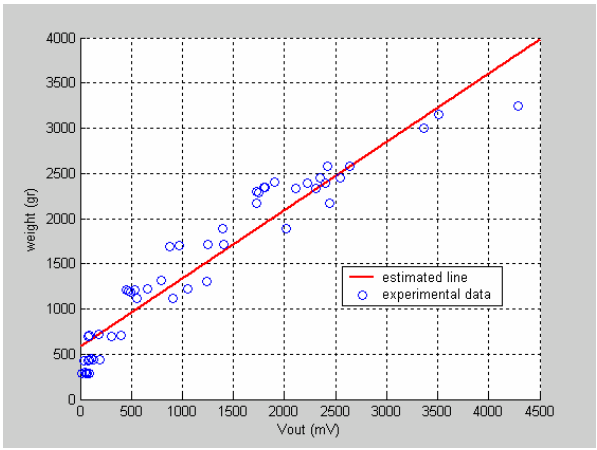


Fig. 3.5. Calibration data for FlexiForce when R_F=62K. The estimated least-square fit line is y = 0.7555x + 578.8518 +/- 283.9 (gr).

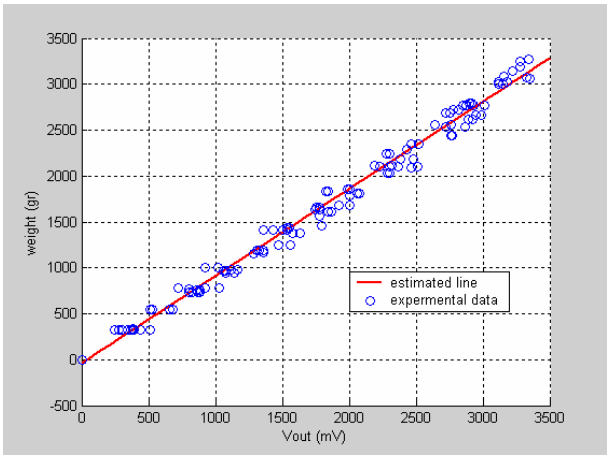


Fig. 3.6. Calibration data for FlexiForce when R_F=40K. The estimated least-square fit line is y = 0.9491x – 40.1495 +/- 93.0815 (gr)

To derive center of pressure from force readings at four corners, a simple static moment equilibrium formula was utilized. The center of pressure is defined as a point on the foot sole where total moment caused by the four forces at the four corners is zero. The coordinates of the center of pressure (CoP) are calculated as followed:

$$x = \frac{(F_3 + F_4)a}{F}$$
$$y = \frac{(F_1 + F_4)b}{F} \tag{3.1}$$

F is the total force; a and b are distances between the sensors along x and y axis respectively. The coordinates of the foot and the applied forces are shown in Figure 3.7.

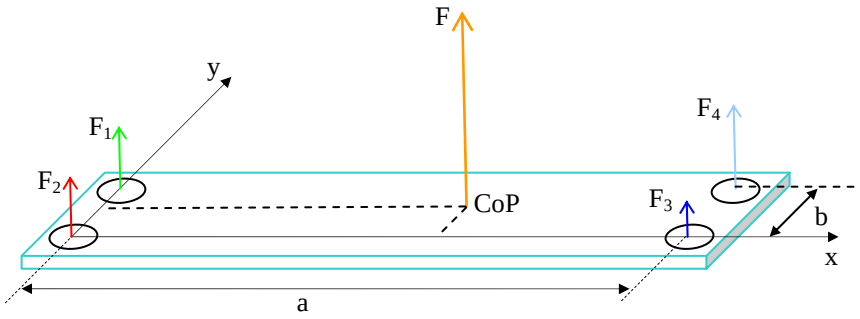


Fig. 3.7. Forces diagram.

3.1.2. Vision sensor

Vision is the dominant sense of human being. In this humanoid robot, vision sensor plays a more modest role due to some limitations. First, an efficient vision system requires intensive computational power which may exceed the capability of a mobile system. Second, the objective of this project is to build a small size humanoid robot, thus size does matter. There is not so many types of camera with suitable size available.

There exists principally two types of camera, namely, CCD (Charge-Coupled Device) and CMOS (Complementary Metal-Oxide Semiconductor) camera. Both of them operate on the same principle: convert light to electrons; however, the technologies used to produce them are totally different and their properties are distinct. CMOS camera is less sensitive to light, noisier, and has a lower resolution than CCD camera. However, CMOS camera is more suitable for this particular application because of its low cost and power consumption.

CMUCam is one of popular CMOS cameras. Its compactness and powerful built-in image processing functions have made it highly competitive. A picture of its is shown in Figure 3.8 Some of its important features are listed in Table 3.1.

Table 3.1. Properties of CMUCam

Speed	17 frames per second
Resolution	80 x 143
Communication	RS-232, maximum 115,2000 bps
Built-in image processing functions	- Extract centroid of a color blob - Gather mean color and variance data - Dump a frame of image - Adjusts image properties

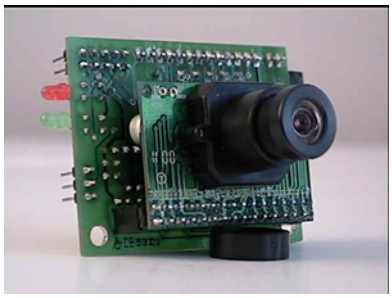


Fig. 3.8. CMUCam

3.1.3. Accelerometer

Beside direct readings of acceleration, accelerometers indirectly provide information of velocity and position from single and double integration of its outputs. A complete set of

these three values, i.e. acceleration, velocity and position, gives the controller important information about the motion and location of the robot's body.

There are several types of accelerometer available in the market: servo electro-mechanical, resonating beam, micro-machined capacitive beam, piezoelectric, piezoresistive. There are also a wide range of brands: Honeywell, Analog Devices, Motorola, Crossbow, STMicroelectronics … Choice of accelerometers depends primarily on performance, cost and size. Among the stated brands, accelerometers of Crossbow and Analog Devices are the most appealing due to their suitable g-range and cost. Therefore, further consideration will be based on products of these two companies. Furthermore, we found from simulation that accelerations of the robot during normal walking rarely exceed 2g. This fact helps to narrows down the options. Table 3.2. shows a comparison of three types of accelerometer from Crossbow and Analog Devices.

Table 3.2. A comparison of accelerometers from Crossbow, Inc. and Analog Devices, Inc.

	Crossbow CXL04LP3	**Analog Devices, Inc. ADXL202**	**Analog Devices, Inc. ADXL210**
Number of axis	3	2	2
Technology	Micro-machined capacitive type	Micro-machined capacitive type	Micro-machined capacitive type
Bandwidth	DC-100Hz	DC-5kHz	DC-5kHz
Range	+/-4g	+/-2g	+/-10g
Supply voltage	5V	3-5.25V	3-5.25V
Output signal	Analog voltage	Analog voltage, Duty cycle	Analog voltage, Duty cycle
Non-linearity	0.2%FS	0.2%FS	0.2%FS
Sensitivity	500 mV/g	312mV/g	100mV/g
Shock	2000g	1000g	1000g

The accelerometer CXL04LP3 from Crossbow has been selected based on several criteria. One of the biggest conveniences of using CXL04LP3 is that it can sense three axes while Analog Devices' sensors have only two axes. Using Analog Devices' sensors,

it is required to arrange two pieces of dual-axis sensor orthogonally to form a tri-axial accelerometer. Without special calibration devices, such arrangement is error-prone. The second factor that affects the selection is the range of accelerations. A range of 2g will marginally satisfy the requirement based on simulation results. Therefore, Crossbow sensor's range of 4g is more reasonable than Analog Devices'. ADXL202 has too small safety margin while ADXL210 has too large margin. Besides, the sensitivity and shock acceptance of Crossbow's product is also better than the other two sensors. The only factor for which Analog Devices' sensor is more advantageous is the bandwidth. However, with a bandwidth of 100Hz, Crossbow's sensor is far more than enough because an average walking has a frequency of 5Hz, equivalent to 2 steps per second. Figure 3.9 shows an image of Crossbow's accelerometer CXL04LP3.

Fig. 3.9. Crossbow's accelerometer.

3.1.4. Rate gyros

Output of accelerometer and its integrations present location and linear motion of the sensor with respect to an earth fixed coordinate frame E when the inertial frame I of the accelerometer is always kept parallel with frame E. When frame I undergoes rotational movement, its orientation vector is needed to derive the sensor's location and motion with respect to frame E.

The orientation of a rigid body can be obtained in a number of ways. The easiest way is using compass and inclinometer. One popular sensor of this kind is the TCM2 of PNI

Corporation. The heading angle is achieved by measuring the earth magnetic field. The tilt angles are measured with respect to the surface of an electrolytic fluid. Using TCM2, the tilt angle is hugely affected by acceleration, and heading angle is affected by magnetic field surrounding the sensor, which is usually the case for a humanoid robot.

Another method to measure the orientation is to use a 3-axis accelerometer. This can be done because the accelerometers can measure the static gravity. By taking the reading of gravity along the axes, tilt angle can be derived. Once again, this reading can only be used in static case. When the robot moves, the readings reflect the dynamic accelerations together with gravitational acceleration.

The third way is to use the rate gyros. Output of the rate gyros is proportional to angular velocity. Its single integration will give rotational angle. A combination of three gyros arranged orthogonally can be used to derive the orientation of the body. Due to the recent development in MEMS technology, the reading of angular velocity can be done in a highly vibrated environment. The selected sensor is CRS03-02 of Silicon Sensing System. This sensor has been selected because of its small size and high resistance to external acceleration of up to 4g at 4kHz.

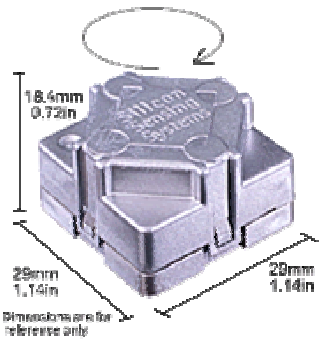


Fig. 3.10. Rate gyro.

3.1.5. Signal processing

Outputs from sensors always contain noise (usually white noise). This factor does not cause much problem when the signal-to-noise ratio is large, and the signals are used directly without being manipulated. However, when the signals are integrated, noise is one of the main factors causing drifting problem. To minimize the effect of noise, the signals need to be processed. This section presents the noise filtering of gyro's and accelerometer's signals as an example of sensory signal processing.

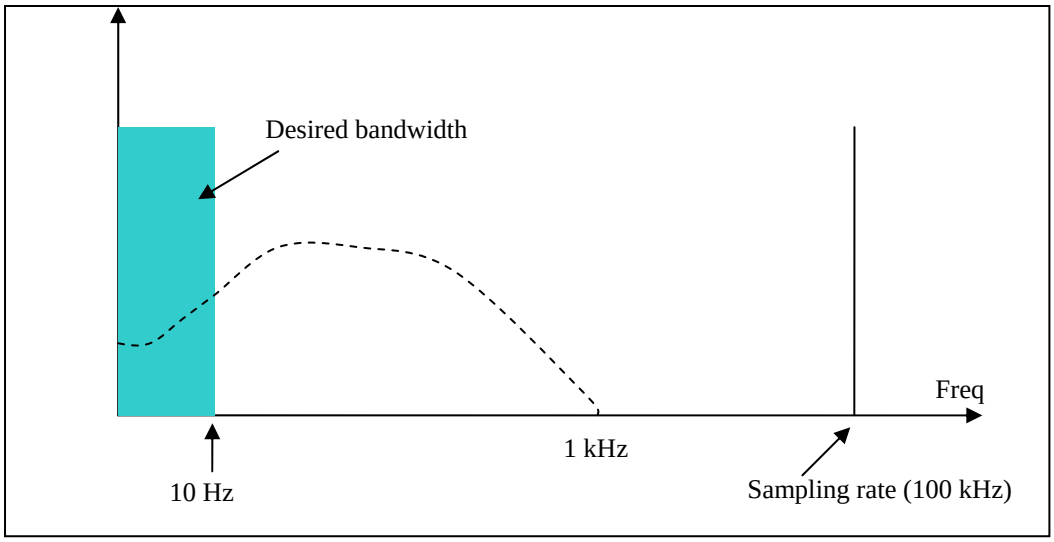


Fig. 3.11. Noise spectrum in the output of gyro. (*Adapted from manufacturer's datasheet)*

Figure 3.11 shows spectrum of the gyro's noise. The noise remains high up to around 1 kHz. If the sampling rate is less than 1 kHz and the signal is fed directly to the DAQ, more noise will be observed in the desired bandwidth. In this case, low-pass filter should be used to reduce the alias noise. However, we can see that the sampling rate of our DAQ system is high enough to avoid aliasing. Thus, low-pass anti-aliasing filter is not necessary. After the AD conversion, a moving average digital filter is used to filter out the noise. The difference equation of moving average filter is relatively simple:

$$y[i] = \frac{1}{M}\sum_{j=0}^{M-1} x[i-j] \tag{3.2}$$

y[i] is the output signal of the filter, x[i] is the input and M is the number of points used in the moving average. This filter is simple but effective in reducing noise. The noise reduction is square-root of the number of point in the average. Figure 3.12 to Figure 3.15 show the noise filtering of accelerometer's and gyro's signals under rest and motion condition.

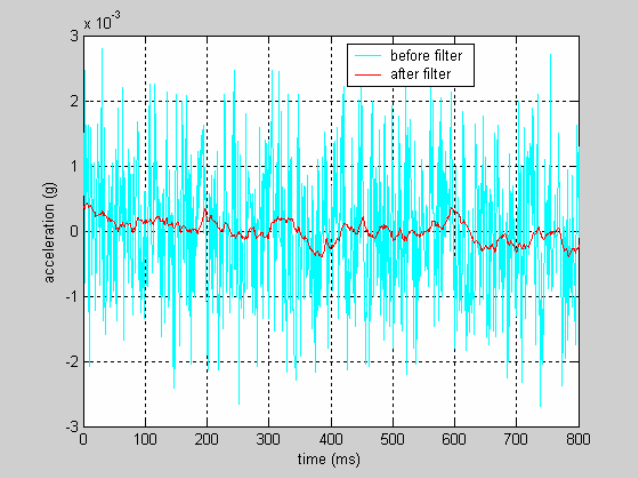


Fig. 3.12. The filtering of output signal of accelerometer when it stands still.

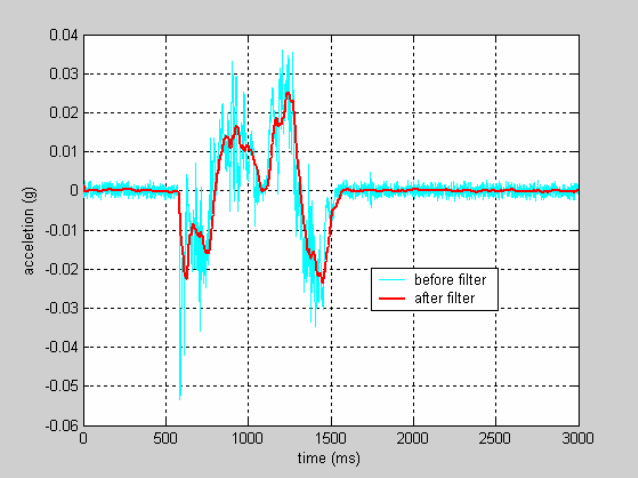


Fig. 3.13. The filtering of output signal of accelerometer when it moves.

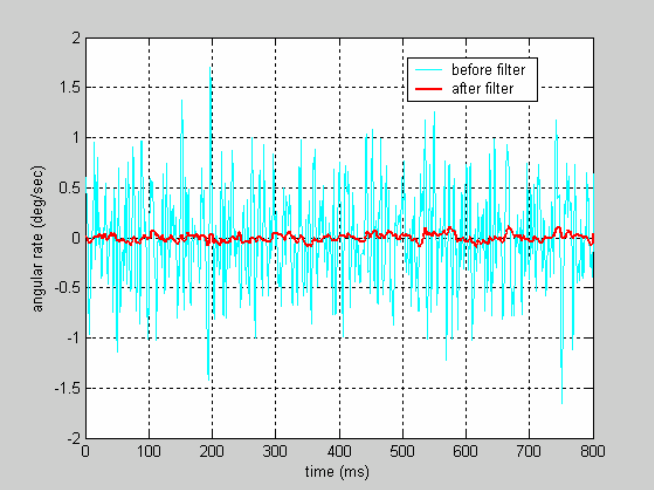


Fig. 3.14. The filtering of output signal of rate gyro when it stands still.

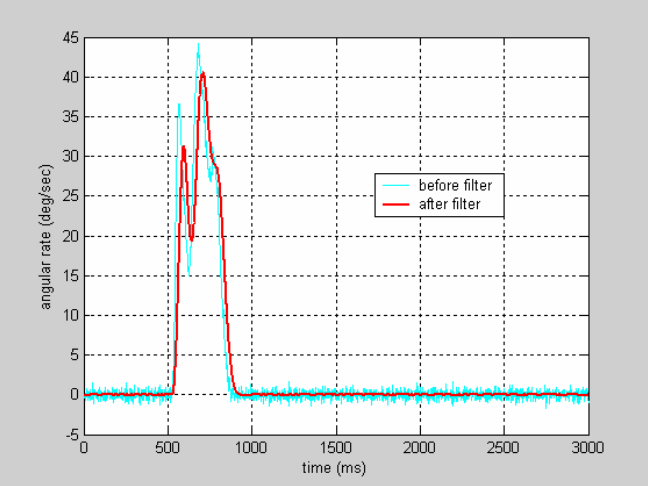


Fig. 3.15. The filtering of output signal of rate gyro when it moves.

3.1.6. Coordinate systems and transformations

To make use of a set of data provided by the attitude and motion sensors (accelerometer and gyros), coordinate systems need to be defined and related to each other. As in any navigational system, three frames are required: a fixed reference frame F attached to the ground, a floating frame B rigidly attached to the moving body, and a non-rotating frame N parallel to F and whose origin attached to the origin of B. With these three frames, the position, orientation and motion of the robot are defined completely.

These informations need to be expressed in reference frame F, while the readings of the sensors are given in local frame B. Transformations are required to relate the three frames.

To relate orientation and position of B with respect to frame F, a homogeneous transformation matrix will be derived. A general movement of B is divided into two fundamental motions: rotation of B with respect to N and translation of N with respect to F. The three rate gyros will give information about rotational motion and 3-axis accelerometer will give information about linear motion.

The absolute rotation angles cannot be used to derive rotation matrix relating frame B to frame N because the sequence of rotation does affect the overall result. This can be checked easily by consecutively rotating a frame about x and y axis by 90^0. If it is done again in reverse order, the final orientation is completely different even though rotation angles given by the gyros are exactly the same. To overcome this problem, the rotation matrix must be updated regularly after short period of time. A discrete-time updating equation can be written as follows:

$$ {}^{N}_{B(m)}R = {}^{N}_{B(m-1)}R.\,{}^{B(m-1)}_{B(m)}R \tag{3.3} $$

where ${}^{N}_{B(m)}R$ is the rotation matrix that relates frame B to frame N at sampled time t_m; ${}^{B(m-1)}_{B(m)}R$ is the rotation matrix that relates frame B at sampled time t_m to frame B at previous sampled time t_{m-1}.

The key point is to find the rotation matrix ${}^{B(m-1)}_{B(m)}R$. If the sampling time Δt is small enough, the angular velocity in one cycle time can be considered to be constant. The rotation axis k expressed in frame B(m) within that cycle time is:

$$k = [k_x\ k_y\ k_z] = \frac{1}{\sqrt{(\omega_x^2 + \omega_y^2 + \omega_z^2)}}\begin{bmatrix}\omega_x & \omega_y & \omega_z\end{bmatrix}^T \quad (3.4)$$

ω_x, ω_y and ω_z are angular velocities read from three gyroscopes at time t_m. The rotation angle in one cycle time about axis k can be approximated as:

$$\theta = \sqrt{(\omega_x \Delta t)^2 + (\omega_y \Delta t)^2 + (\omega_z \Delta t)^2} \quad (3.5)$$

Rotation matrix that relates frame B(m) to frame B(m-1) can be written from vector k and angle θ as follows [47]:

$$^{B(m-1)}_{B(m)}R = rot(k,\theta) = \begin{bmatrix} k_x^2 vers\theta + \cos\theta & k_y k_x vers\theta - k_z \sin\theta & k_z k_x vers\theta + k_y \sin\theta \\ k_x k_y vers\theta + k_z \sin\theta & k_y^2 vers\theta + \cos\theta & k_z k_y vers\theta - k_x \sin\theta \\ k_z k_x vers\theta - k_y \sin\theta & k_z k_y vers\theta + k_x \sin\theta & k_z^2 vers\theta + \cos\theta \end{bmatrix}$$

(3.6)

where $vers\theta = 1 - \cos\theta$.

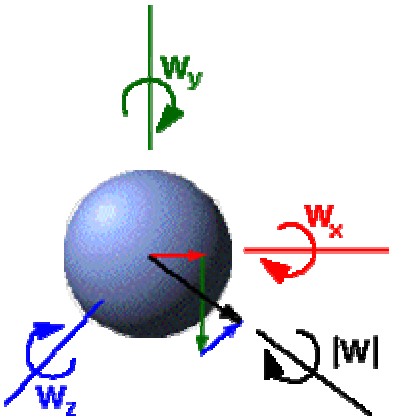


Fig. 3.16. Rotation axis.

Assuming that initially rotation matrix $^{N}_{B(0)}R$ is an identity matrix, which means frame B is in parallel with frame N when the robot begins to move, rotation matrix $^{N}_{B}R$ can be

updated using Eq.3.3. Here it should be also noted that ${}^{N}_{B}R = {}^{F}_{B}R$ because frame N and frame F are parallel.

Given rotation matrix ${}^{N}_{B}R$ and acceleration a_B read from the 3-axis accelerometer and expressed in local frame B, the acceleration vector a_F expressed in frame F can be derived easily. First, acceleration vector a_B in frame B, which is given by the 3-axis accelerometer, will be transformed to frame F using the above-derived rotation matrix: $a_F = {}^{F}_{B}R.a_B$. This acceleration vector is not only caused by linear motion of the robot but also by the static gravity. In order to extract the components due to actual motion of the robot only, the gravity component G_F must be subtracted from a_F:

$$a_F = {}^{F}_{B}R.a_B - G_F \tag{3.7}$$

This acceleration vector will be integrated once to get linear velocities and twice for translational distances along three axes of F. The three translational components will form translation vector ${}^{F}_{B}P = \begin{bmatrix} P_x & P_y & P_z \end{bmatrix}$ which is used together with rotation matrix ${}^{F}_{B}R$ to form the homogeneous transformation matrix relating position and orientation of frame B to frame F as follows:

$${}^{F}_{B}T = \begin{bmatrix} {}^{F}_{B}R & {}^{F}_{B}P \\ 0 & 1 \end{bmatrix} \tag{3.8}$$

3.2. Actuators

Selecting actuators is a tradeoff between size and power. It is desired that the actuators is as powerful as possible; however, more powerful actuators usually means heavier and bulkier ones, and this will add more load to the robot. The right balance between size and power of the actuators can be solved if the actuators can be custom-made just for use in

this robot. Many commercial humanoid robots have been created this way such as ASIMO from Honda, Qrio from Sony. However, it is not suitable for this project because huge resources will be needed. Off-the-shelf actuators will be selected instead.

There are a few options of actuator: DC motors, stepper motors, RC servomotors, force actuators, artificial muscles and pneumatic/hydraulic actuators. For a small robot with less than 50 cm height, pneumatic/hydraulic actuators are out of choice because of its bulky size. Force actuators and artificial muscles are not mature yet. DC motors and stepper motors are available in different sizes and shapes; however they require drivers and motion controller to operate. Off-the-shelf drivers and controllers are bulky, and therefore not suitable. RC servomotor is a special type of motor consisting of a coreless motor, a potentiometer, a gear box, PD controller and amplifier in one package. Its size and torque is very suitable for this application.

The torque requirements for the motors are estimated using static torque calculation as well as dynamic simulation. To calculate the torque at the joints statically, the configurations of swing leg and stance leg shown in Figure 3.17 are considered. For swing leg, the extreme condition occurs when the leg is straightened and kept at 90^0 with respect to vertical axis. In this configuration, the hip joint is under more load than the knee and ankle joint. Therefore, torque requirement is only calculated for the hip:

$$\tau_{hip} = 0.5kg \times 10cm = 5kg.cm \tag{3.7}$$

For stance leg, the robot configuration in Figure 3.17b. is considered. In this configuration, the thigh and the shank are at 45^0 with respect to the vertical line, and the hip height is at 70% of the full height of straightened leg. During walking, smaller hip height will not be considered because it requires high torque and is not effective in term of

power consumption. In this case, the torque requirement is calculated only for the knee joint because torques at hip and ankle joints is relatively small. Thus:

$$\tau_{knee} = 1.5kg \times 7.07cm = 10.6kg.cm \tag{3.8}$$

Simulation results also show that peak torques during walking is less than 10kg.cm. From torque requirement calculation, it is concluded that the motors used to drive the legs must have torque higher than 10.6kg.cm. Hitec servomotor HS-5945MG has been selected to perform this task. Its specifications are shown in Table 3.3.

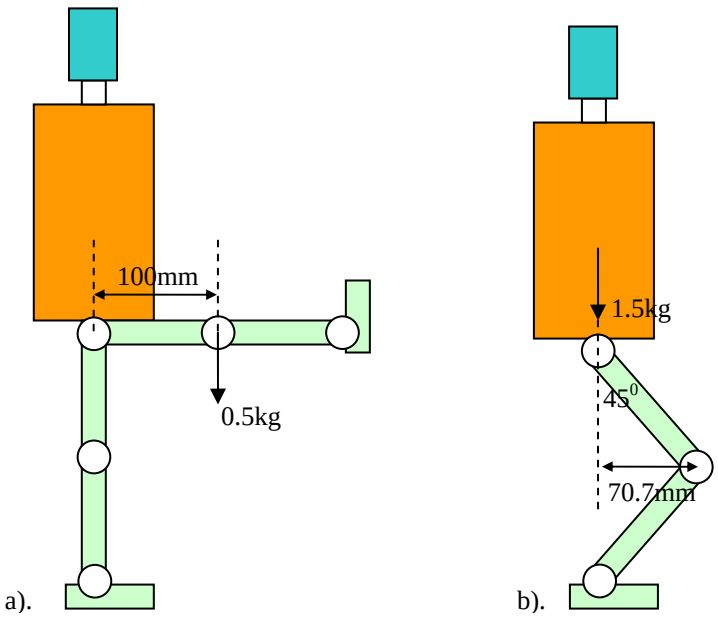


Fig. 3.17. Two configurations of robot used to calculate static torques requirement.

Besides, motors used to drive the head and arms were also selected. The torque requirements for head and arms are less than that of legs' joints. Because the total weight of the head and arms should be less than 20% of the total weight of the robot, the torque of the selected motors should also equal to that proportion of torque of the motors driving the

legs. Specifications of selected motors driving the head and arms are shown in Table 3.4 and 3.5.

Table 3.3. Specifications of Hitec servomotors HS-5945MG

Output Torque	13kg.cm
Speed	$0.13sec/60^0$
Weight	56g
Size	39.4 x 20 x 37.8mm
Amplifier	MOSFET

Table 3.4. Specifications of Hitec servomotors HS-5125MG driving the arms

Output Torque	3.5kg.cm
Speed	$0.13sec/60^0$
Weight	24g
Size	30 x 10 x 34mm
Amplifier	MOSFET

Table 3.5. Specifications of Hitec servomotors HS-81MG driving the head

Output Torque	2.6kg.cm
Speed	$0.11sec/60^0$
Weight	19g
Size	29.8 x 12 x 29.6mm
Amplifier	MOSFET

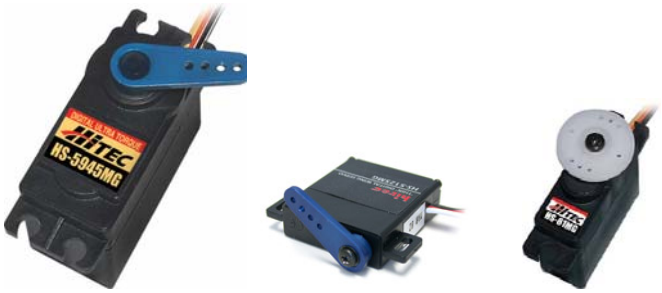


Fig. 3.18. From left to right: HS-9545MG, HS-5125MG and HS-81MG

3.3. Control system

3.3.1. Hardware system

Computer system is the brain of a humanoid robot. It has to process sensory information and control the limbs accordingly. It must possess sufficient computational

power and sufficient number of I/O ports to interface with the sensors and actuators. A summary of I/O signals interfaced to the computer system is presented in Table 3.6.

Table 3.6. I/O signals interfaced to PC-104.

	Devices	**Signals**	**I/O ports**
Sensors (inputs)	Gyros	3 analog voltages	3 A/D
	Accelerometer	3 analog voltages	3 A/D
	Force sensors	8 analog voltages	8 A/D
	Camera	1 asynchronous serial signal	1 RS-232
Actuators (outputs)	Legs	12 PWM signals	12 PWM
	Arms	4 PWM signals	4 PWM
	Head	2 PWM signals	2 PWM
Low level processor	Basic Stamp	1 asynchronous serial signal	1 RS-232

In total, the computer system must have at least 14 A/D and 18 PWM channels together with two RS-232 ports. In addition, a few digital channels are also needed for on/off switches and LED indicators. Since it is costly to support a huge number of I/O channels using a single controller, a two-level computer system is adopted. High level of this system collects the sensory data and performs necessary algorithm to generate control actions. The low level receives control signals from high level and convert them to PWM signals to control the motors. Architecture of the hardware system is shown in Figure 3.19.

The high-level controller is actually a PC system equipped with a DAQ card to interface with the sensors. The most advantage of PC systems is its great flexibility in programming and its computational power compared with microcontrollers, DSP or FPGA, which are gaining more popularity recently for use in embedded systems; however, the commercial PCs are usually bulky. Fortunately, there is a small type of PC called PC-104 with a surprising small form factor of 3.6" x 3.8". PC-104's architecture has all standard interfaces of a normal PC like ISA bus, serial ports, parallel port, USB ports, Ethernet, etc. It has similar features as a desktop computer except for the size.

The low-level controller reduces computational load from the main processor. It interprets the position commands sent by the high-level controller and generates PWM signals accordingly to control the servomotors. A Basic Stamp® BS2p24 microcontroller was used to fulfill this task. This microcontroller can produce high speed and high resolution pulse output. Its processor speed is 20 MHz, and has serial interface of up to 115,200 bps. It can generate a pulse at relatively high resolution of 0.75µs, and therefore, the RC servomotors can have a resolution of 0.083^0 (the servos move 1^0 with pulse width of 9µs).

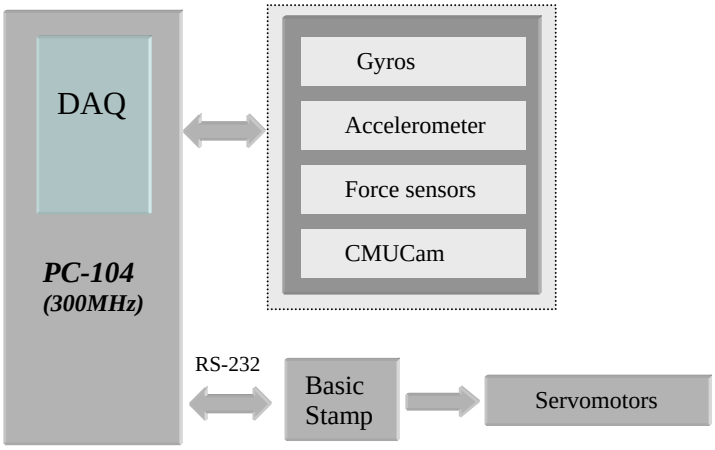


Fig. 3.19. Architecture of hardware system.

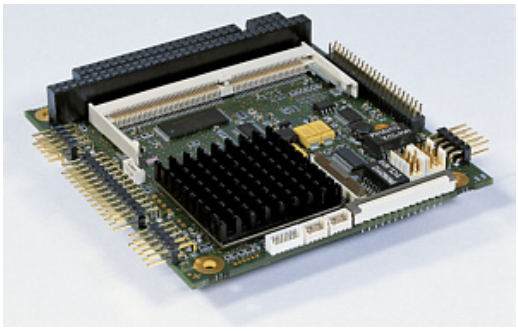
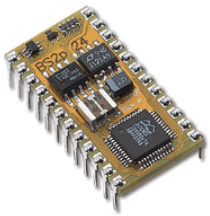
a).  b). 

Fig. 3.20. a). PC-104. b). Basic Stamp® BS2p24.

An interface board has been designed and fabricated to facilitate the connection among various components,. The pin layout and I/O addresses of the board is described in Appendix B.

3.3.2. Operating system (OS)

There are two categories of OS: realtime and non-realtime. Microsoft Windows is a typical non-realtime OS. It tries to optimize the common case where desktop users want to run several applications simultaneously and responsively. Usually, there is no guarantee of processing time for these applications. The users may sometimes encounter unexpected delay, in word processing software for example, when the OS takes care of other programs. This delay usually does not cause major problem to the end user.

However, time delay may cause major problems in some other situations. A playback of a video may become disturbing if the delay lasts for a few seconds. In controlling a robot, a long delay in processing the feedback signal of a PID loop may result in unstable performance.

RT-Linux is a real-time operating system. Other real-time OS are available but they are either costly or less common. RT-Linux is freely distributed and has large user community with huge knowledge repository in various websites. RT-Linux was thus selected to be used for this project.

RT-Linux is not a stand alone OS, instead it runs based on Linux, which is a non-realtime OS. Neither is it an extension of Linux because it runs on its own kernel independent from Linux's kernel. This is a special design of hard realtime OS in which the realtime and non-realtime OS co-exist in one system. This design introduces real-time features without sacrificing important features of the existing non-realtime OS.

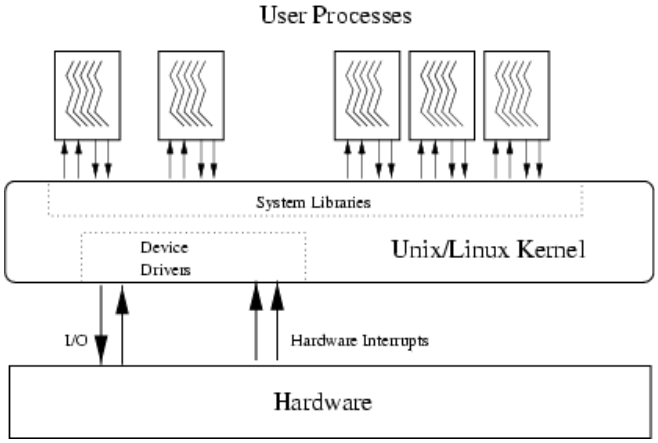


Fig. 3.21. Bare Linux Kernel. (*Adapted from [30])*

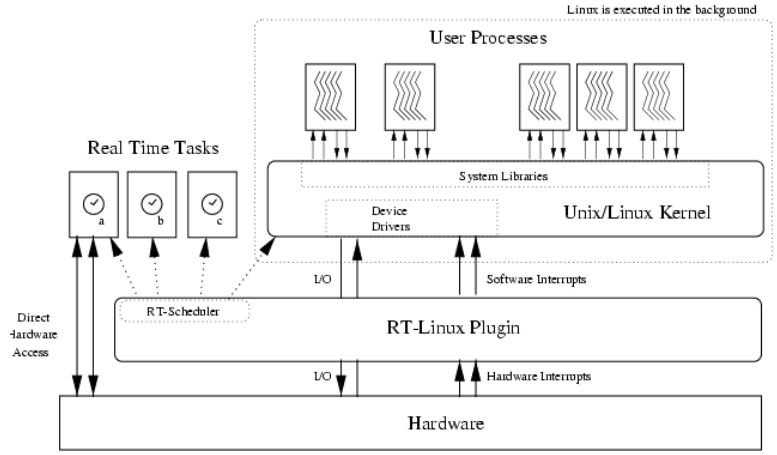


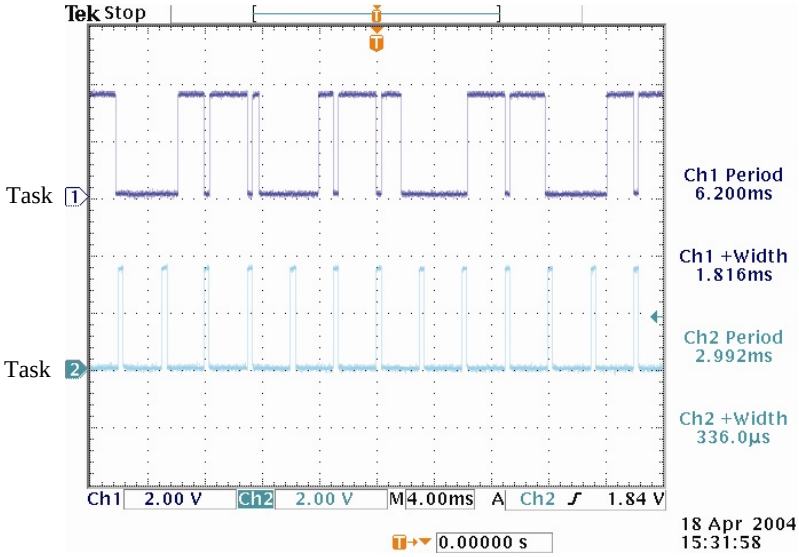
Fig. 3.22. RT-Linux kernel. *(Adapted from [30])*

The design goals of Linux OS are for optimizing the general performance of the system, thus, tasks are treated without priorities. Even the most important and least important tasks are sometimes given even time-slice, so they appear to run simultaneously. Linux processes requests of tasks based on efficient use of hardware instead of their priorities. For example, Linux may treat disk reading request from lower priority process before request from higher priority ones to minimize disk head movement. Therefore, the delay of a process is unpredictable; it depends on the usage of the system at the moment. This

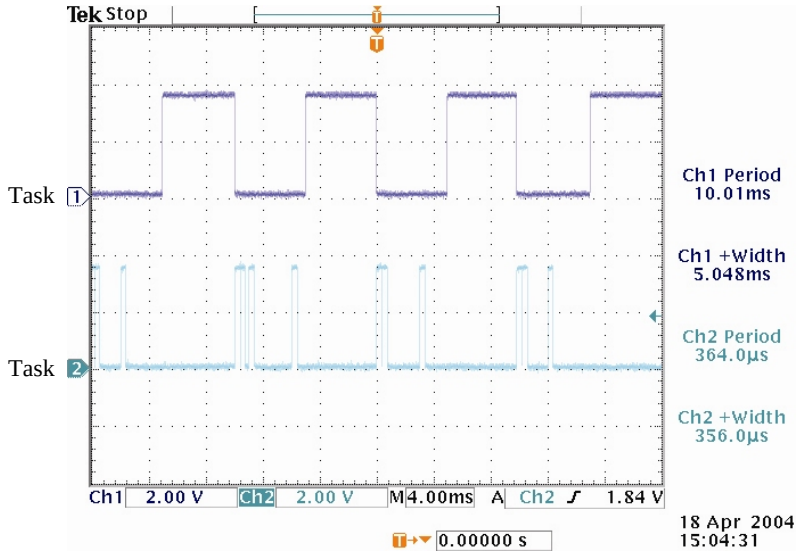
unpredictability makes it impossible for use in robot control. Figure 3.21 shows the Linux kernel without hard real-time kernel. This architecture prevents user programs from directly controlling the hardware as the way the users want. Thus, it is potentially able to suspend running tasks once they outrun the time-slice allotted to them regardless of how extremely important they are, like controlling a robot arm.

To solve this problem, the RT-Linux's developers put a small and predictable kernel in between Linux kernel and the hardware. This hard real-time kernel, which is shown in Figure 3.22, has its own priority-based scheduler. It sees the Linux kernel as a running task like other real-time tasks but with lowest priority. That means real-time tasks will be always executed in prior to Linux tasks when both of them need to be processed. This scheduler will also execute real-time processes based on priorities set by programmers. This feature makes sure that the most important tasks will be executed with least latency.

To illustrate how the priority-based scheduler of RT-Linux works, two simple tasks are implemented. Each task just generates a chain of pulses with different pulse widths and frequencies. Their priorities are varied to see the effect of the priority. In Figure 3.23a, task 1 has lower priority than task 2. Therefore, task 2 has the right to temporarily suspend the operation of task 1 to take control of the PC system. After completion of task 1, task 2 is resumed. Figure 3.23b shows the operation when task 2 has higher priority than task 1. In this case, the period of execution of task 2 is not guaranteed because processing time of task 1 is longer than one period of task 2. From this illustration, we can see that the real-time tasks must be designed carefully to guarantee proper operation when they are run simultaneously.



a) Task 1 has lower priority than task 2.



b) Task 1 has higher priority than task 2.

Fig. 3.23. The execution of two tasks of different priorities.

Chapter 4

Mechanical Design

The mechanical design of ROPE will be presented in this chapter. The design approach for this robot will be introduced in the first section. In the next section, a thorough analysis is carried out in order to develop a set of technical specifications for the robot. The rest of this chapter will be detailed design of the robot, and possible improvements for next design are also recommended wherever appropriated.

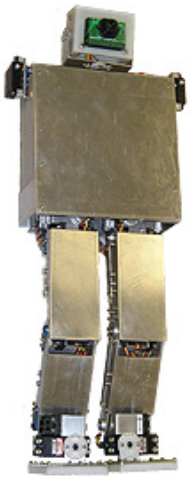
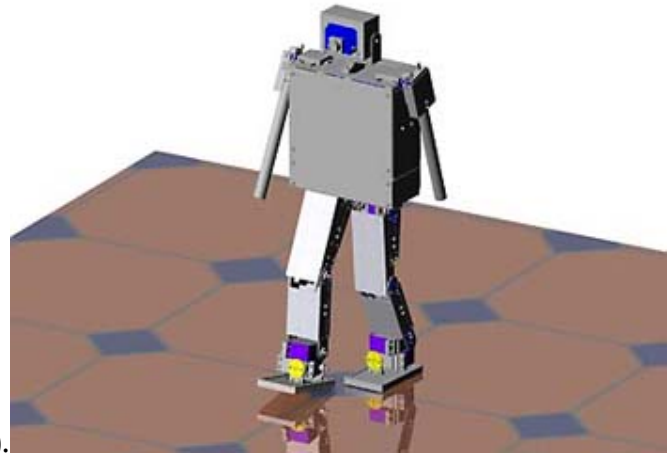
a).  b). 

Fig. 4.1. a). Realized ROPE-II. b). Design of ROPE-II in SolidWork®

4.1. Design approach

4.1.1. Functional decomposition

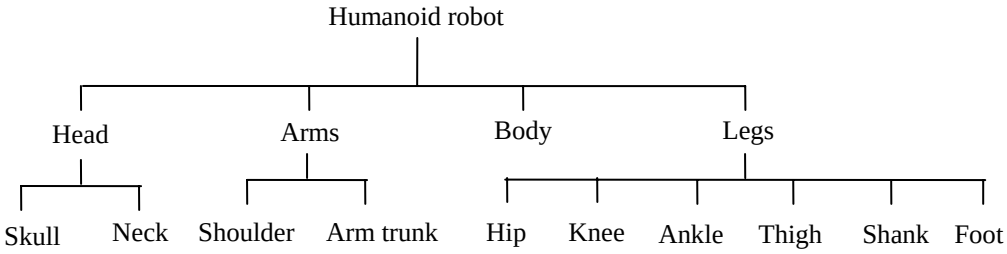


Fig. 4.2. A decomposition of a humanoid robot into its sub-systems

Table 4.1. Functions of sub-systems of ROPE

Sub-systems		Functions
Body		- House computer system, sensors and batteries - Serve as a base to mount other sub-systems
Head	Skull	- House the camera
	Neck	- Join the skull to the body
Arms	Shoulder	- Join the arm trunk to the body
	Arm trunk	- Make up the external shape of the arm
Legs	Hip	- Join the entire leg to the body
	Thigh	- Link between hip and knee joint
	Knee	- Join the shank to the thigh
	Shank	- Link between knee and ankle joint
	Ankle	- Join the foot to the shank
	Foot	-Connected to the ankle and be the support for the whole robot to stand on the floor

Bipedal robot is a complex system with anthropomorphic properties. Its structure is similar to that of human; however, it is difficult to replicate the structure of human's skeleton due to the very large number of degrees of freedom. To design such a complicated system, we have to investigate its specifications as a whole. Based on those ultimate goals, the robot will be broken down into individual sub-systems which are manageable and easier to design.

For humanoid robot, a natural way of decomposing the system is based on particular function of each part as shown in Figure 4.2. Function of each sub-system is presented in Table 4.1.

4.1.2. Design philosophy

ROPE is designed for manufacturing and assembly. All of its parts are designed to be fabricated with traditional machining methods to save manufacturing cost. Sheet metal is used wherever possible to trim down the cost further. However, components made from manually bent sheet metal are not accurate unless die is used. Sheet metal is therefore only used for components which do not require high accuracy.

To facilitate the assembly, common parts are used as many as possible among the sub-systems, especially the joints of the legs. Types of screw are also kept at minimum. The nature of module-oriented design also makes the assembly easier. Assembly is made at two levels: module and sub-module. Individual parts are assembled into modules before modules are assembled together to make a complete robot.

4.2. Technical specifications

4.2.1. Degree of freedom

Fred R. Sias, Jr. and Yuan F. Zheng [15] have done a comprehensive study on the number of degree of freedom that a biped needs to maneuver through most types of terrain. According to this study, each leg must have at least six degrees of freedom: three at the hip, one at the knee and two at the ankle. This configuration is closest to that of human, and not over-complicated to realize. Human limbs actually have more degrees of freedom. They involve joints which are not purely rotational but also translational.

For the upper part of the robot, the design is simpler with less degree of freedom. ROPE is not designed with fully functional arms. Instead, the arm is designed with two-DOF shoulder to which a dummy arm is attached. With this configuration, the arm is able to swing back and forth during walking or make a waving motion. The head also has two DOFs, and therefore can turn up and down and sideways. The body is designed as a rigid object without any DOF. A flexible body with certain DOF can makes the walking easier because the motion of the body trunk can actively shift the CoG around; however, such feature introduces a great deal of complexity to the design due to limited space of the body. In addition, most humanoid robots nowadays can walk with a rigid body. Next design of ROPE can take this feature into consideration. All degrees of freedom of ROPE are shown in Figure 4.3.

4.2.2. Range of joints' motion

Based on zero position and direction of motion of the joints shown in Figure 4.3, expected range of motion of each joint are tabulated in Table 4.2. These are data of human motion, adapted from [41]. These data are considered as guidelines during design. However, the design may not follow exactly these data but will be changed to suit the specific hardware chosen.

Table 4.2. Range of joints' motion. *(Adapted from [41])*

Joints			***Range of motion (degree)***
Legs	Hip	Roll	-25 to 40
		Pitch	-10 to 135
		Yaw	-20 to 65
	Knee		-135 to 5
	Foot	Roll	-60 to 30
		Pitch	-50 to 30

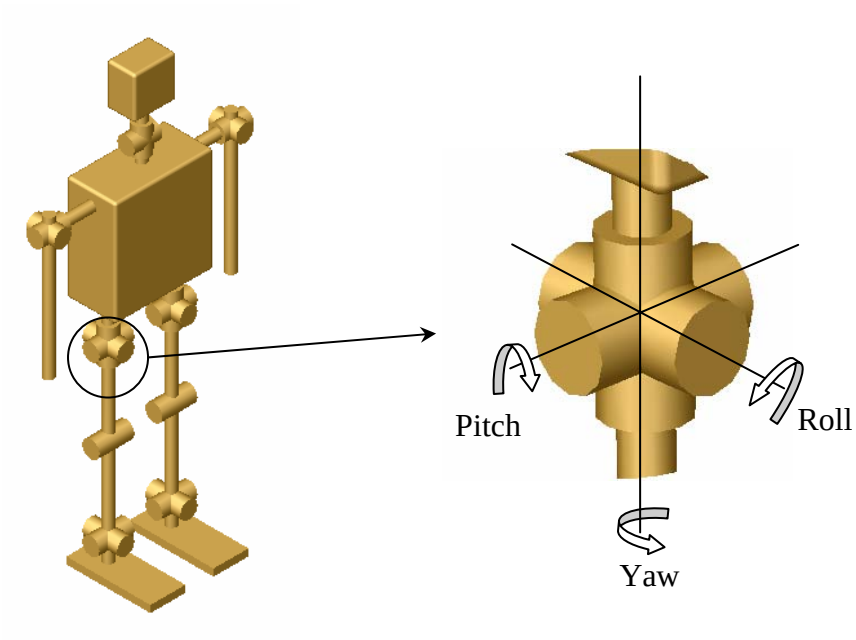


Fig. 4.3. Degrees of freedom of ROPE

4.2.3. Dimension

Dimension is a big issue in design. A rough boundary dimension can be chosen from the power of selected actuator. They were also decided partly based on requirements of the category of the competition in which this robot were intended to participate. Critical dimensions are selected as shown in Figure 4.4.

For a typical adult human, the ratio L:L1:L2:L3:L4 is 1:0.123:0.386:0.247:0.245 [41]. The corresponding desired ratio of ROPE is 1:0.16:0.36:0.2:0.2. This ratio will be achieved through iterative design. First, the desired total height was chosen according to the height limit of the competitions. Then the ratio between the segmental lengths is decided. This ratio was first set to that of human and then iteratively modified until it can accommodate all of the hardware.

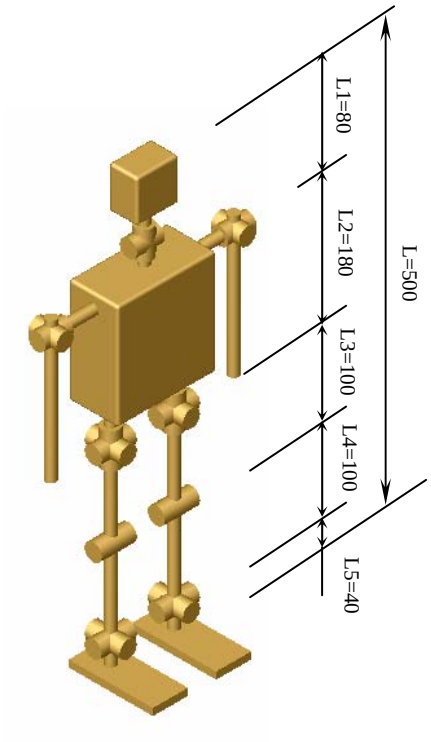


Fig. 4.4. Critical dimensions of ROPE.

4.3. Joint design

4.3.1. Dual-axis module

Considering the configuration of each leg, it is found that both ankle joint and hip joint have a combination of roll and pitch degree of freedom. This fact suggests that it can share the same structure. This design will result in common parts that can be used in ankle and hip joint. With this objective in mind, a dual-axis module is designed for use in roll and pitch joint of the ankle and hip (Figure 4.5).

This module is constructed from a two identical plates made from sheet aluminum. They are arranged in a way such that each plate is connected to both motors but the two plates are not connected. This module is used in hip and ankle joint.

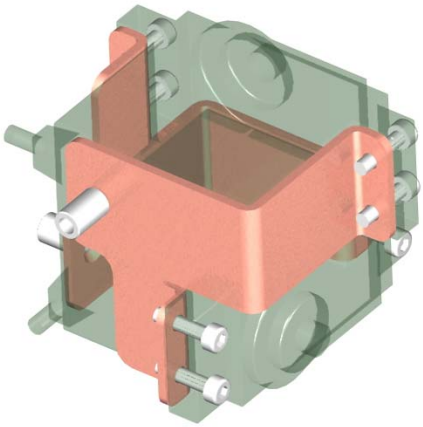


Fig. 4.5. Dual-axis module. The motors are shown transparently.

4.3.2. Hip joint

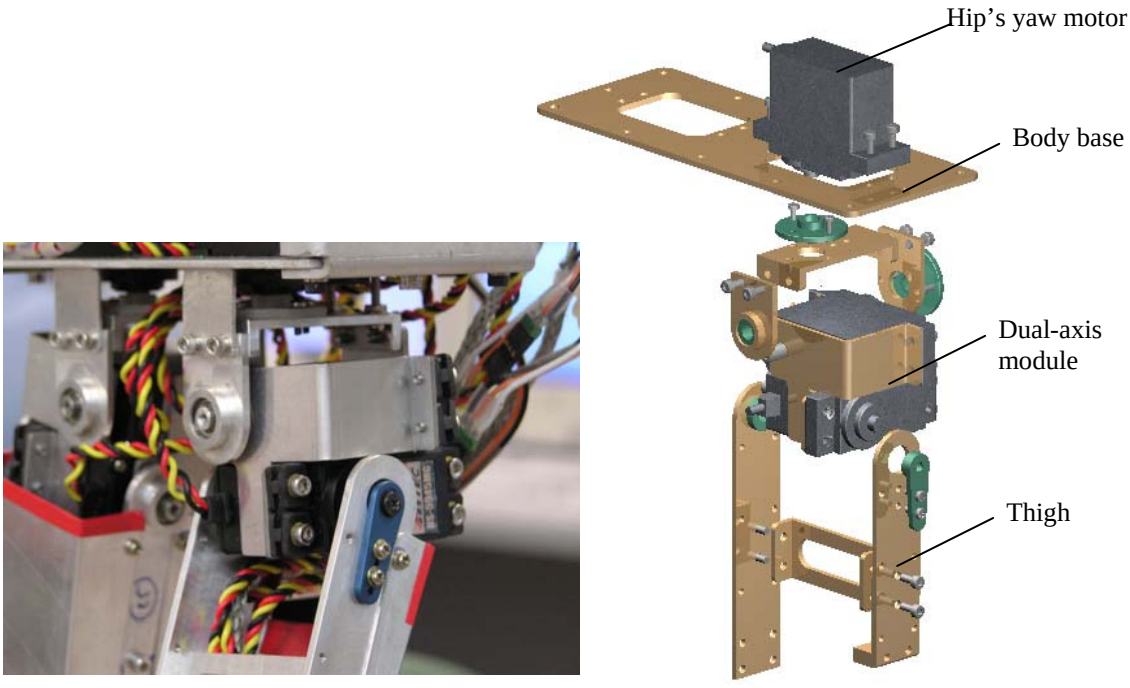


Fig. 4.6. Hip joint. a) The realized joint. b) An exploded view of the joint.

Hip joint has three degrees of freedom: roll, pitch and yaw. The roll and pitch degree of freedom can be realized by using the above dual-axis module. One axis of this module is

connected to lower limb and the other is connected to the body. There are two ways to attach the yaw motor: to the hip or to the body. The latter is chosen because its weight will be supported by the body instead of the leg. This arrangement will make the leg lighter, and therefore less torque is required during the swing phase. Moreover, this arrangement is also suitable for limited space at the hip joint.

4.3.3. Knee joint

The knee joint design is straightforward because it has only one degree of freedom. The important point is the placement of the motor. It is decided to fix the motor to the thigh rather than to the shank. This makes the center of mass of each leg closer to the hip joint. This design helps reducing the load affecting the hip joint. Figure 4.7 shows the knee joint. Knee joint is the connection point between the thigh and the shank.

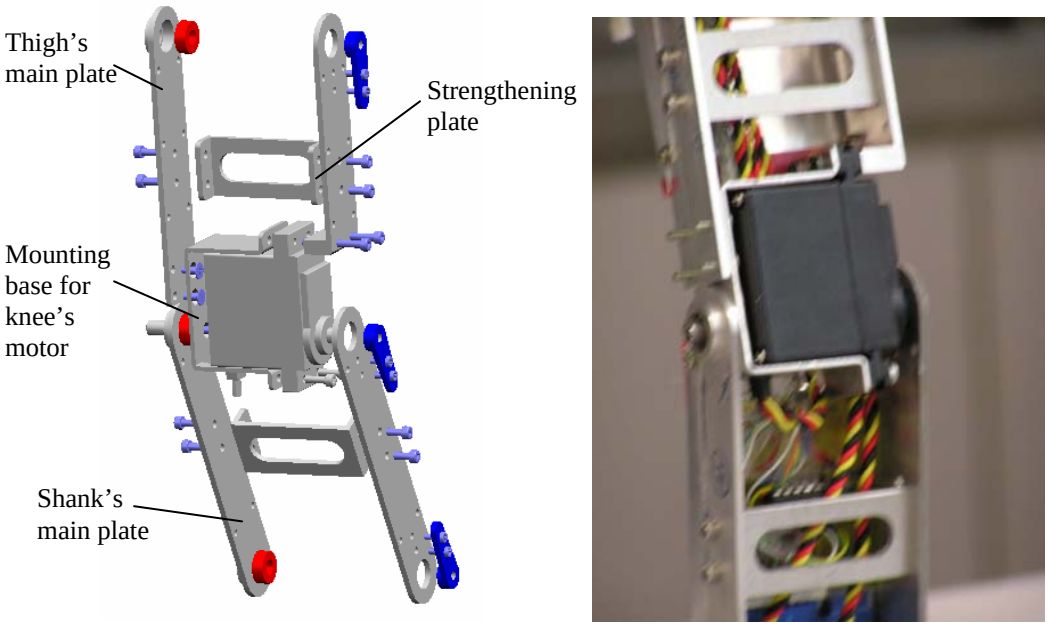


Fig. 4.7. Knee joint. a) An exploded view of the joint. b). The realized joint.

The thigh consists of four components. The mounting base for the motor is a separated part, and it is connected to the rest of the thigh by four countersink screws. This design can greatly simplify the machining. The two main plates are attached rigidly to this motor assembly at one end, and to the hip at the other end. These two plates are further strengthened by a bar connecting them in the middle.

The shank is even simpler with three pieces. There are two main plates connecting the shafts of knee joint to the shafts of the pitch joint of the ankle. A small bar connects them in the middle to strengthen the structure.

4.3.4. Ankle joint

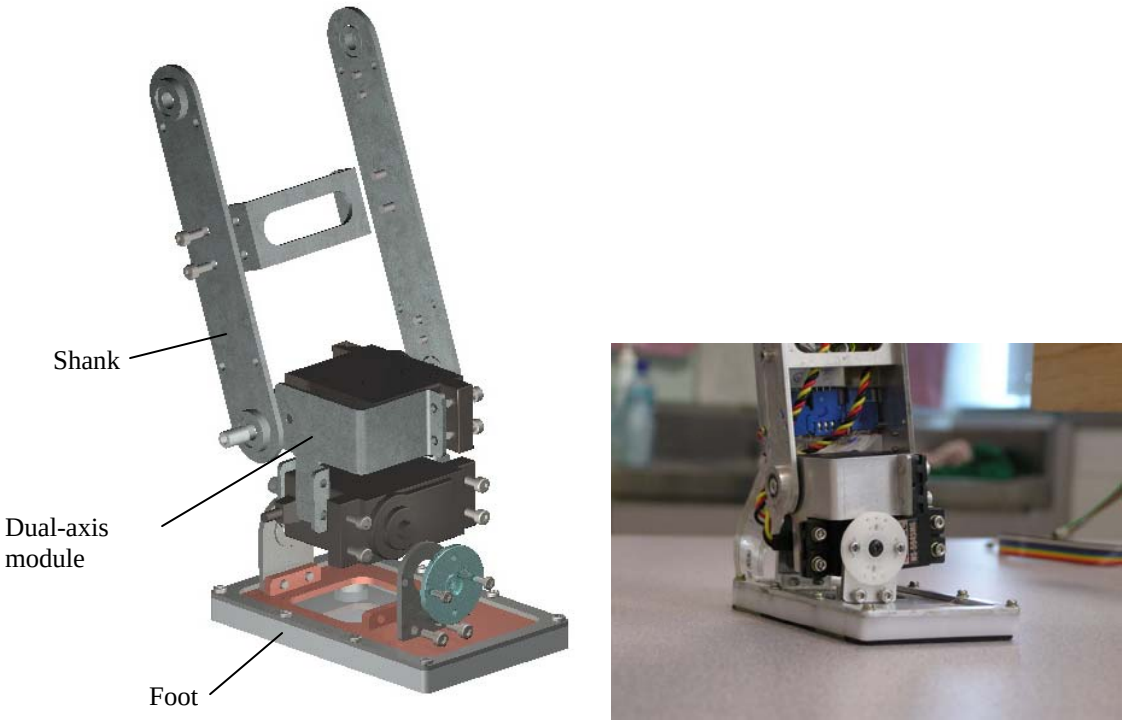
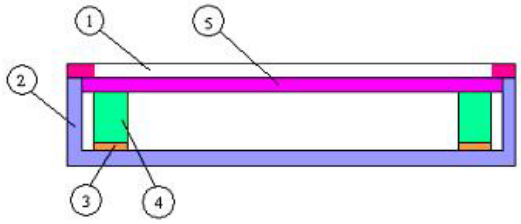


Fig. 4.8. Ankle joint. a). An exploded view of ankle joint. b). Realized ankle joint

Ankle joint has two degrees of freedom: roll and pitch. This can be realized using the dual-axis module. One of its axes is connected to the shank and the other connected to the foot. Figure 4.8 shows the ankle joint.

4.3.5. Foot design

Foot is the part that directly has contact with the ground. It has two critical features: an impact absorption mechanism and ground reaction force sensor.



1. Top Plate
2. Bottom Plate
3. Force Sensor
4. Force-concentrating piece
5. Middle Plate

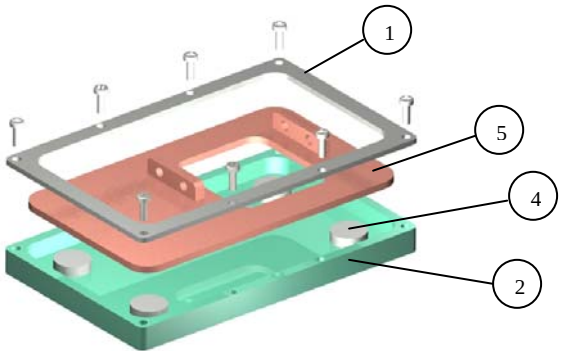


Fig. 4.9. Foot structure. a). Cross-sectional view. b). Exploded view.

The force sensor in use can only detect compression force and is very fragile. The foot must have certain kind of mechanism such that the force applied on the force sensor is always a compression force and receive full impact force once the foot touches the ground. A cross-sectional view of the foot structure is shown in Figure 4.9a. This design has 5 components. One side of force sensor (3) is glued to the bottom plate (2). On top of it, there is a small round piece of plastic (4) used to concentrate applied force onto the

sensor's active area. The middle plate (5), which is connected to the ankle, is put on top of the force-concentrating pieces (4) without any connection, and it is free to move up and down. Now this structure is able to sense the force that acting downward on the middle plate (5). To constrain the middle plate (5) from moving up when the leg lifts up, top plate (1) is fixed to the bottom plate (2) using M2 screws.

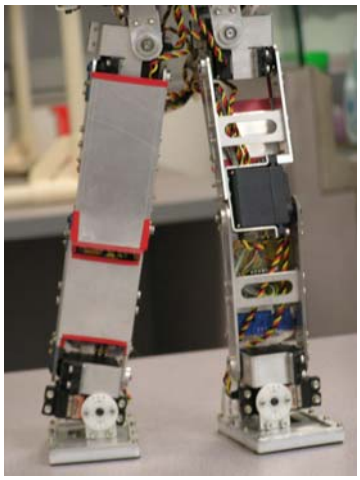
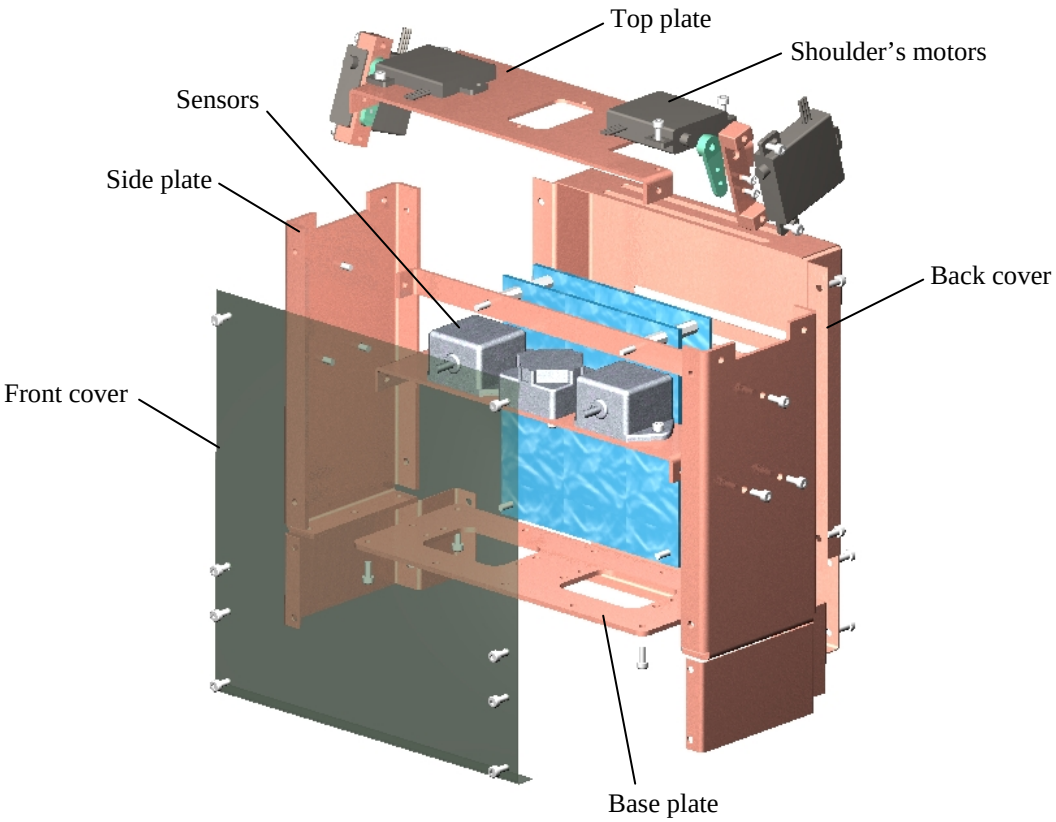


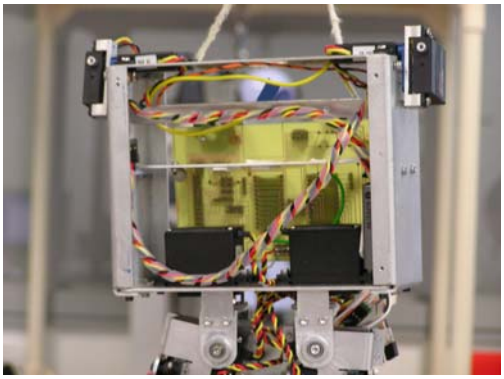
Fig. 4.10. Image of the entire legs with covers on right leg.

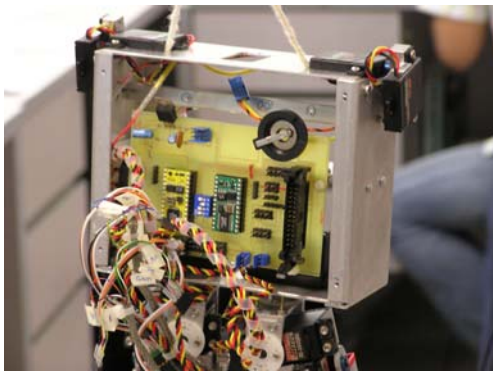
4.3.6. Body design

Human has spine system that provides several degrees of freedom to the body. This gives us flexibility to balance ourselves during walking. Since the control system is so bulky that it occupies most of the space in the body, introducing this feature to robot is not easy, even a single degree of freedom. Therefore, ROPE's body was designed without any degree of freedom.



a)





b) c)

Fig. 4.11. Body structure. a). An exploded view with front cover shown transparently. b) and c). Front and back view of the realized body, respectively.

Being a rigid object, the only function of the body is to house the control system, which consists of PC-104, interface board, the sensors and all kinds of wire. The design is quite straightforward. The structure can be thought of as a box with four surrounding plates. The bottom plate serves as a base of the body and is used to mount the two legs. The top plate is used to mount the shoulder joints of the two arms and the head. Two side plates complete the structure of the body. Inside the box, two crossing bars are used to mount the sensors and the electronic boards. Besides, a front and back cover were also designed to make a good appearance for the robot. These covers are designed with consideration of heat dissipation from the controllers.

4.3.7. Head design

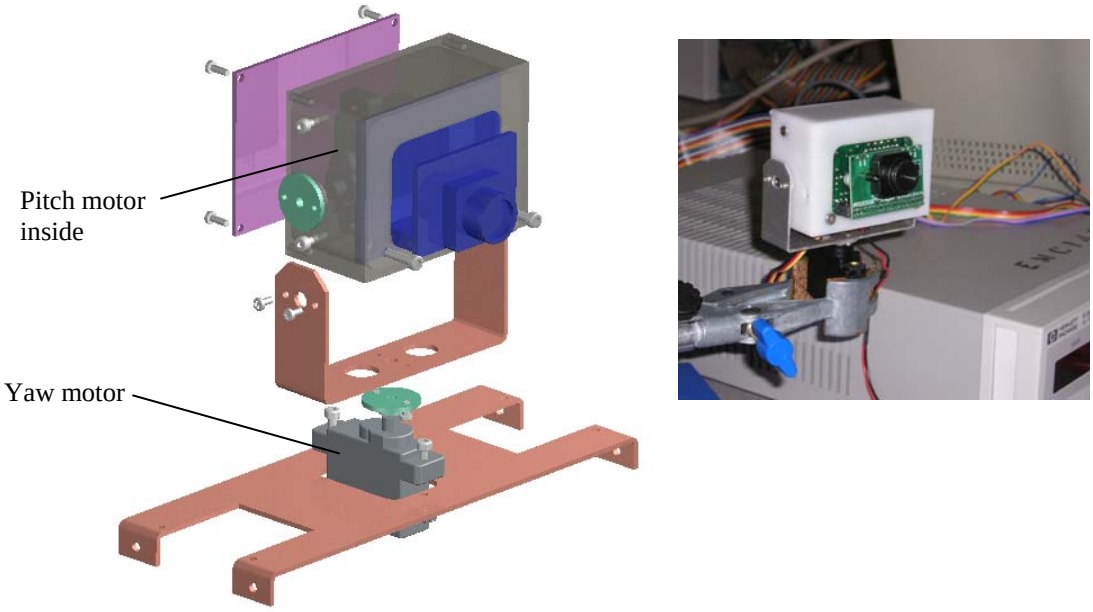


Fig. 4.12. Head's structure. a). An exploded view. b). External appearance of the realized head.

The head is designed to house a camera and has two degree of freedom allowing the camera to turn around for collecting surrounding images. To make it light, plastic material

is selected to create the housing. The structure of the head is shown in Figure 4.12. The u-shaped plate, which is connected to the rotational axis of the box to make the pitch motion, is attached to another motor to make up yaw motion. This yaw motor is in turn attached to the body.

Chapter 5

Bipedal walking control

5.1. Modeling

Several types of model have been developed to analyze kinematics and dynamics of bipeds. The simplest one is a single mass model: a fixed-length inverted pendulum with a single mass located at the body's center of gravity while the legs are assumed massless. This model is simple and intuitive to understand the exchange of potential and kinetic energy during walking. However, it is too simple to capture the actual dynamic of a biped. It can be used to illustrate the nature of bipedal walking but fails to be used as a model to control bipeds.

S. Kajita and K. Tani [20] proposed a more accurate model of bipeds by introducing one degree of freedom along the link of the pendulum to make its length changeable. This model was named Linear Inverted Pendulum because the differential equation describing its motion will be linear if the mass of the pendulum is constrained to move along a straight line. A variation of this model adds another pendulum to account for the dynamics of the swing leg in addition to the main pendulum, which captures the dynamics of the body and stance leg [23]. This model is called gravity-compensated inverted pendulum model. Illustrations of these two models are shown in Figure 5.1.

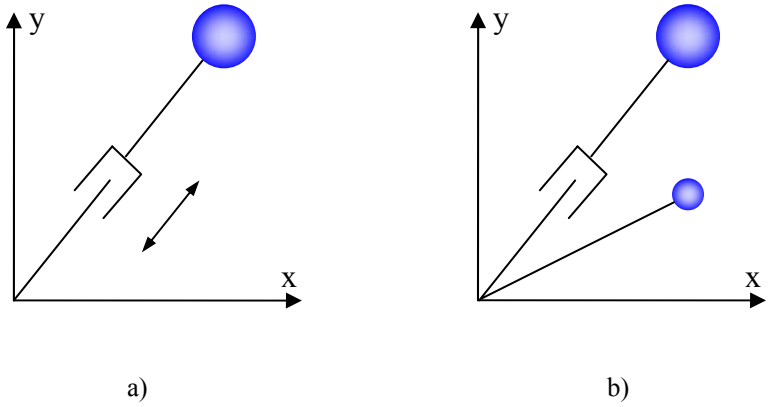


Fig. 5.1. Linear inverted pendulum model. a). Single mass linear inverted pendulum. b). Gravity-compensated inverted pendulum model.

A more comprehensive model of a biped is multilink model which is widely used among researchers. Each link of the model has a counterpart in human, namely, the body, the thigh, the shank and the foot, each of which is modeled as one rigid link. The mass of each link concentrates at its center of gravity. This model is partitioned into two planes, sagittal and frontal planes, for simplicity in analysis. Figure 5.2 and 5.3 show the model on sagittal plane and frontal plane, respectively. This model is suitable for kinematics analysis of bipedal walking; however, it is quite complex to derive the dynamics equation.

In this study, the multilink model is used for kinematics analysis of the robot and serves as a mean to plan the walking for the real robot ROPE. Linear Inverted Pendulum model is used to study the dynamics of bipedal walking in simulation. The actual walking gait realized in real robot ROPE is planned based on kinematics. Future studies can consider realization of dynamic walking based on understandings gained from the simulation.

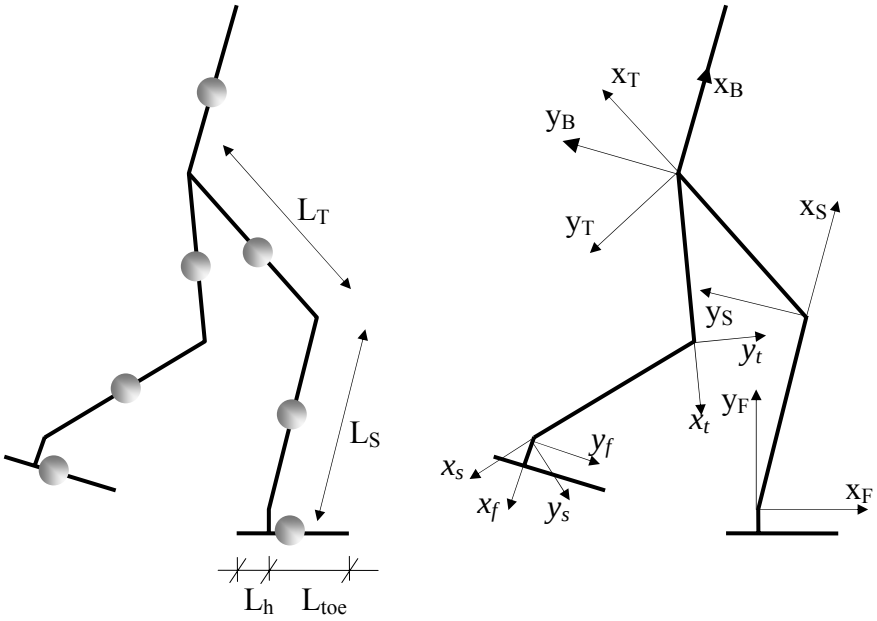


Fig. 5.2. Multilink model on sagittal plane. The frames attached to the links are shown in the right figure. The capital subscript texts refer to the stance leg while the small texts refer to swing leg. F is abbreviation of foot; S is shank; T is thigh; B is body.

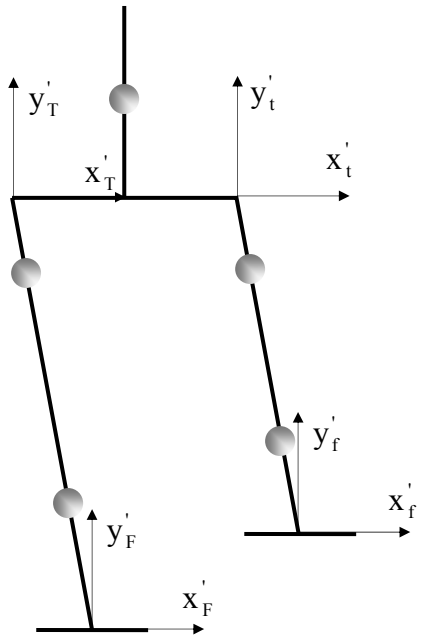


Fig. 5.3. Multilink model on frontal plane.

5.2. Walking cycle

Human walking is a periodic process. A good understanding of its cycle is useful in planning walking for humanoid robot. Each leg in the cycle undergoes two phases: support and swing. The support phases of the two legs within one walking cycle overlaps; therefore, each cycle of a biped can be thought of as consisting of two states: single support and double support. In double support state, both legs are on the ground, whereas there is only one leg on the ground in single support state. It is observed in human that the duration proportion of the two states within one cycle is related to walking speed. Experiments showed slower walking speed is associated with longer double support phase [46]; however, a detailed and accurate relationship between these two quantities has not been obtained yet. Figure 5.4 shows details of walking cycle of human.

Knowledge about human's walking cycle is used to design the state machine for humanoid robot. State machine is a mechanism for determining the state of a biped such that relevant strategy can be applied for that state. State machine is necessary to control a humanoid robot because there is no single function which can describe motion of the biped throughout one walking cycle. Different strategies must be used in different states.

5.3. Cartesian space planning

Cartesian space planning is more effective than joint space planning in high dimensional system. In joint space planning, too many parameters need to be solved until the robot can walk. There was project studying the modification of recorded joint motions of human and applying it to control a biped [31]. This approach is not a good strategy because it does not help understand the system and so much effort will be required to modify human motion data due to differences of dynamics models. Some others use

mathematical tools such as Central Pattern Generator (CPG) to generate the joint motions [26][28][29]. On the contrary, planning in Cartesian space can greatly help reduce number of parameters that need to be tuned, and it is also much more intuitive than joint space planning.

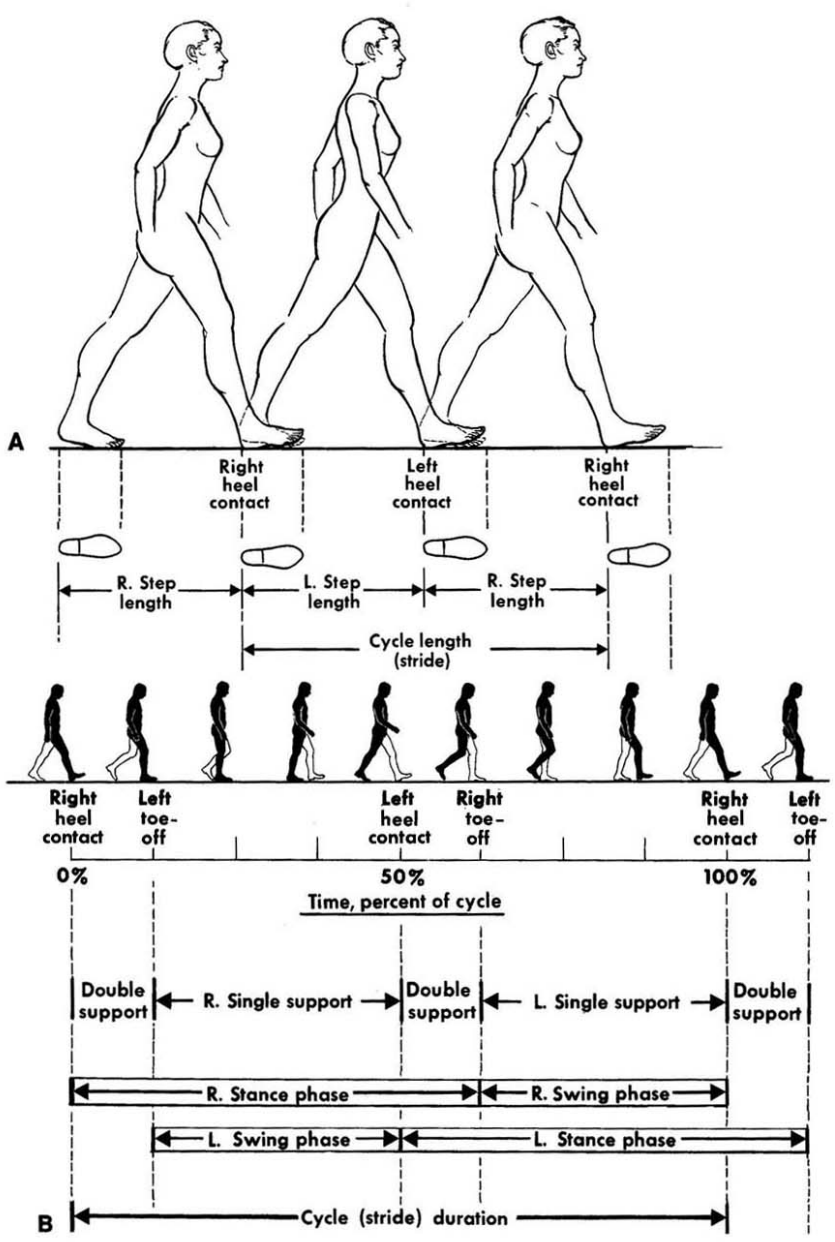


Fig. 5.4. Human's walking cycle. *(Adapted from [46])*

In Cartesian space planning method, the motion of the body is used to obtain joints' trajectories of the stance leg, while motion of swing foot is used to derive joints trajectories of the swing leg. These Cartesian space trajectories are transformed to joint trajectories using inverse kinematics transformation, which is presented in the next section.

The body's trajectory in Cartesian space can be expressed by vector $\left[x_H(t) \quad y_H(t) \quad \varphi(t)\right]$, in which $x_H(t)$ and $y_H(t)$ are coordinates of hip joint with respect to time, $\varphi(t)$ is the tilt angle of the body. Similarly, the swing foot's trajectory is denoted by vector $\left[x_a(t) \quad y_a(t) \quad \phi(t)\right]$, in which $x_a(t)$ and $y_a(t)$ are coordinates of swing ankle joint with respect to time, $\phi(t)$ is the tilt angle of the swing foot. All the vectors are expressed in the base frame attached to the stance foot, which is considered to be fixed to the ground during one step. Figure 5.5 shows a stick diagram of a biped in one step with traces of the body and swing ankle, trajectories of which need to be planned.

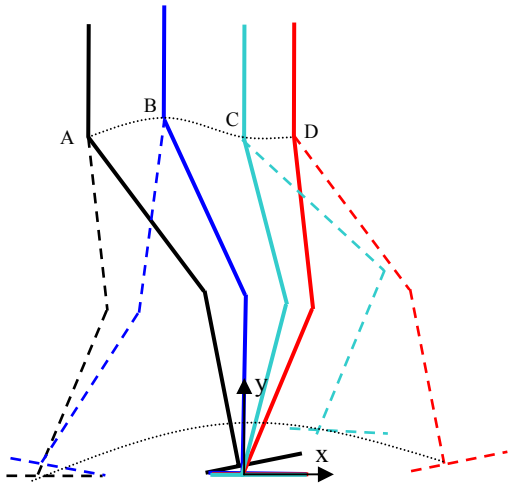
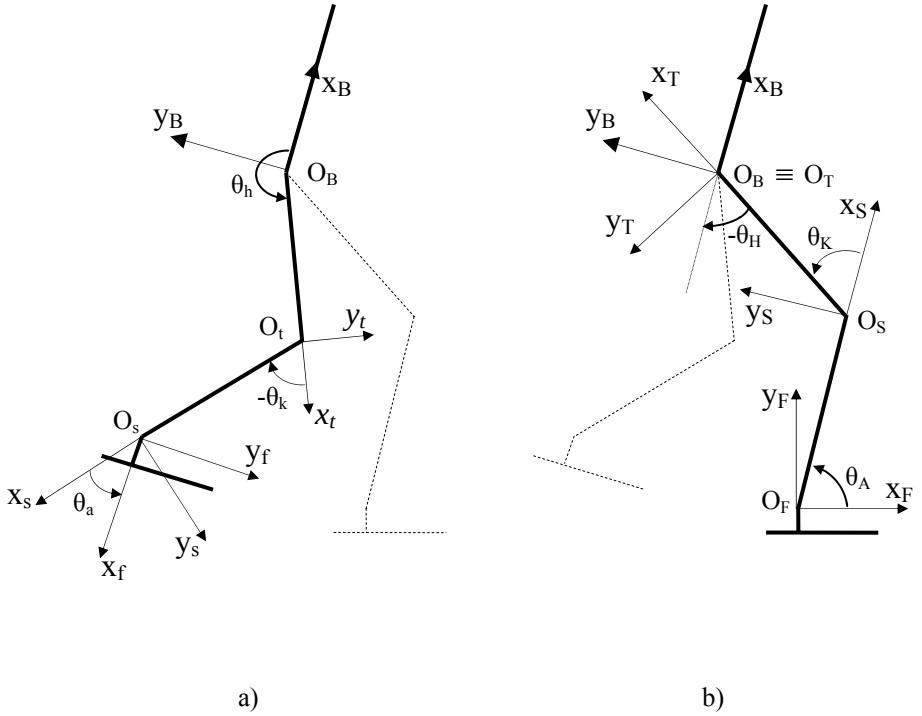


Fig. 5.5. Stick diagram of biped in one walking cycle

In this study, the Cartesian space trajectories which is planned using polynomial interpolation is used in reality to control real robot ROPE; whereas, Cartesian space trajectories which is planned using Linear Inverted Pendulum model are studied in simulation.

5.4. Inverse kinematics transformation

In this section, inverse kinematics transformation is derived to serve as a basis to convert Cartesian space trajectories to joints' trajectories. First, coordinate systems are defined for all the links using Denavit-Hartenberg convention. Frame attachments diagrams are shown in Figure 5.6. The derivation of joints' trajectories is done for stance leg and swing leg individually.



a) b)

Fig. 5.6. Frame attachments on a) swing leg and b) stance leg using D-H convention.

For the stance leg, at a certain point of time the coordinates of the hip with respect to base frame O_F is (x_H, y_H) and the tilt angle of the body is $\varphi(t)$, which are known from the plan. Lengths of the shank and thigh, L_S and L_T respectively, are given. The position of the joints can be calculated using the following formulas, (the derivation of which are presented in appendix D):

$$\text{Knee joint: } \theta_K = \text{atan2}(\sin\theta_K, \cos\theta_K). \quad (5.1)$$

$$\text{Ankle joint: } \theta_A = \text{atan2}(\sin\theta_A, \cos\theta_A). \quad (5.2)$$

$$\text{Hip joint: } \theta_H = \varphi - \theta_A - \theta_K. \quad (5.3)$$

In which:

$$\cos\theta_K = \frac{x_H^2 + y_H^2 - L_S^2 - L_T^2}{2L_S L_T}. \quad (5.4)$$

$$\sin\theta_K = \sqrt{1 - \cos\theta_K^2}. \quad (5.5)$$

$$\sin\theta_A = \frac{(L_S + L_T\cos\theta_K)y_H - L_T\sin\theta_K x_H}{x_H^2 + y_H^2}. \quad (5.6)$$

$$\cos\theta_A = \frac{(L_S + L_T\cos\theta_K)x_H + L_T\sin\theta_K y_H}{x_H^2 + y_H^2}. \quad (5.7)$$

For the swing leg, the joints' positions can also be obtained from the above-derived formulas. In order to utilize those formulas for the swing leg, the coordinates of the swing foot must be expressed in frame O_B attached to the body. However, the swing foot trajectories are not planned in frame O_B but in the base frame O_F attached to stance ankle.

The planning carried out in the base frame O_F is much simpler than in frame O_B because the base frame is fixed during one walking cycle. Because of this, the trajectory of the swing foot, which is planned in base frame O_F, must be transformed to frame O_B before the above formulas can be used to derived joints' positions. To perform this transformation, a homogeneous transformation matrix relating frame O_F to O_B needs to be derived.

The homogeneous transformation matrix transforming coordinates from frame O_B to frame O_F can be obtained from the current coordinates of the body $\begin{bmatrix} x_H(t) & y_H(t) & \varphi(t) \end{bmatrix}$ as follows:

$$T_B^F = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 & L_S\cos\theta_A + L_T\cos(\theta_A+\theta_K) \\ \sin\varphi & \cos\varphi & 0 & L_S\sin\theta_A + L_T\sin(\theta_A+\theta_K) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.8)$$

The homogeneous transformation matrix T_F^B that relates frame O_F to frame O_B is the inverse of T_B^F:

$$T_F^B = (T_B^F)^{-1} = \begin{bmatrix} \cos\varphi & \sin\varphi & 0 & -L_S\cos(\varphi-\theta_A) - L_T\cos(\varphi-\theta_A-\theta_K) \\ -\sin\varphi & \cos\varphi & 0 & L_S\sin(\varphi-\theta_A) + L_T\sin(\varphi-\theta_A-\theta_K) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5.9)$$

Thus, the swing foot trajectory in frame O_B, which is needed to derive swing leg's joints' motion, can be calculated from its planned trajectory expressed in frame O_F by mean of homogenous transformation matrix T_F^B:

$$\begin{bmatrix} {}^B x_a(t) & {}^B y_a(t) & {}^B\phi(t) & 1 \end{bmatrix}^T = T_F^B \begin{bmatrix} {}^F x_a(t) & {}^F y_a(t) & {}^F\phi(t) & 1 \end{bmatrix}^T \quad (5.10)$$

From this trajectory, the joints' position of the swing leg can be computed using the formulas derived for the stance leg:

$$\text{Knee joint: } \theta_k = \text{atan2}(\sin\theta_k, \cos\theta_k) \quad (5.11)$$

$$\text{Hip joint: } \theta_h = \text{atan2}(\sin\theta_h, \cos\theta_h) \quad (5.12)$$

$$\text{Ankle joint: } \theta_a = \phi^B - \theta_h - \theta_k \quad (5.13)$$

In which:

$$\cos\theta_k = \frac{{}^B x_a^2 + {}^B y_a^2 - L_S^2 - L_T^2}{2L_S L_T} \quad (5.14)$$

$$\sin\theta_k = \sqrt{1 - \cos\theta_k^2} \quad (5.15)$$

$$\sin\theta_h = \frac{(L_S + L_T\cos\theta_k)\,{}^B y_a - L_T \sin\theta_k\, {}^B x_a}{{}^B x_a^2 + {}^B y_a^2} \quad (5.16)$$

$$\cos\theta_h = \frac{(L_S + L_T\cos\theta_k)\,{}^B x_a + L_T \sin\theta_k\, {}^B y_a}{{}^B x_a^2 + {}^B y_a^2} \quad (5.17)$$

5.5. Linear inverted pendulum model

S. Kajita and K. Tani [20] have derived that if an inverted pendulum is constrained to move along a straight line, its horizontal motion can be characterized by a linear differential equation:

$$\ddot{x} = \frac{g}{y_c} x + \frac{1}{m y_c} \tau_a \quad (5.18)$$

g is gravitational acceleration; y_c is the intersection of the constraint line with vertical axis y that passes through the ankle joint (Figure 5.7); τ_A is the torque at the pin joint.

Assuming that all the mass of the robot concentrates at the CoG, a biped can be modeled as linear inverted pendulum whose pin joint is the stance ankle joint. If the ankle joint is limp, the horizontal motion of the hip is characterized by the following equation:

$$\ddot{x} = \frac{g}{y_c} x \qquad (5.19)$$

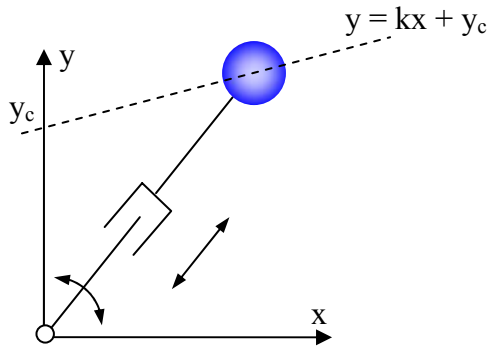


Fig. 5.7. Linear inverted pendulum.

Solution of the above equation is:

$$x(t) = x_i \cosh\left(\frac{t - t_i}{T_c}\right) + T_c \dot{x}_i \sinh\left(\frac{t - t_i}{T_c}\right) \qquad (5.20)$$

in which, $T_c = g/y_c$; x_i, $\dot{x}_i$ are initial position and velocity of the hip on horizontal axis.

Integrate Eq.5.19, we have:

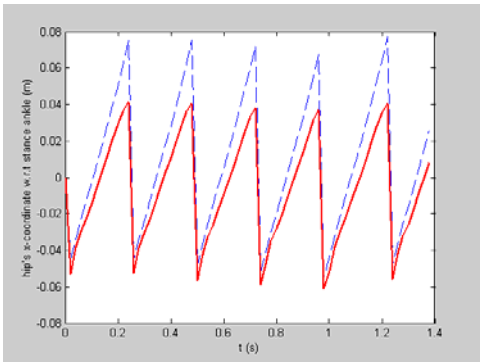
$$\frac{\dot{x}^2}{2} = \frac{g x^2}{2 y_c} + C \qquad (5.21)$$

C is a constant depending on the initial conditions. Because C has dimension of energy per unit mass, S. Kajita and K. Tani called it orbital energy. This quantity is used to decide the placement of swing foot such that the two consecutive steps have the same orbital energy.

To realize this control algorithm, the stance knee joint is used to constrain the hip to move along the constraint line while the stance ankle joint is controlled based on a feedback control law such that the hip can track the trajectory expressed in Eq. 5.20. The hip joint is used to keep the body at upright position.

The placement of the swing foot is decided based on the orbital energy as discussed above. During stable walking where average velocity is maintained, the swing foot lands down at the point where orbital energy of the step is conserved. The average velocity can be changed by varying the placement of the swing foot.

Using this strategy, the simulated biped can be controlled to walk stably both on flat terrain as well as up and down slopes. The tracking of planned trajectories is shown in Figure 5.8.

a). 

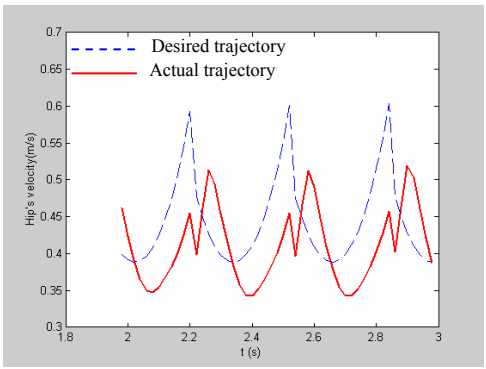
b). 

Fig. 5.8. Trajectories tracking of simulated biped on flat terrain. a). CoG's horizontal coordinate. b). CoG's horizontal velocity.

On flat terrain, the position tracking is done relatively well as shown in Figure 5.8a. However, the velocity tracking is disturbed at the moment when the swing foot touches down, as depicted in Figure 5.8b. At this point, due to the impact of the swing foot on the ground, some kinetic energy is lost; therefore the velocity is decreased abruptly. However, a brief double support after the touchdown of the swing leg can compensate for this error.

In ideal case, the stance ankle torque is limp; however, in reality it must be controlled using feedback control law whose error is the deviation of the actual trajectories of the CoG from its reference trajectories. The reason why the ankle torque needs to be actively controlled is because the model cannot capture all the dynamics of the real robot. Figure 5.9 shows the ankle torque of the stance leg in one cycle. Ideally, the torque must be zero.

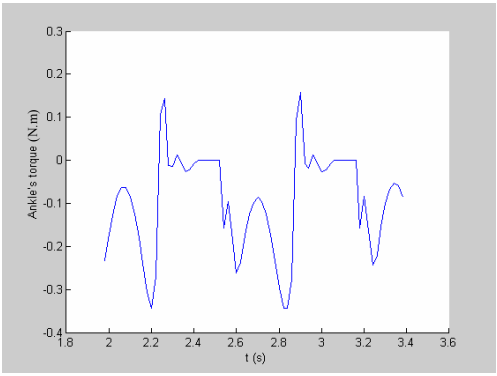
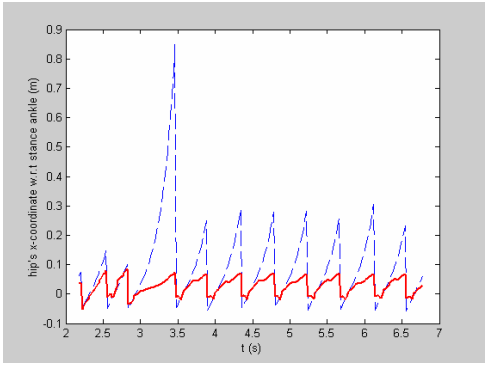


Fig. 5.9. Ankle torque used to compensate disturbances.

a). 

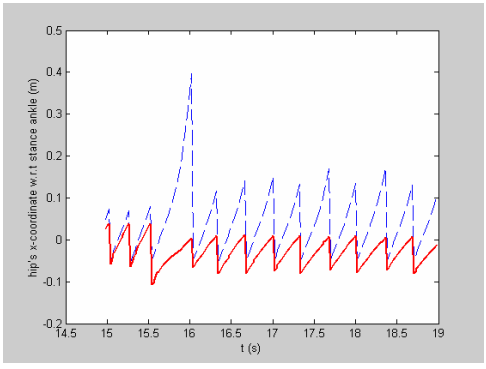
b). 

Fig. 5.10. Trajectories tracking of simulated biped on a) uphill slope and b) downhill slope.

When the robot climbs up or down a slope, the reference tracking is worse than on flat floor as shown in Figure 5.10. During walking on slope terrain, more disturbances will affect the robot under effect of gravity. Therefore, the ankle torque must be actuated more than in the case of walking on flat terrain. On the other hand, the ankle torque is limited to avoid foot rotation. So, the ankle torque easily reaches the limit, and therefore cannot fully compensate for the disturbances. This fact explains the poor tracking of the reference trajectory as observed in the simulation result.

This algorithm is simple to implement in simulation; however, it is harder to implement in real robot using servomotors. Firstly, servomotors use position control method while the ankle joints are commanded in term of torque such that disturbances can be compensated. Secondly, RC servomotors' reference signal is Pulse-Width Modulated signal whose period is 20ms. Therefore, the reference is discontinuous in nature; as a result, the reference tracking of the motors cannot be as good as DC motors. In the next section, a simple and more suitable way of planning the walking for real robots which use servomotors is presented. This method has been used to control ROPE.

5.6. Polynomial interpolation

Linear inverted pendulum may be a good algorithm to implement in robot which is driven by DC motors with continuous reference signals. However, for humanoid robot driven by servomotors, due to the nature of reference signals and position control, linear inverted pendulum may not be applied well. In this section, planning strategy based on polynomial interpolation will be presented. Using this method, we have more freedom in varying the trajectories until the real robot can walk.

To simplify the implementation, a few assumptions are made. Firstly, the body is always kept straight. This assumption agrees with the fact that human walks with a straight body. Secondly, the swing foot is always kept parallel to the ground. This assumption makes sure that when the swing leg lands down on the ground, it is in full contact with the ground. If this condition is not met, i.e. the swing foot touches the ground either by heel or by toe, the biped is not controllable for a short while. This may cause the biped to be unstable. Thirdly, the hip is assumed traveling along a straight line.

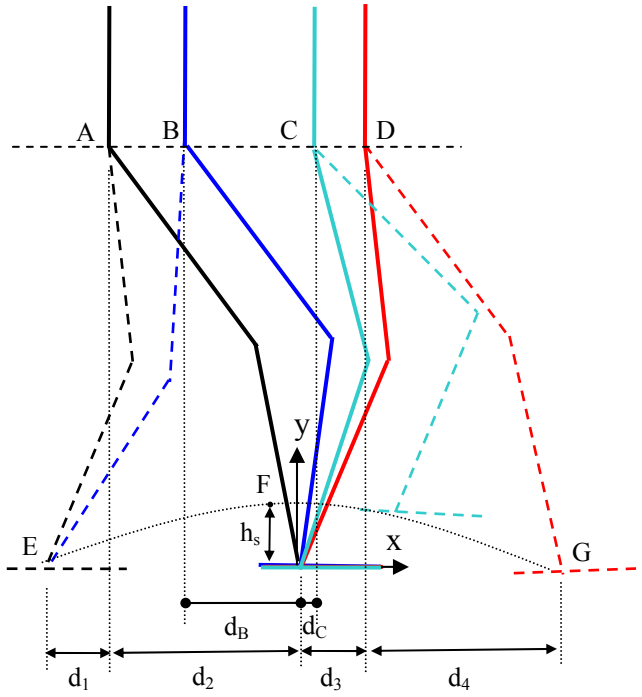


Fig. 5.11. Stick diagram of one walking step. Section AB is corresponding to double support; BC is corresponding to the lift-up of swing leg; CD is corresponding to the land-down of the swing leg.

One walking step is divided into three portions AB, BC and CD as shown in Figure 5.11. Portion AB is associated with the double support, while BC is first half of single support when the swing leg is being lifted up, and CD is the other half when the swing leg is being landed down on the floor. In this step, the left leg is going to swing forward, and

the right leg is the stance leg. The next walking step is just the mirror of this step and will not be discussed.

For continuity between two consecutive walking steps, the geometry configurations of the legs and traveling velocities at the beginning and the end of the step must be the same. Some studies even constrain the continuity of acceleration [19]; however this is not practical in implementation to humanoid robots, especial the ones driven by servomotors. The continuity condition will be:

$$\begin{cases} v_A = v_D \\ d_1 = d_3 \\ d_2 = d_4 \end{cases} \tag{5.22}$$

To plan trajectories for the body, three third-order polynomials are needed corresponding to those three portions of the step. The initial and final position and velocity in each portion need to be given in order to interpolate third-order polynomial.

In portion AB, the initial velocity of the hip will be the final velocity of previous step. This velocity should be the desired average velocity for the stable phase of walking. It will be zero if the robot start to walk from rest. At the end of this double support phase, the hip velocity will be increased by an amount proportional to the difference between the initial velocity and desired velocity. This amount of velocity difference will take effect when the biped is traveling at speed different from desired speed. In other words, it is useful to start the walking from rest. It will not have any effect if the biped is traveling at desired speed. The constraints used to form the trajectory for the hip in double support phase can be summarized as:

$$\text{AB}\begin{cases} x_H(t_A) = d_2 \\ x_H(t_B) = d_B \\ \dot{x}_H(t_A) = v_{k-1} \\ \dot{x}_H(t_B) = v_{k-1} + \alpha(v_{k-1} - v_{desire}) \end{cases} \tag{5.23}$$

t_A and t_B are the times when the robot reaches position A and B; v_{k-1} is the final velocity of previous walking step; v_{desire} is desired velocity; α is a coefficient used to regulate velocity.

The other portions are planned in the same manner. The conditions can be written as:

$$\text{BC}\begin{cases} x_H(t_B) = d_B \\ x_H(t_C) = d_C \\ \dot{x}_H(t_B) = v_B \\ \dot{x}_H(t_C) = v_C \end{cases} \tag{5.24}$$

$$\text{CD}\begin{cases} x_H(t_C) = d_C \\ x_H(t_D) = d_3 \\ \dot{x}_H(t_C) = v_C \\ \dot{x}_H(t_D) = v_D = v_A \end{cases} \tag{5.25}$$

For the swing leg, both horizontal and vertical motions are required. The rear leg start to swing at time t_B, thus the planning is mainly done for interval from t_B to t_D. From time t_A to t_B, the coordinates of the swing ankle in base frame are unchanged. The swing leg's trajectory is divided to two portions EF and FG, with the via-point F determined at the moment the swing ankle is at the highest point. Suppose this point of time is t_F, the horizontal trajectory $x_a(t)$ of the swing ankle can be formulated by the following conditions:

$$\text{EF}\begin{cases} x_a(t_E) = x_a(t_B) = d_1 + d_2 \\ x_a(t_F) = d_F \\ \dot{x}_a(t_E) = 0 \\ \dot{x}_a(t_F) = v_F \end{cases} \tag{5.26}$$

$$\text{FG}\begin{cases} x_a(t_F) = d_F \\ x_a(t_G) = d_1 + d_2 \\ \dot{x}_a(t_F) = v_F \\ \dot{x}_a(t_G) = 0 \end{cases} \tag{5.27}$$

v_F is the horizontal velocity of the swing ankle at point F. Value of v_F can be roughly estimated at double the traveling speed of the hip because it has to move double the distance of the hip in same period of time.

The vertical motion can be formulated by using the following conditions:

$$\text{EF}\begin{cases} y_a(t_E) = y_a(t_B) = 0 \\ y_a(t_F) = h_s \\ \dot{y}_a(t_E) = 0 \\ \dot{y}_a(t_F) = 0 \end{cases} \tag{5.28}$$

$$\text{FG}\begin{cases} y_a(t_F) = h_s \\ y_a(t_G) = 0 \\ \dot{y}_a(t_F) = 0 \\ \dot{y}_a(t_G) = 0 \end{cases} \tag{5.29}$$

h_s is the highest point to which the swing ankle reaches. This height is constrained based on the terrain on which the biped is walking.

By adjusting the via-points of the trajectories, which are B and C for stance leg and F for swing leg, the gait of the robot can be flexibly varied. The Cartesian-space trajectories,

in principle, can be generated online using the actual position and velocity feedback at the beginning of each step. However, this online planning method requires intensive computational power of the controller, and the quality of sensory feedback is not sufficient. In our implementation, Cartesian-space trajectories are planned offline for each type of gait such as starting, stopping and continuous walking. During execution, Cartesian-space trajectories are transformed to joint's trajectories on-the-fly.

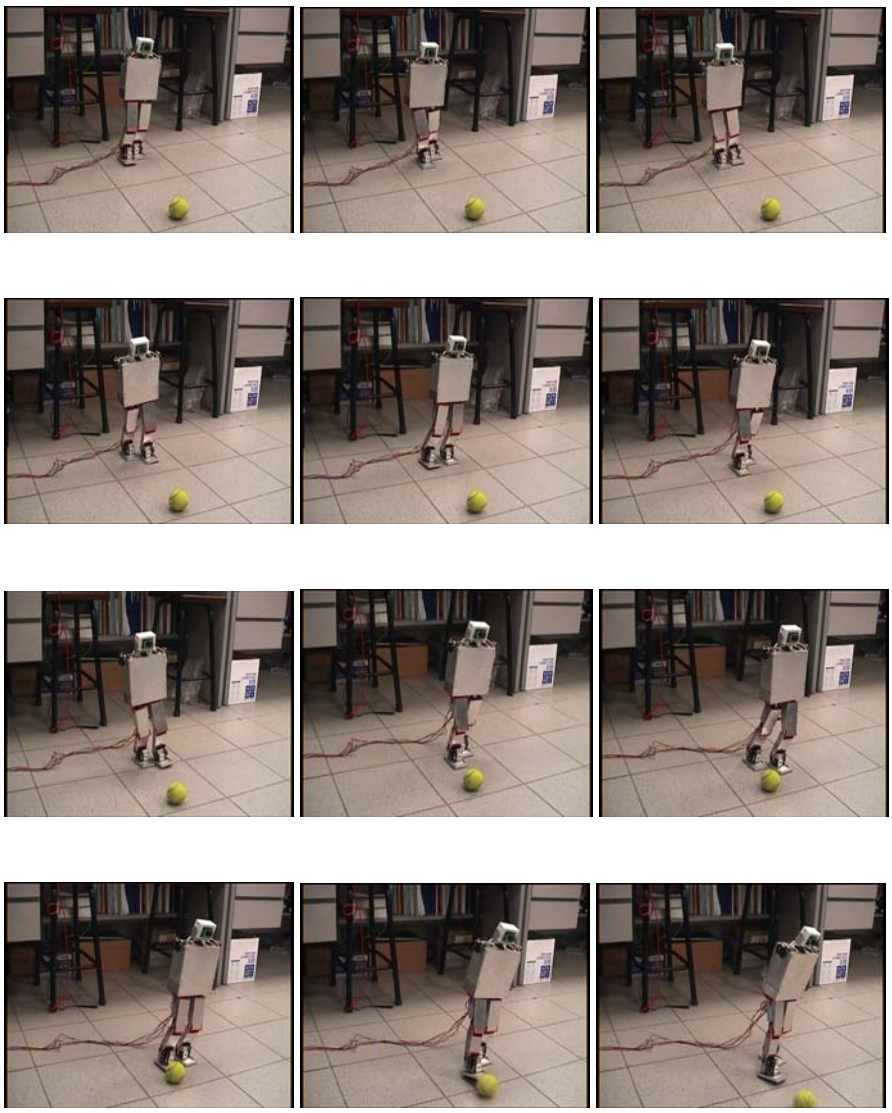


Fig. 5.12. Snapshots of ROPE-II when it is approaching a ball to kick it. *(From left to right and top to bottom).*

Chapter 6

Learning Algorithm

6.1. Introduction

The introduction of several humanoid robots during the past few years has made us think that humanoid robots are going to co-exist with human both at home and at work. Humanoid robots nowadays can walk, run, go upstairs and downstairs, shake hands, answer simple questions, recognize similar objects etc. However, those behaviors need to be planned and programmed carefully by humans. They cannot create new behaviors by themselves. In order to operate in scenarios which are designed for humans, the robots must be capable of learning new things.

Learning is a broad research topic. Humanoid robots can learn to behave properly in the human society. They can also learn to optimize some specific aspects of operation like energy consumption, walking gracefulness, joints' torques. In this study, genetic algorithm will be used to optimize the stance ankle torque such that the resultant horizontal motion of the biped is smooth, which means more graceful and stable. The control algorithm used

to control the biped is the Virtual Model Control which was developed by MIT Leg Lab [17].

In this chapter, the Virtual Model Control framework is briefly reviewed. The application of GA to optimize the stance ankle torque will then be presented. After that, simulation results will be discussed.

6.2. Virtual Model Control

The Virtual Model Control is a control framework that places virtual components at strategic positions to achieve certain behaviors for the robot. Based on the desired effect, the virtual components will generate corresponding virtual forces which will be realized in joint space. One advantage of this algorithm is its simple in implementation. We can plan the motion of the biped in terms of virtual components placed in the Cartesian space, which can be very intuitive, rather than in the joint space. [17] describes details of the Virtual Model Control. In this section, some of its key points will be analyzed. To achieve smooth motion for the body, the ankle torques need to be effectively controlled. The mean to do so will be presented.

As mentioned earlier, the idea of Virtual Model Control is to place virtual components such as spring, damper etc. at appropriate positions in the robot system. In the case of bipeds, the body is connected to a virtual rotational spring and damper to maintain the pitch angle at upright position. Each leg of the biped has two phases: stance and swing. In the stance phase, a virtual spring and damper are mounted vertically on the hip to generate vertical forces that can maintain the body height. In the swing phase, the ankle of the leg is mounted to a spring and a damper vertically and a damper horizontally to maintain swing

height and swing speed, respectively. Figure 6.1 shows the mounting of virtual components on the body, the stance and the swing leg.

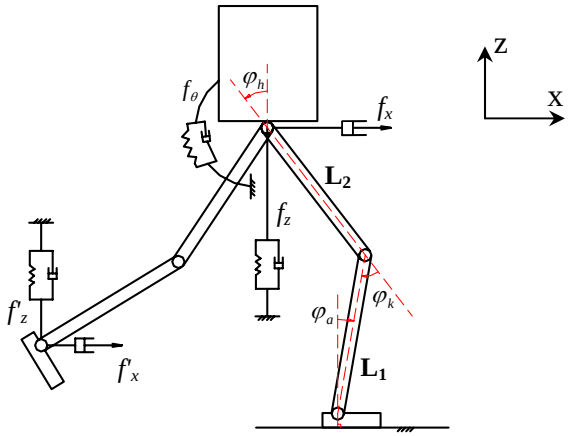


Fig. 6.1. The placement of virtual components on the biped.

For the stance leg, the torques of the ankle, knee and hip joint $[\tau_a \quad \tau_k \quad \tau_h]^T$ can be derived from the virtual forces $[f_x \quad f_z \quad f_\theta]^T$,which are generated by the virtual components, by means of Jacobian:

$$\begin{bmatrix} \tau_a \\ \tau_k \\ \tau_h \end{bmatrix} = \begin{bmatrix} -L_1 c_a - L_2 c_{a+k} & -L_1 s_a - L_2 s_{a+k} & -1 \\ -L_2 c_{a+k} & -L_2 s_{a+k} & -1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\theta \end{bmatrix} = \begin{bmatrix} A & B & -1 \\ C & D & -1 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} f_x \\ f_z \\ f_\theta \end{bmatrix} \tag{6.1}$$

where f_x, f_z and f_θ are the horizontal, vertical and rotational virtual force, respectively, applied on the body of the biped; $c_a = \cos(\varphi_a)$; $c_{a+k} = \cos(\varphi_a + \varphi_k)$; $s_a = \sin(\varphi_a)$; $s_{a+k} = \sin(\varphi_a + \varphi_k)$;

The effect of the virtual force f_x is to pull the biped to move forward while f_z and f_θ maintain the body's height and pitch angle. Ideally, all the virtual forces can be set independently to achieve a desired gait. If the stance foot is fixed to the ground, these effects of the virtual forces can be achieved easily. However, in humanoid robot the contact between the foot and the ground is limp. As a result, if the resultant ankle torque derived from the prescribed virtual forces is too high, the foot may rotate about the toe or heel, and this will lead to unpredictable result. Therefore, the ankle torque must be constrained. As the ankle torque is constrained, the virtual forces cannot be set arbitrarily.

To solve this problem, the virtual forces f_z and f_θ are set independently based on the parameters of the attached vertical and rotary virtual components about the hip, and the ankle torque will be actively controlled such that it can prevent the foot from rotating and achieve f_x. The effect of f_x is to maintain forward velocity, which is taken to be less important than to maintain the body height and body pitch angle. The virtual force f_x is decided to be passive, which means its value is not controlled actively, because it is less important than f_z and f_θ in acquiring a stable gait.

Although the knee torque will contribute to the effect of f_x the ankle torque contributes more significantly to the horizontal virtual force f_x and thus can be used to regulate the walking speed. The ankle torque is formulated as follows:

$$\tau_a = K(v_{desired} - v)\,. \tag{6.2}$$

v is the current forward velocity of the body; $v_{desired}$ is the desired velocity. The optimal value of gain K will be searched by GA.

From Eq.6.1, stance ankle torque can be expressed as follows:

$$\tau_a = Af_x + Bf_z - f_\theta . \tag{6.3}$$

Thus,

$$f_x = \frac{\tau_a + f_\theta - Bf_z}{A} . \tag{6.4}$$

Substituting Eq.6.4 into hip and knee torque equations, we have:

$$\tau_k = \frac{C}{A}\tau_a + (D - \frac{BC}{A}) f_z + (\frac{C}{A} - 1) f_\theta , \tag{6.5}$$

$$\tau_h = -f_\theta . \tag{6.6}$$

In summary, in the stance phase, the joint torques τ_a, τ_k and τ_h are controlled based on Eq.6.2, 6.5 and 6.6, respectively. For the knee torque in Eq.6.5 to be determined, the denominator A= -L_1c_a -L_2c_{a+k} must be non-zero. A equals to zero when $\varphi_k + 2\varphi_a = 180^o$ (for the case of L_1=L_2). This condition only happens when the hip is at the same level as the ankle joint as demonstrated in Figure 6.2. However, this condition may not happen during walking, so we are sure that A is always different from zero.

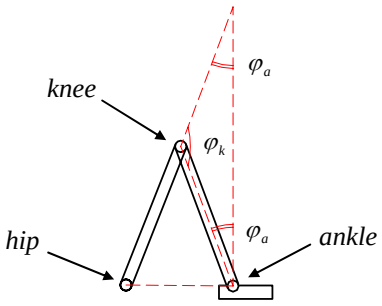


Fig. 6.2. The stance leg configuration when A=0.

For the swing leg, the torque of the joints is easier to control because all the virtual forces can be set independently. The Jacobian for the swing leg is slightly different from that of the stance leg because the reference frame for derivation is different:

$$\begin{bmatrix} \tau_h' \\ \tau_k' \\ \tau_a' \end{bmatrix} = \begin{bmatrix} -L_1 c_{h+k} - L_2 c_h & L_1 s_{h+k} + L_2 s_h & 1 \\ -L_1 c_{h+k} & L_1 s_{h+k} & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_x' \\ f_z' \\ f_\theta' \end{bmatrix}. \quad (6.7)$$

6.3. Genetic algorithm

6.3.1. GA's operations

Genetic algorithm is inspired by the natural selection process that governs the evolution of all creatures. During this process, the fitter individuals have higher chance to survive and the inappropriate ones are gradually phased out. The superior individuals are then mated to produce better and better generations. All creatures in our planet evolve in this manner and have been adapting to the external environment very well. GAs have been being applied successfully in a wide range of fields from biology, medicine, computer science, engineering to social science [48].

In GAs, all the parameters are coded into a string of symbols, usually binary digits {0, 1}. The coded string is called an individual or chromosome. Initially, a population of individuals is generated randomly. GA will test each individual in the population and be returned an associated fitness value. Based on these fitness values, GA's operations will be performed to generate a new generation which will contain fitter individuals. These operations include reproduction, crossover and mutation.

During the reproduction phase, the individuals with higher fitness values will have higher chance to be reproduced. To perform this operation, each individual is fixed to a slot in a roulette wheel. The size of the slots is proportional to the fitness value of the individuals. After each spinning of the wheel, it is obvious that the individuals fixed to larger slots will have higher chance to be selected.

After the reproduction, the individuals in the new generation will be mated randomly to perform crossover operation in which two individuals will exchange their “genes”, which is a portion of the coded string. The crossover operation can be illustrated in Figure 6.3.

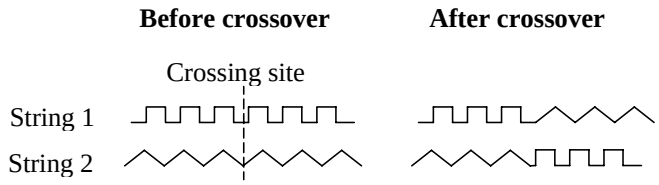


Fig. 6.3. Crossover operation. *(Adapted from [48]).*

The mutation operation occasionally modifies some specific position in the string to try new solutions. This operation sometimes brings in better individuals, but sometimes destroys good individuals. So the probability for this operation to happen has to be low.

One advantage of using GA over other algorithms is that this technique performs the search at different points simultaneously in the searching space rather than at one point. This helps to solve the problem of local optima. The other advantage of GAs is that this algorithm performs the search using the fitness value as the only information. Hence, it does not require the function to be differentiable.

6.3.2. GA’s parameters

In this study, GA is used to search for the optimal value of ankle gain. This gain is coded into a ten-bit binary number. We wish to search for an optimal value within the range from 0 to 15. This range is selected because the gains above 15 always make the biped fall. Therefore, the resolution of ankle gain is $15/2^{10}$. The percentage of crossover is selected to be 90%. The rate of mutation is zero. The parameters of GA are summarized in Table 6.1.

Table 6.1. Value of GA's parameters

GA's parameters	Value
1. Crossover rate	90%
2. Mutation rate	0%
3. Number of individual in a population	20
4. Maximum number of generation	30

6.3.3. Fitness function

Fitness function is one of the most important factors of this searching algorithm. This is the only basis that the algorithm bases on to perform searching operation. The fitness value must reflect the objective that we want to achieve. In this simulation, we want to find the value of ankle gain such that the horizontal velocity profile of the body is smooth and the average velocity is close to desired velocity. So the performance index must consist of both the average amplitude of the velocity profile and the difference between average speed and desired speed. The performance index is as follows:

$$P = \alpha \frac{1}{n} \sum_{i=1}^{n} (V_{\max}^{i} - V_{\min}^{i}) + \beta \left| V_{avg} - V_{desired} \right|. \tag{6.8}$$

n is the number of walking step, and i denotes ith step. $V_{\max}^{i}$ and $V_{\min}^{i}$ V_{min} are the maximum and minimum speed within that step. V_{avg} is average speed of the biped. V_{avg} is equal to the distance of walking divided by the duration of walking. $V_{desired}$ is desired speed of the biped.

The ultimate objective is to find a gain for the ankle torque equation such that this performance index is minimized. In our simulation, we have converted this performance index into a fitness function by taking its inverse. GAs will maximize this fitness function. The fitness function is written as follows:

$$F = \frac{1}{P} = \frac{n}{\sum (V_{\max} - V_{\min}) + n \left| V_{avg} - V_{desired} \right|}. \tag{6.9}$$

The above-derived fitness function is only evaluated for the gain value which is able to make the biped walk stably. There are some gain values which cause the biped fall down. In this case, the fitness function is set to zero so that those values are left out in the next generation. This helps the GA to converge faster.

6.4. Simulation Results and Discussion

6.4.1. Convergence to optimal solution

The simulation is done in Yobotics[1], a dynamics simulation software which allows the running of batches of simulation. After each simulations, fitness values and ankle gains of the individuals are recorded in a data file. The fitness values of 30 generations are shown in Figure 6.4 and the ankle gain of all 600 individuals is shown in Figure 6.5. The convergence occurs after 14 generations.

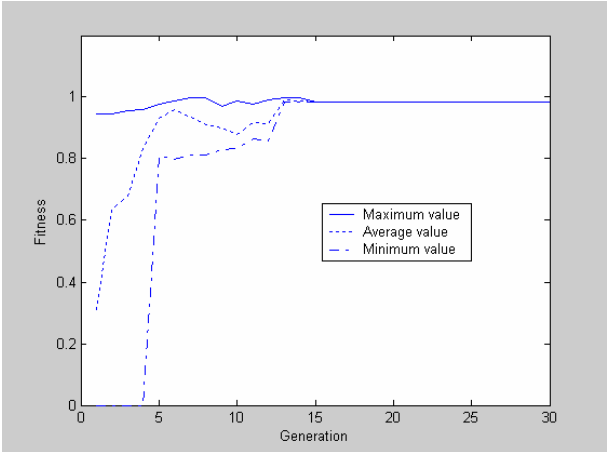


Fig. 6.4. Fitness value of 30 generations.

[1] Developed by Yobotics, Inc., http://yobotics.com

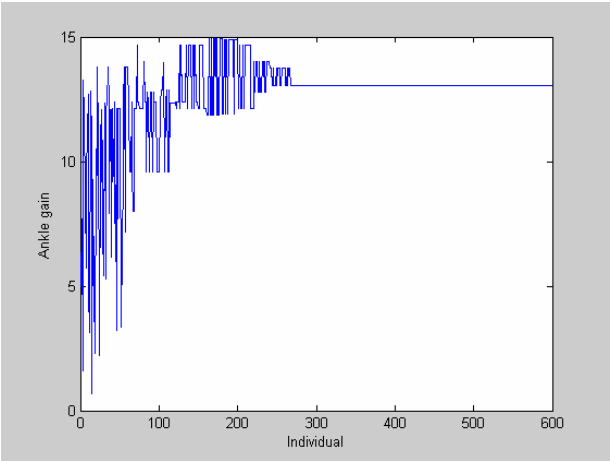


Fig. 6.5. The ankle gain of all individuals.

The range of ankle gain's values shrinks after a few generations and finally converges to a single value. In the first generation, the ankle gains are distributed randomly over a wide range of value. This generation usually contains some values that make the biped fall down quickly. These values of ankle gain are called defective individuals. The fitness values of defective individuals are set to zero such that they have no chance to be reproduced to the next generations. However, defective individuals keep appearing in the next few generations despite of the above filtering mechanism. This phenomenon suggests that the mating of two superior individuals does not guarantee a production of the two new individuals of higher quality. However, the probability for this to happen is very low. As shown in Figure 6.6, the defective individuals keep decreasing and finally disappear after fifth generation.

Velocity profiles of the biped with respect to some values of the ankle gain are shown in Figure 6.7. The optimal ankle gain is K=13.04 which produces a more uniform and lower variation velocity profile than those of non-optimal values. Even though some non-optimal values of ankle gain are able to make the biped to walk longer than the time limit set for the simulation, their derived bipedal motion are jerkier, more oscillatory and less

natural than the one produced by the optimal value. Therefore, it is much better to use a systematic optimization tool rather than a manually tuning of the parameters which is usually stopped at the values resulted in stable but non-optimal gaits.

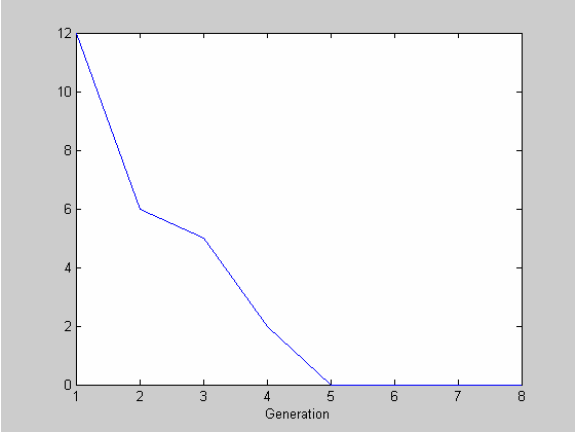


Fig. 6.6. The decline of number of defective individuals over generations.

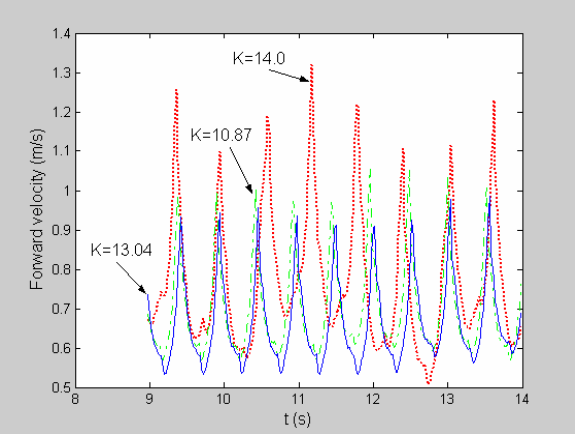


Fig. 6.7. The velocity profiles for the case of optimal ankle gain K=13.04 compared to those of K=10.87 and K=14.

6.4.2. A comparison with enumerative method of optimization

Among the optimization tools, enumerative method is the most costly one. It has to examine every point in the searching space and select the optimal solution. However, the results obtained from this method are reliable because all possible values of the parameters are evaluated and compared. Therefore, this method is used to verify the results obtained

by GA. The enumerative method should only be used in low dimensional space because it is extremely computationally costly. In our study, the time required to run the simulation using this method is almost three-fold that of GA.

In this experiment, the range of ankle gain's value from zero to sixteen is divided into small divisions with increment step of 0.01. The ankle gain at each point within that range is simulated, and the corresponding walking duration and fitness value are collected. Some values of ankle gain cannot keep the biped walking stable but make it fall down after a few seconds. Figure 6.8 shows the durations a biped can walk with corresponding gain's value. It is shown that ankle gain from 8.5 to 15 can make the biped walk stably until the end of the simulation.

The fitness values corresponding to gain values within the range from 0 to 16 are shown in Figure 6.9. From this graph we can see that the optimal value is 13.1 with corresponding fitness value of 1.0113. The optimal solution obtained by GA is 13.04 and 0.9852 for ankle gain's value and fitness values, respectively. These results show that GAs are quite accurate and much more effective than the enumerative method.

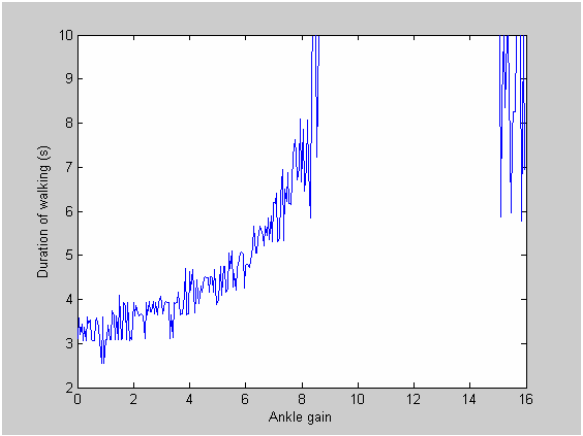


Fig. 6.8. Duration that the biped can walk with respect to ankle gain's value.

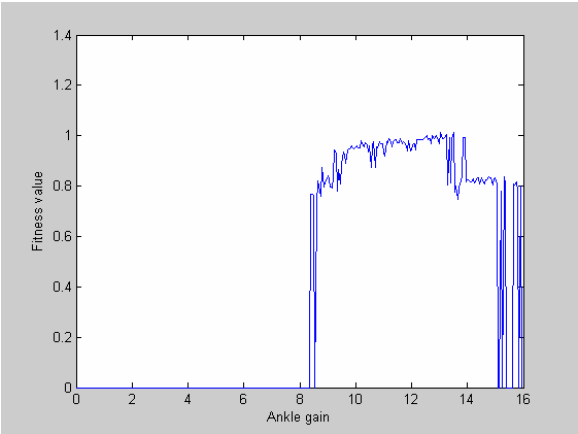


Fig. 6.9. Fitness values vs. ankle gain's value.

6.4.3. Effects of GA's parameters

In the above discussion, the simulation is done with 90% crossover rate. In an attempt to examine the effect of the crossover parameter, we have tried several other values. The results show that when the crossover rate is increased, GA will take longer to converge. Inversely, if the crossover rate is low, GA converges faster. However, in such a case the converged fitness value is usually not as good as that of higher crossover rate. Lower crossover rate means more individuals in the reproduction pool will be retained in the new generation. This will make GA converge faster, but the problem of local optima will arise. Figure 6.10 show the fitness value for the simulation batch with 80% crossover rate.

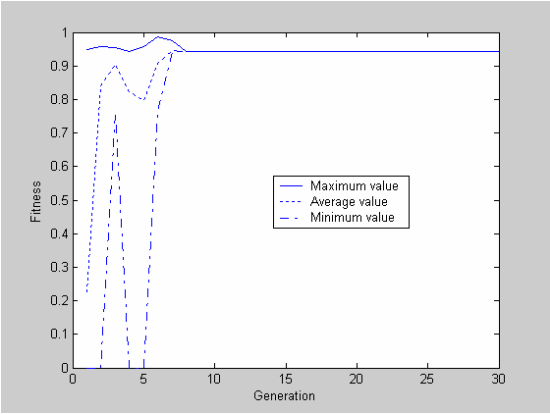


Fig. 6.10. The convergence of fitness values when the crossover rate is 80%

The convergence rate is also affected by the initial population. The initial population in GA is generated randomly, therefore it is quite possible that all the individuals of the population are bad or all are good or a mixture of both. The bad individual is the one that cannot make the biped walk stably. If the initial population contains all bad individuals, GAs may not converge. If there are more good individuals in initial population, the convergence rate is faster. To increase the number of good individuals in the initial population, the size of the population must be sufficiently large.

In summary, simulation results have shown that GAs can be used to search for an optimal value of certain parameter. GAs differ from random search despite it bases on some random choice to make decision. They exploit past experiences to evolve by combining the building blocks of the individuals. Building blocks are sub-strings of each individual that have good contribution to increasing the fitness value. In this simulation, although it is quite a simple search of only one parameter, the main significance of the study is to automate the tuning process with much less effort than manual tuning processes. In future studies, the problem of searching more than one parameter can be considered. In such studies, the fitness function must be formulated carefully because the searching space is much larger than in the case of one parameter.

6.5. Conclusion

In this chapter, we have presented the application of GA to search for the optimal value of ankle gain. The simulation results show that the fitness value tends to increase over generations, and after 14 generations it converges to a single value. The converged ankle gain value results in a smooth motion. The average speed is also close to desired speed. This result has been verified by comparing with the result produced by enumerative method of optimization.

This study is limited to the searching for optimal value of only one parameter whereas biped walking requires several parameters to be controlled. The other parameters are tuned manually. However, this fact does not defeat the purpose of this study because among those parameters that need to be tuned, there are only one or two critical parameters which affect much on the walking stability; other parameters can be tuned quite intuitively.

In future studies, the problem of searching multi-variables should be solved more thoroughly such that GA will converge fast and reliable. Other fitness functions should also be studied.

Chapter 7

Conclusion and Future Works

7.1. Conclusion

This project has realized a small-sized humanoid robot that has basic bipedal motions like walking forward, backward, sideways, and turning. From these primitive motions and with vision system, ROPE can follow a ball and kick it toward a goal. To realize this robot, a unique control system has been designed, in which PC-104 is used as the central processor. Besides, a DAQ card is used to collect sensory information from the sensors. The actuators are directly controlled by Basic Stamp microcontrollers which communicate with PC-104 through serial ports. This control system runs on RT-Linux operating system which is a hard real-time OS.

The sensor system of this robot includes three main components: vision, attitude and force sensors. Vision sensor is a CMUCam placed on the head to collect visual images of the surroundings. Attitude sensor consists of three rate gyroscopes and one 3-axis accelerometer. It is used to collect information related to displacement and orientation of the robot during its operation. Force sensors are placed on the feet to retrieve the location

of the centers of pressure on the feet sole. Signal processing needs to be done in order to interpret the raw sensory signals into useful information.

The actuators in use are servomotors. This type of actuator is suitable for use in small-sized robot because it is compact and has high power-to-weight ratio.

The walking algorithm to control this robot is planned in Cartesian rather than joint space, which is convenient to select suitable parameters and to implement different gaits for the robot. The planned Cartesian-space trajectories are realized by using inverse kinematics algorithm.

Learning algorithm was also studied in simulation. Learning is a promising strategy to improve the performance of the robot. In this study, the Virtual Model Control was used as a framework to control the robot. Genetic Algorithm was used to learn the best value of the ankle gain used in that control framework. The result is a gait that has the smoothest horizontal velocity profile.

7.2. Future works

Other learning algorithm (e.g. reinforcement learning) can be implemented to the real robot to improve its performance. Second, a library of primitive behaviors of the robot can be developed. This library should facilitate the accumulation of behaviors and the combination of the primitive behaviors to carry out more complex behaviors.

Bibliography

Academic Papers

[1] Hoan-Nghia Ho, Chee-Meng Chew, Geok-Soon Hong, Sateesh Talashila. *Using Genetic Algorithms To Optimize Stance Ankle Behavior for Bipedal Walking*, Proceedings of International Conference on Computational Intelligence, Robotics and Autonomous Systems (CIRAS), 15-18 December 2003, Singapore. (A revised version has been submitted to the International Journal of Humanoid Robotics.)

[2] Hoan-Nghia Ho. *A Platform for Developing Humanoid Robot Based on PC-104 Architecture and RT-Linux OS*, Technical report, April 2004.

[3] Humanoid Robotics Institute, Waseda University. *Humanoid Robots in Waseda University-Hadaly-2 and WABIAN*, Autonomous Robots 12, 25-38, 2002.

[4] F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, S. Kajita, K. Yokoi, H. Hirukawa, K. Akachi, T. Isozumi. *The First Humanoid Robot That Has the Same Size as a Human and That Can Lie Down and Get Up*, Proceedings Of The 2003 IEEE ICRA, Taipei, Taiwan, Sep 14-19-2003.

[5] K. Hirai, M. Hirose, Y. Haikawa. *The Development Of Honda Humanoid Robot*, Proceedings of the 1998 IEEE ICRA, Leuven, Belgium, May 1998.

[6] Y. Kuroki, M. Fujita, T. Ishida, K. Nagasaka, J Yamaguchi. *A Small Biped Entertainment Robot Exploring Attractive Applications*, Proceedings Of The 2003 IEEE ICRA, Taipei, Taiwan, Sep 14-19-2003.

[7] F. Yamasaki, K. Endo, m. Asada, H. Kitano. *An Energy-Efficient Walking For A Low-Cost Humanoid Robot PINO*, AI magazine, Vol.23, No.1, pp.60-61, 2002.

[8] T. Furuta, T. Tawara, Y. Okkumura, M. Shimizu, K. Tomiyama. *Design and Construction of a Series of Compact Humanoid Robots and Development of Biped Walk Control Strategies*, Robotics and Autonomous Systems 37, 81-100, 2001.

[9] S.N. Oh, K.I. Kim, S. Lim. *Motion Control of Biped Robots Using A Single-Chip Drive*, Proceedings Of The 2003 IEEE ICRA, Taipei, Taiwan, Sep 14-19-2003.

[10] K. Loffler, M. Gienger, F. Pfeiffer. *Sensor And Control Design Of A Dynamically Stable Biped*, Proceedings Of The 2003 IEEE ICRA, Taipei, Taiwan, Sep 14-19-2003.

[11] Y. Sakagami, R. Watanabe, C. Aogama, S. Matsunaga, N. Higaki, K. Fujimura. *The Intelligent ASIMO: System Overview and Integration*, Proceedings of the 2002 IEEE IROS, EPFL, Lausanene, Switzerland, Oct, 2002.

[12] O. Lorch, A. Albert, J. Penk, M. Gerecke, R. Cupec, J.F.Seara, W. Gerth, G. Schmidt. *Experiments In Vision-Guided Biped Walking*, Proceedings of the 2002 IEEE IROS, EPFL, Lausanene, Switzerland, Oct, 2002.

[13] J. M. Bourgeout, N. Cislo, B. Espiau. *Path Planning And Tracking In A 3D Complex Environment For An Anthropomorphic Biped Robot*, Proceedings of the 2002 IEEE IROS, EPFL, Lausanene, Switzerland, Oct, 2002.

[14] Atkeson CG, Hale J, Pollick F, Riley M, Kotosaka S, Schaal S, Shibata T, Tevatia G, Vijayakumar S, Ude A, Kawato M. *Using Humanoid Robots To Study Human Behavior*. IEEE Intelligent Systems: Special Issue on Humanoid Robotics, **15,** 46-56, 2000.

[15] Fred R. Sias, Jr. and Yuan F. Zheng. *How Many Degree-Of-Freedoms Does A Biped Need?*, IEEE international workshop on intelligent robots and systems IROS 1990.

[16] Arthur D. Kuo. *A Simple Model of Bipedal Walking Predicts the Preferred Speed–Step Length Relationship*, Transactions of the ASME vol. 123, p. 264-269, June 2001.

[17] Jerry Pratt, Chee-Meng Chew, Ann Torres, Peter Dilworth, Gill Pratt. *Virtual Model Control: An Intuitive Approach for Bipedal Locomotion*, The International Journal of Robotics Research, Vol. 20, No. 2, pp. 129-143, February 2001.

[18] Jerry E. Pratt and Gill A. Pratt. *Exploiting Natural Dynamics in the Control of a Planar Bipedal Walking Robot*, Proceedings of the Thirty-Sixth Annual Allerton Conference on Communication, Control, and Computing, Monticello, Illinois, September 1998.

[19] Qiang Huang, Kazuhito Yokoi, Shuuji Kajita, Kenji Kaneko, Hirohiko Arai, Noriho Koyachi, and Kazuo Tanie. *Planning Walking Patterns for a Biped Robot*, IEEE Transactions on Robotics and Automation, vol. 17, no. 3, June 2001.

[20] Shuuji Kajita and Kazuo Tani. *Experimental Study of Biped Dynamic Walking*, IEEE International Conference on Robotics and Automation, Nagoya, Japan, May 21-27, 1996.

[21] K. Mitobe, G. Capi and Y. Nasu. *Control of walking robots based on manipulation of the zero moment point*, Robotica, volume 18, pp. 651–657, 2000.

[22] Jong Hyeon Park and Hoam Chung. *ZMP Compensation by On-Line Trajectory Generation for Biped Robots*, Proceedings of IEEE Conference on Systems, Man and Cybernetics (SMC'99), pp. 960-965 (IV), Tokyo, Japan, October 1999.

[23] Jong H. Park and Kyoung D. Kim. *Biped Robot Walking Using Gravity-Compensated Inverted Pendulum Mode and Computed Torque Control*, Proceedings of the 1998 IEEE International Conference on Robotics & Automation, Leuven, Belgium l May 1998.

[24] Ambarish Goswami. *Foot Rotation Indicator (FRI) Point: A New Gait Planning Tool to Evaluate Postural Stability of Biped Robots*, Proceedings of the 1999 IEEE International Conference on Robotics & Automation, Detroit, Michigan, May 1999.

[25] T. Sugihara, Y. Nakamura, H. Inoue. *Real-time Humanoid Motion Generation Through ZMP Manipulation Based on Inverted Pendulum Control*, Proceedings of

the 2002 IEEE International Conference on Robotics & Automation, Washington, DC, May 2002.

[26] Teresa Zielinska. *Coupled Oscillators Utilized As Gait Rhythm Generators Of A Two-Legged Walking Machine,* Biological Cybernetics, Vol 74, 263-273, 1996.

[27] Gentaro Taga. *A Model of The Meuro-Musculo-Skeletal System For Human Locomotion,* Biological Cybernetics, vol. 73, 97-111, 1995.

[28] Laci Jalics, Hooshang Hemami and Yuan F.Zheng. *Pattern Generation Using Coupled Oscillations For Robotic And Biorobotic Adaptive Periodic Movement,* Proceedings of the 1997 IEEE International Conference on Robotics & Automation, Albuquerque, New Mexico, April 1997.

[29] John S. Bay and Hooshang Hemami. *Modeling Of A Neural Pattern Generator With Coupled Nonlinear Oscillators*, IEEE Transactions On Biomedical Engineering, vol. MBE-34, No. 4 April 1987.

[30] FSM Labs, Inc. *Getting Started with RT-Linux,* Technical Report, April 20, 2001

[31] F.M. Silva and J.A. Tenreiro Machado. *Research Issues in Natural and Artificial Biped Locomotion System*, Proc. 2nd Portuguese Conference on automatic control, pp211-216, Controlo'96, Porto 1996.

[32] Paul G. Savage. *Strapdown Inertia Navigation Integration Algorithm Design Part 1: Attitude Algorithms*, Journal of Guidance, Control and Dynamics Vol.21, No.1, Jan 1998.

[33] Paul G. Savage. *Strapdown Inertia Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms*, Journal of Guidance, Control and Dynamics Vol.21, No.2, March 1998.

[34] G. Capi, S. Kaneko, K. Mitobe, L. Barolli, Y. Nasu. *Optimal Trajectory Generation For A Prismatic Joint Biped Robot Using Genetic Algorithms*. Robotics And Autonomous Systems 38, 119-128, 2002.

[35] D. Golubovic, H. Hu. *A Hybrid Evolutionary Algorithm for Gait Generation of Sony Legged Robots*, 28th Annual Conference of the IEEE Industrial Electronics Society, Sevilla, Spain, November 5-8 2002.

[36] F. Yamasaki, K. Endo, H. Kitano, M. Asada. *Acquisition Of Humanoid Walking Motion Using Genetic Algorithm- Considering Characteristics Of Servo Modules*, Technical Report.

[37] T. Shibata, T. Abe. *Motion Planning By Genetic Algorithm For A Redundant Manipulator Using A Model Of Criteria Of Skilled Operators*, Information Sciences 102, 171-186, 1997.

[38] F. Yamasaki, K. Hosoda, M. Asada. *An Energy Consumption Based Control For Humanoid Walking*, Technical Report, 2000.

[39] J. Kamiura, T. Hiroyasu, M. Miki, S. Watanabe. *MOGADES: Multi-Objective Genetic Algorithm with Distributed Environment Scheme*, Proceedings of the 2nd International Workshop on Intelligent Systems Design and Applications, 2000.

Thesis

[40] Chew Chee Meng. Dynamic Bipedal Walking Assisted by Learning, Doctor of Philosophy Dissertation, MIT, September 2000.

[41] Choong Sim Ming Eddie. *Design of an Anthropomorphic Bipedal Robot*, Master of Engineering Thesis, submitted to NUS, 2003.

[42] Daniel Joseph Paluska. Design of a Humanoid Biped for Walking Research, Master of Science Thesis, MIT, September 2000.

[43] Anthony Hunter. The Mechanical Design of a Humanoid Robot, Undergraduate Thesis, University of Queensland, October 2001.

Books

[44] John J. Craig. *Introduction to Robotics Mechanics and Control*, 2nd edition, Pearson Education, Inc., 1989.

[45] Mark E. Rosheim. *Robot evolution: the development of anthrobotics*, New York, N.Y.: Wiley, 1994.

[46] Verne T. Inman, Henry J. Ralston, Frank Todd. *Human walking*, Baltimore, USA, Williams & Wilkins, 1981.

[47] L. Sciavicco and B. Siciliano. Modeling and Control of Robot Manipulator, Springer, 2nd edition, 1999.

[48] D.E. Goldberg. *Genetic Algorithm in Search Optimization and Machine Learning*, Addison-Wesley, MA, 1989.

Websites

[49] http://www.androidworld.com/prod01.htm

[50] http://www.automation.fujitsu.com/en/products/products12.html

[51] http://www.asimo.honda.com/

[52] http://www.honda-robots.com/english/html/asimo/stage5.html

[53] http://www.sony.net/SonyInfo/QRIO/top_nf.html

[54] http://www.zmp.co.jp/e_home.html

[55]http://www.nasa.gov/audience/forstudents/9-12/features/F_Human_Vestibular_System_in_Space.html

[56] http://www.spp.co.jp/sssj/sirikon-e.html

Appendix A.

Costs

This appendix lists the cost of different items used to make ROPE. These costs are quoted at the time of purchase, i.e. 2002-2003.

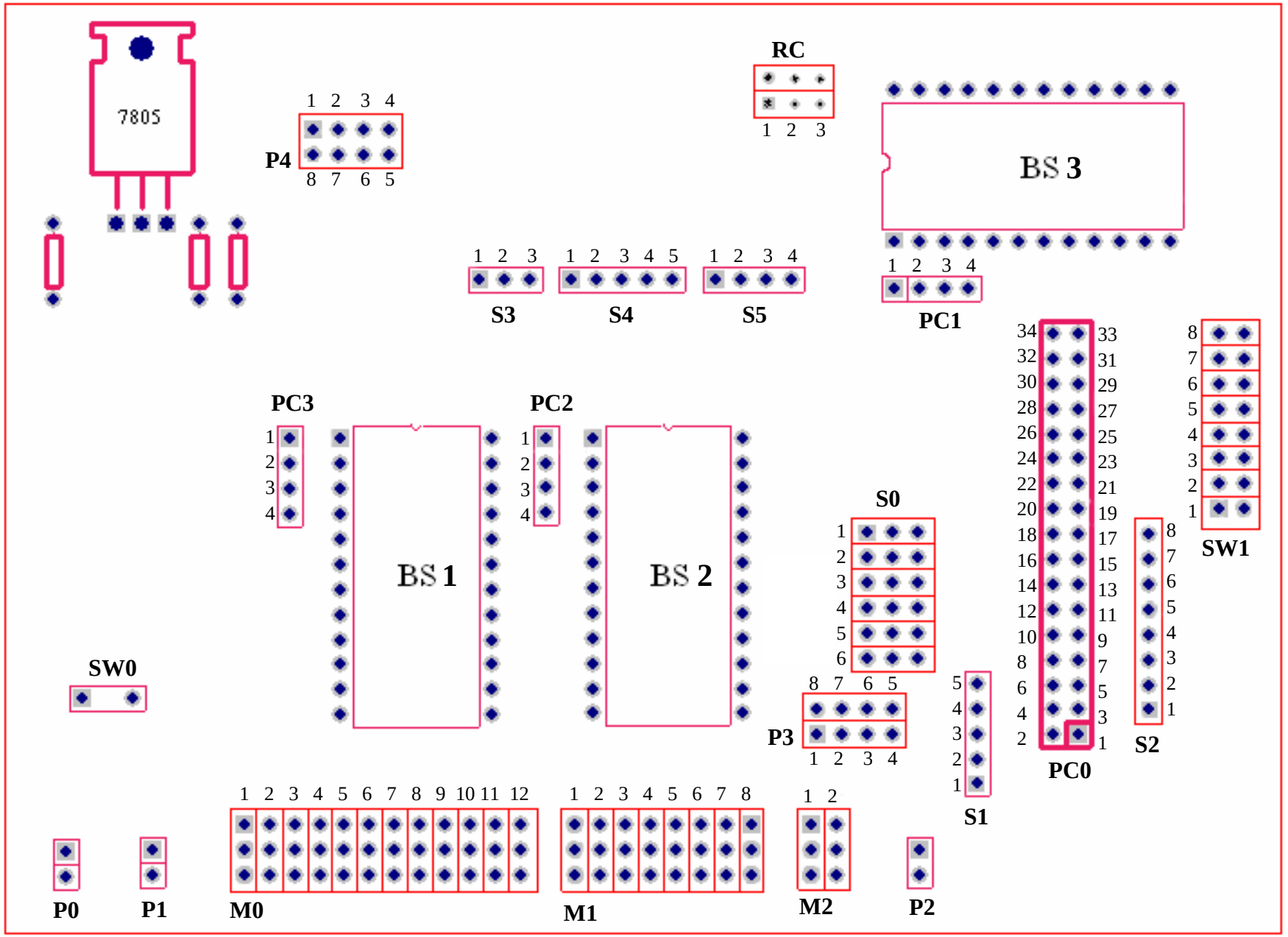
Table A.1. Costs of ROPE.

Items	**Quantity**	**Costs (S$)**
Parts fabrication	80	2,300
RC servomotor	22	2,860
Accelecrometer	1	500
Gyroscope	3	3,300
FlexiForce	8	480
CMUCam	1	110
PCB fabrication	3	30
PC-104	1	1,500
DAQ	1	1,000
Basic Stamp	3	390
Battery	2	220
Screw		50
Total		**12,740**

Appendix B.

Interface Board

Fig. B.1. Pin layout of the interface board.



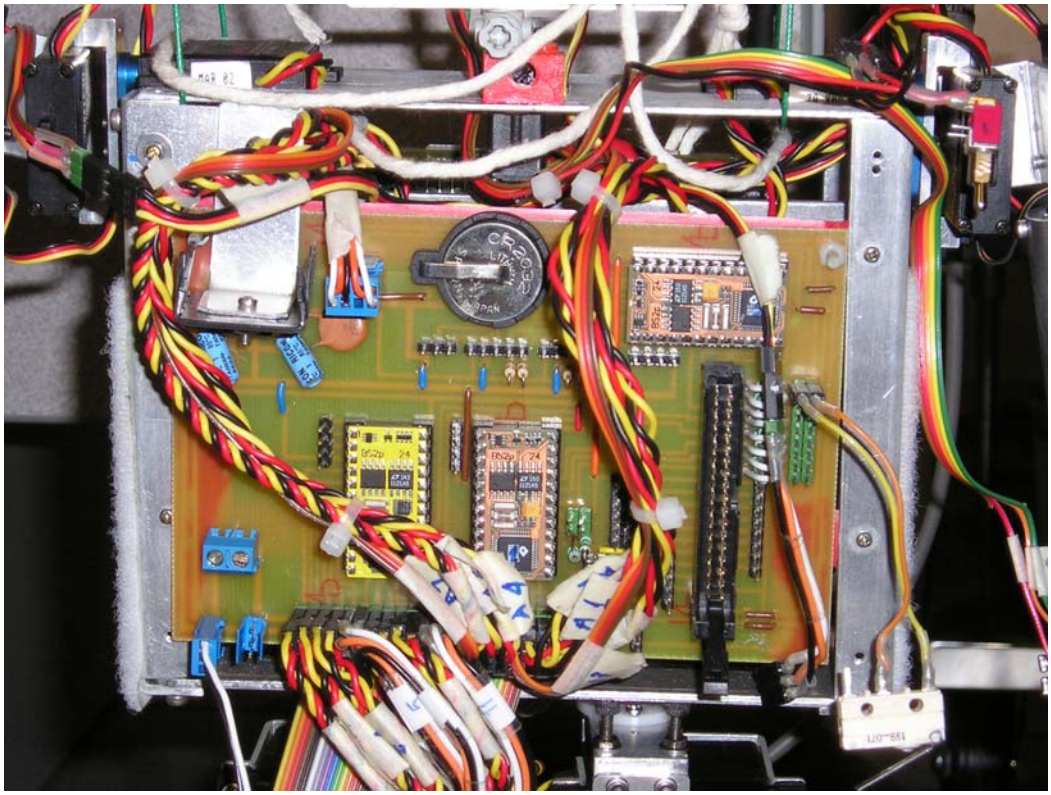


Fig. B.2. A picture of the interface board

The interface PCB board was designed and fabricated to facilitate the connection between the PC104, Basic Stamps, sensors and RC servomotors. There are six types of connectors on the board: P, M, S, SW, RC and PC, which stand for Power, Motor, Sensor, SWitch, Remote Control and Personal Computer, respectively. This appendix presents the pin layout of each connector.

1. Connector M (*for motor connection*)

Connectors of this type are arranged in rows of three pins, which are compatible with the standard connector of any Hitec and Futaba RC servomotors. Each row is used to connect to one motor.

Table B.1. Pin layout of connector M.

M0: for legs' joints		**M1: for arms' joints**		**M2: for head's joints**	
Row	**Joint**	**Row**	**Joint**	**Row**	**Joint**
1	Left hip roll	1	Right shoulder roll	1	Head pitch
2	Left hip pitch	2	Right shoulder pitch	2	Head yaw
3	Left hip yaw	3	Right shoulder yaw		
4	Left knee	4	Left elbow		
5	Left ankle roll	5	Left shoulder roll		
6	Left ankle pitch	6	Left shoulder pitch		
7	Right hip roll	7	Left shoulder yaw		
8	Right hip pitch	8	Left elbow		
9	Right hip yaw				
10	Right knee				
11	Right ankle roll				
12	Right ankle pitch				

2. Connector S (*for sensor connection*)

Connectors of this type are used to interface three rate gyros and three IR sensors to the DAQ card.

a). S0

These connectors are grouped into rows of three pins. The first pin, which is on the left hand side, is VCC; the center pin is GND signal; the third pin is the signal output of the sensors.

Table B.2. Pin payout of connector S0.

Connector S0		
Row	**Sensor**	**Address on DAQ card**
1	IR1	J3-19
2	IR2	J3-17
3	IR3	J3-15
4	Gyro X	J3-13
5	Gyro Y	J3-11
6	Gyro Z	J3-9

b). S1

This connector is used to interface the 3-axis accelerometer to the DAQ card.

Table B.3. Pin payout of connector S1.

Connector S1		
Pin	**Signal**	**Address on DAQ card**
1	VCC	N/A
2	GND	N/A
3	Channel X	J3-3
4	Channel Y	J3-5
5	Channel Z	J3-7

c). S2

This connector is used to interface outputs from 8 force sensors on the feet to the DAQ card. The common ground for these signals is shorted to the ground signal at connector P3.

Table B.4. Addresses of the force sensors' signals on the DAQ card.

Connector S2	
Pin	**Address on DAQ card**
1	J3-4
2	J3-6
3	J3-8
4	J3-10
5	J3-12
6	J3-14
7	J3-16
8	J3-18

d). S3

This connector is used to interface the CMUCam to Basic Stamp 3.

Table B.5. Pin payout of connector S3.

Connector S3		
Pin	**Signal name**	**Address on BS3**
1	GND	N/A
2	RxD	P6
3	TxD	P5

d). S4

This connector is used to interface the compass to Basic Stamp 3.

Table B.6. Pin payout of connector S4.

Connector S4		
Pin	**Signal name**	**Address on BS3**
1	VCC	N/A
2	GND	N/A
3	PWM	P4
4	SDA	P3
5	SCL	P2

d). S5

This connector is used to interface the sonar ranger to Basic Stamp 3.

Table B.7. Pin payout of connector S5.

Connector S5		
Pin	**Signal name**	**Address on BS3**
1	VCC	N/A
2	GND	N/A
3	SDA	P1
4	SCL	P0

3. Connector SW (*for switches connection*)

Connectors of this type are used to connect toggle switches or limit switches.

a). SW0

A toggle switch is connected here to select modes of power supply: single power source or dual power source. This mode-selection switch was introduced regarding to the type of power supply. If an external power supply is used, single power source mode is needed. If batteries are used, dual power source mode is required because two batteries are used to operate the robot. One battery is for the PC-104 system, the other is used to power up the motors.

b). SW1

These are series of multi-purpose limit switch connectors. Their addresses on DAQ card are as follows:

Table B.8. Pin payout of connector SW1.

Connector SW1	
Row	**Address on DAQ card**
1	J4-1
2	J4-2
3	J4-3
4	J4-4
5	J4-5
6	J4-6
7	J4-7
8	J4-8

4. Connector P (*for connection to power sources*)

Connector of this type is for power lines, either power supplied to the PCB board or power from the PCB board supplied to PC-104 or to sensors.

a). P0

This connector is for the external power supply to PC-104 and the sensors. Its voltage is from 6V to 15V.

b). P1

This connector is connected to the power supply for the motors. The voltage supply is from 6V to 8V.

c). P2

This is connector for negative voltage supplied to amplifiers of force sensors. The voltage range is from -6V to -15V.

d). P3

The force sensor amplifiers draw current from this connector.

Table B.9. Pin payout of connector P3.

Connector P3	
Pin	**Signal name**
1	V+
2	V-
3	5V
4	GND
5	V+
6	V-
7	5V
8	GND

e). P4

This connector is the power source of PC-104 system.

Table B.10. Pin payout of connector P4.

Connector P4	
Pin	**Signal name**
1	5V
2	GND
3	N/A
4	3.3V
5	N/A
6	N/A
7	5V
8	GND

5. Connector PC (*for connection to either PC-104 or desktop PC*)

Connectors of this type are used to transmit data between the interface PCB board and PC-104, or between the interface board and desktop computer.

a). PC0

The communication between PC-104 and the various components on PCB board are done through this connector.

Table B.11. Pin payout of connector PC0.

Connector PC0					
Pin	**Signal name**	**Address**	**Pin**	**Signal name**	**Address**
1	GND	GND	18	IR 3	J3-19
2	Accelerometer X	J3-3	19	Limit switch 1	J4-1
3	Force 0	J3-4	20	BS2-P13	COM1-RxD
4	Accelerometer Y	J3-5	21	Limit switch 2	J4-2
5	Force 1	J3-6	22	BS2-P14	J4-9
6	Accelerometer Z	J3-7	23	Limit switch 3	J4-3
7	Force 2	J3-8	24	BS2-P15	J4-10
8	Gyro X	J3-9	25	Limit switch 4	J4-4
9	Force 3	J3-10	26	BS1-P10	J4-11
10	Gyro Y	J3-11	27	Limit switch 5	J4-5
11	Force 4	J3-12	28	BS1-P11	COM2-TxD
12	Gyro Z	J3-13	29	Limit switch 6	J4-6
13	Force 5	J3-14	30	BS3-P13	COM2-RxD
14	IR 1	J3-15	31	Limit switch 7	J4-7
15	Force 6	J3-16	32	BS3-P7	J4-12
16	IR 2	J3-17	33	Limit switch 8	J4-8
17	Force 7	J3-18	34	BS3-P12	J4-17

b). PC1, PC2, PC3

All these three connectors are communication channels between the Basic Stamps and desktop computer for debugging purposes.

Table B.12. Pin payout of connector PC1, PC2 and PC3.

Connectors PC1, PC2, PC3	
Pin	**Signal name**
1	TxD
2	RxD
3	ATN
4	GND

6. Connector RC (*for connection to receiver of remote control*)

This connector is used to interface receiver of radio remote control to BS3. This feature has not been tested actually.

Table B.13. Pin payout of connector RC.

Connector RC		
Pin	**Signal name**	**Address on BS3**
1	GND	N/A
2	GND	N/A
3	5V	N/A
4	5V	N/A
5	channel 1	P14
6	channel 2	P15

Appendix C.

Routines of the control program

1. void * mainthread(void *arg)

This is the main thread of the control program which is executed every 20ms. After doing the inverse kinematics calculation, the joints' position data is sent to Basic Stamp to control the motor to desired position. There is no passing argument or return value for this thread.

2. void * posture(void *arg)

This thread is used to process the attitude sensors, which include a 3-axis accelerometer and three rate gyros. It samples the sensory signals every 10ms. The signal processing includes noise filtering, position and orientation calculation.

3. void * force(void *arg)

This thread is executed every 10ms to sample the force signals on the feet. The center of pressure of each foot is computed when this thread is called.

4. int init_module(void)

This function is used for initialization of the various threads, the variables, the DAQ card, serial ports and the fifos (a mechanism in rt-linux used to exchange information between user program and real-time thread).

5. void cleanup_module(void)

This function is used to close the various devices such as DAQ card, serial ports, etc … when the program is terminated.

6. void SendData(void)

All the desired position data for the RC servomotors will be sent to the Basic Stamps when this function is called.

7. void IntToBytes(int *Data, unsigned char *HighByte, unsigned char *LowByte)

This routine is used to convert an integer number into two bytes. It is a facility to convert position information of RC servomotors to suitable form such that the transmission of these data to Basic Stamp is fastest. The address of the integer and the storage of two bytes need to be passed as arguments.

8. void InvKine (int leg, int state, float x, float y, float theta)

This function will perform the inverse kinematics calculation based on the following arguments passed to it:

- leg: whether it is a RIGHT leg or LEFT leg
- state: whether it is SWING or SUPPORT phase
- x, y, theta: coordinates of the body or the swing foot

9. void MoveJoint (int * Joint, int zero_pos, float angle_in_deg, float vel)

This function is used to move an individual joint from one position to another given the desired velocity. The meanings of the arguments are as follows:

- Joint: a pointer points to a variable whose respective joint needs to move
- zero_pos: the zero_position of that joint
- angle_in_deg: specify the absolute final angle of that joint in degree
- vel: desired velocity specifying the increment of the joint in degree after one cycle of 20ms

10. void MoveBody(int stance_leg, float x, float y, float theta, float x_vel, float y_vel, float theta_vel)

This function is used to move the body forward. The arguments include:

- stance_leg: the current stance leg is RIGHT or LEFT leg

- x, y, theta: the coordinates of the body

- x_vel, y_vel, theta_vel: the amount of change in value of x, y and theta after one cycle of 20ms.

11. void MoveSwingAnkle(int swing_leg, float x, float y, float theta, float x_vel, float y_vel, float theta_vel)

The same as the function MoveBody except that it is for the swing leg.

12. float cur_ang(void)

The current heading angle given by the compass sensor is returned whenever this function is called.

Appendix D.

Inverse Kinematics Transformation

Inverse kinematics transformation is used to relate joints' angle to coordinates of the body. The following derivation is based on [47].

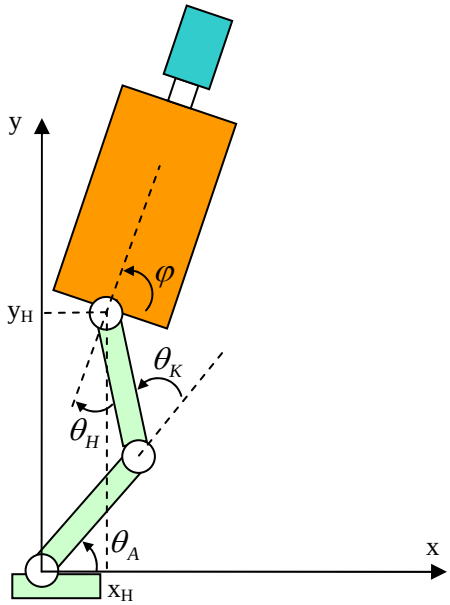


Fig. D.1. Convention of joint angles for inverse kinematics transformation.

Let L_1 and L_2 be the length of the shank and the thigh, respectively. The position of the hip can be expressed as follows:

$$\begin{aligned} x_H &= L_1 \cos\theta_A + L_2 \cos(\theta_A + \theta_K) \\ y_H &= L_1 \sin\theta_A + L_2 \sin(\theta_A + \theta_K) \end{aligned} \tag{1}$$

Squaring and summing the above two equations, we have:

$$x_H^2 + y_H^2 = L_1^2 + L_2^2 + 2L_1L_2 \cos\theta_K \tag{2}$$

From (2), we have:

$$\cos\theta_K = \frac{x_H^2 + y_H^2 - L_1^2 - L_2^2}{2L_1L_2} \tag{3}$$

Thus,

$$\sin\theta_K = \sqrt{1-\cos^2\theta_K} \tag{4}$$

Actually, there should be two solutions for $\sin\theta_K$; however, considering the fact that the knee is always bent backward, its value only takes positive sign.

The knee angle θ_K can be derived using the atan2 function:

$$\theta_K = \text{atan2}(\sin\theta_K, \cos\theta_K) \tag{5}$$

Substitute the above-derived value of knee angle θ_K to equations (1) and solve for $\sin\theta_A$ and $\cos\theta_A$, we have:

$$\sin\theta_A = \frac{(L_1 + L_2\cos\theta_K)y_H - L_2\sin\theta_K x_H}{x_H^2 + y_H^2} \tag{6}$$

$$\cos\theta_A = \frac{(L_1 + L_2\cos\theta_K)x_H + L_2\sin\theta_K y_H}{x_H^2 + y_H^2} \tag{7}$$

From (6) and (7), the ankle angle θ_A can be obtained using atan2 function:

$$\theta_A = \text{atan2}(\sin\theta_A, \cos\theta_A) \tag{8}$$

The hip angle θ_H is calculated as follows:

$$\theta_H = \varphi - \theta_A - \theta_K \tag{9}$$

Appendix E.

Rules of RoboCup Competition

In this appendix the rules of RoboCup is presented. This rules is not the final version and is still under discussion such that it will be fairer and relevant to the current development of technology.

RoboCup Humanoid League 2004 Rules Argument version.

Last Updated: Mar 7, 2004.

Argument version.
Fair copy By Foot-Prints keiichi okamoto hfd01454@nifty.ne.jp

---

1. Definition of humanoid

1.1 Structure

A humanoid robot that is eligible to participate in RoboCup Humanoid League shall meet the following requirements:

A) A humanoid robot shall be able to walk using two legs. No wheel/s shall be allowed to assist its walk.
B) A humanoid robot shall have the approximate body proportions as described in figure.
C) A humanoid robot shall consist of two legs, two arms, one body, and one head.

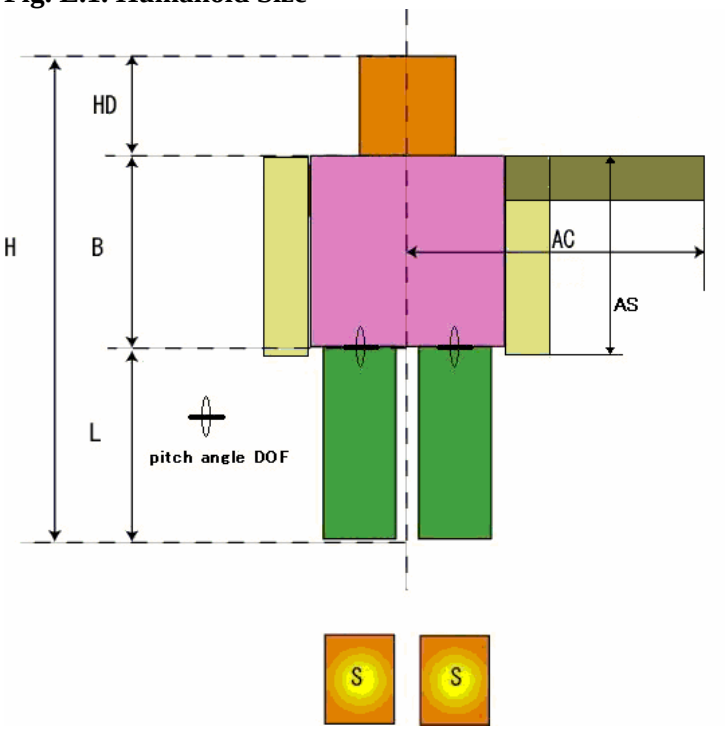
1.2 Proportion

Hmax is a maximum permitted height of the humanoid
H is the actual height of the humanoid
L is the length of the leg
AS is the length of the arm measured from the shoulder
AC is the maximum width of measured from the center of the body
HD is the length of the head, including the neck.

$0.4 * H < L < 0.6 * H$
$2 * AC < H$
$0.1 * H < HD$
$S < (H/3 * H/3)/2$

A tolerance of 10% is applied to the relative proportions as well as to Hmax, except for the H-120 league where Hmax is 180 cm.
The foot of the robot shall not overlap while standing, and a rectangle shaped surface (S) of each foot must satisfy: $S < (H/3 * H/3)/2$.
The humanoid should be able to stay in equilibrium on one leg during one minute (this will force the number of degrees of freedom of the legs of the robot)

Fig. E.1. Humanoid Size



1.3 Specific Dimensions

This section provides concrete examples of the specific proportion of the humanoid robot for each class.

1.3.1 H-40 Class Dimensions

- Hmax = 44 cm (in compliance with 10% tolerance)
  H = 40 cm (Assuming as an example that the humanoid's height is 40 cm)
- 16 cm < L < 24 cm
- 16 cm < AC < 24 cm
- 16 cm < AS < 24 cm
- HD > 4 cm
- Humanoid shall fit within cylinder of 24 cm diameter.
- S < 89 cm^2

1.3.1 H-80 Class Dimensions

- Hmax = 88 cm (in compliance with 10% tolerance)
  H = 80 cm (Assuming as an example that the humanoid's height is 80 cm)
- 32 cm < L < 48 cm
- 32 cm < AC < 48 cm
- 32 cm < AS < 48 cm
- HD > 8 cm
- Humanoid shall fit within cylinder of 48 cm diameter. B
- S < 356 cm^2

1.3.1 H-120 Class Dimensions

- Hmax = 180 cm
- H = 120 cm (Assuming as an example that the humanoid's height is 120 cm)
- 48 cm < L < 72 cm
- 48 cm < AC < 72 cm
- 48 cm < AS < 72 cm
- HD > 12 cm
- Humanoid shall fit within cylinder of 72 cm diameter. B
- S < 800 cm^2

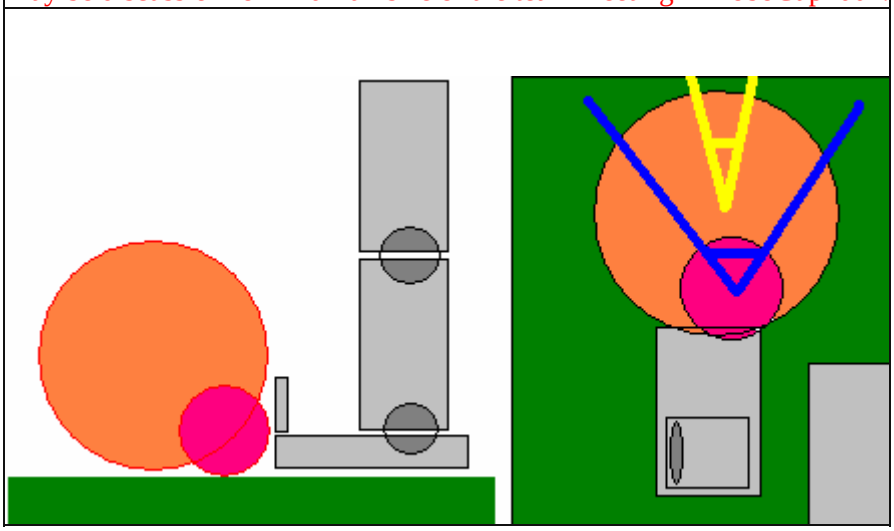
1.4 Ball specifications

The ball specifications for the humanoid competitions are the following:

1.4.1 H-40 Class Ball

- Orange ball 83mm, weight 26 g (same as the 4-legged League).

May be discussion for H40 Ball size of the team meeting in RoboCup2004.



1.4.2 H-80 Class Ball

- Orange ball 83mm, weight 26 g (same as the 4-legged League).

1.4.3 H-120 Class Ball

- Standard FIFA size 5 football, orange color (same as RoboCup Midle Size League)

2. Competitions

2.1 Solo Games

A) Humanoid Walk

Humanoid shall be placed at the designated location in the field. It shall walk along the defined course in the field. It should start from one end of the field, walk to the other end, round the marker placed in the middle of the defense area, and come back to the initial position. Once the game has started, no human assistance shall be allowed to reposition the robot.

Href is the reference height referring to the value in the league name, e.g. 40 cm for H-40.

H is the actual height of the humanoid that is less or equal to **Hmax**

D is the distance from the start line to the marker.
W is the width of the allowed walk area.
MH is the height of the marker.
MR is the radius of the marker.

D = 5 * H
W = 3 * Href
MH = 100 cm
MR = 10 cm

H-40 Class:

D = 200 cm (Assuming as an example that the humanoid's height is 40 cm)
W = 120 cm
MH = 100 cm
MR = 10 cm

H-80 Class:

D = 400 cm (Assuming as an example that the humanoid's height is 80 cm)
W = 240 cm
MH = 100 cm
MR = 10 cm

H-120 Class:

D = 600 cm (Assuming as an example that the humanoid's height is 120 cm)
W = 360 cm
MH = 100 cm
MR = 10 cm

For the first one or two years, the marker could transmit IR. This allows a robot without vision system to perform this task.

The intention of this challenge is to evaluate the stable walking behavior of the humanoid. The course has two straight routes and one 180 degree turn. The 180 degree turn is included in order to evaluate orientation change capability. A minimum visual perception

of the robot is needed, because the marker is red, and there is a yellow panel behind the start/end zone that will help the robot to orient itself.

Fig. E.2 (a). Walk Field.

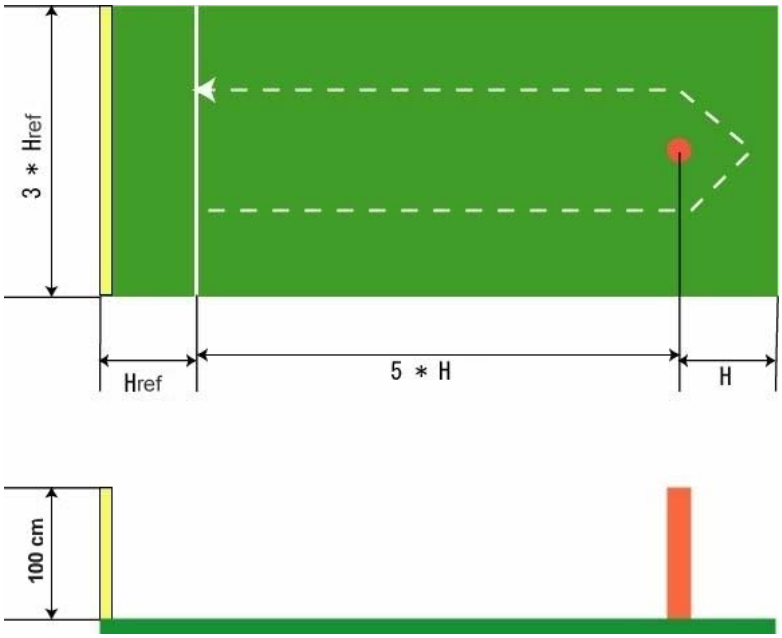
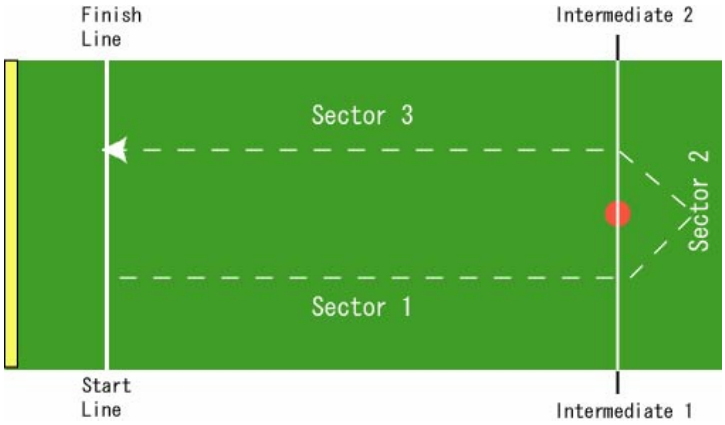


Fig. E.2 (b) Walk field time measurement points

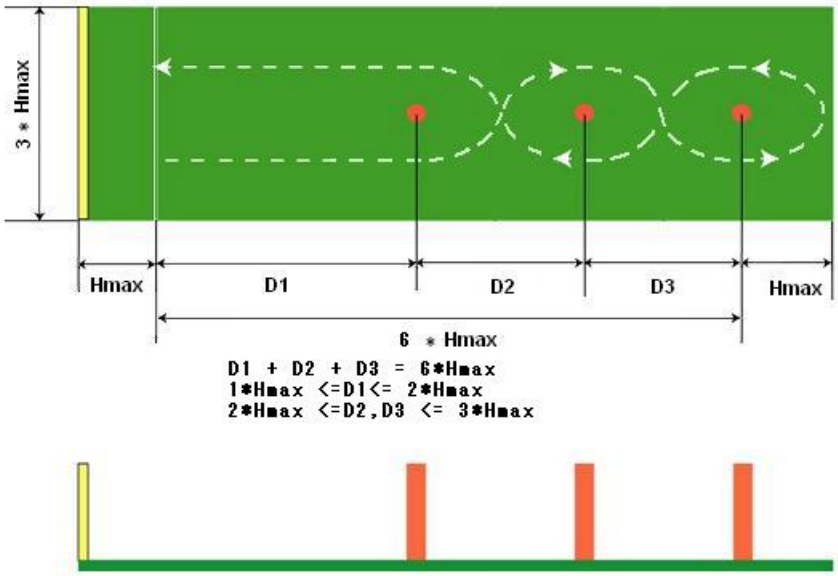


Total time is measured, as well as timing for each one of the sectors. Sector 1 and 3 measures the speed of the robot between the straight lines, and sector 2 measures the duration of the circular movement.

B) Obstacle Walk Challenge

To demonstrate the robot is able to perform obstacle avoidance. An obstacle is the maker pole used for Humanoid Walk. The referee will place 3 obstacles according to the following requirements.

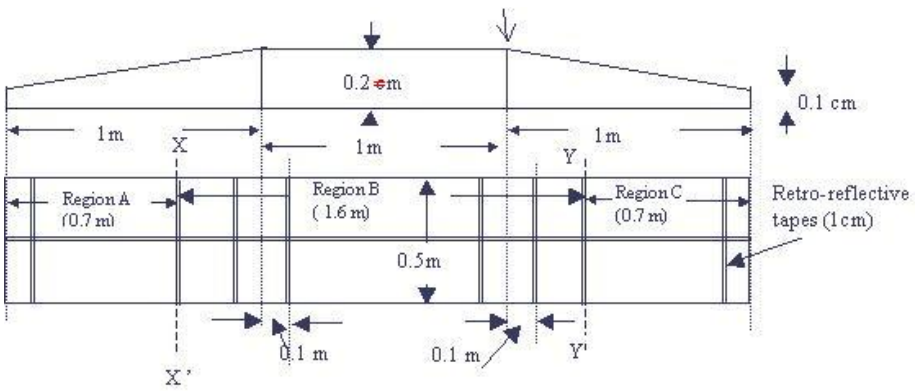
D1 + D2 + D3 = 6*Hmax
1*Hmax <=D1<= 2*Hmax
2*Hmax <=D2,D3 <= 3*Hmax



C) Balancing Challenge

The walking time attack game of this bridge.
Considering that all the robots (from H40 to H120) will share the same platform with threeslopes and each slope is about one meter,
I believe let each robot walk 0.6*Hmax on each slope is reasonable,
e.g. a 150cm tall robot will walk 0.6*150 = 90 cm in each region.
So, I guess we will have to let the robot walk in two trials separately,
one is walking up and another is walking down.

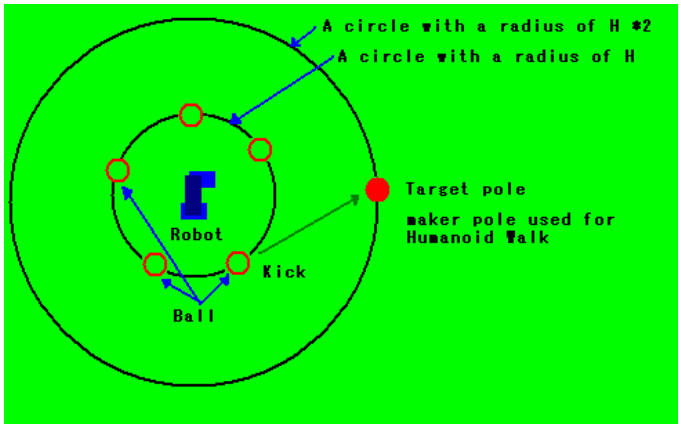
(1) Trial 1: Region A (0.6*Hmax) then Region B (0.6*Hmax)
(2) Trial 2: Region B (0.6*Hmax) then Region C (0.6*Hmax)



D) Passing Challenge

To demonstrate the robot is able to pass the ball deliberately from one to another.

(1) The team set up the robot;
(2) The referee puts the ball in any direction.
(3) The distance of the robot and the ball is the height of the robot.
(4) The target is a the maker pole (which is used for humanoid walk).
(5) The distance between the target and the robot is 2 * height of the robot.



2.2 Games

A) Penalty Shoot-out

Team A's robot is placed behind the ball. Team B's robot is placed in front of the goal. Team A's Robot shall walk and kick the ball to the goal.

D1 is a distance from the initial position of the humanoid to the ball
D2 is a distance from the ball to the goal line.
GW is the width and GH is the height of the goal.

D1 > 0.5 * H
D2 = 3.0 * Href
GW = 3.0 * Href
GH = Href

Goalie robot can be placed within Href from the goal line.

H-40 Class:

D1 > 20 cm (Assuming as an example that the humanoid's height is 40 cm)
D2 = 120 cm
GW = 120 cm

Goalie robot can be placed within 40 cm from the goal line.

H-80 Class:

D1 > 40 cm (Assuming as an example that the humanoid's height is 80 cm)
D2 = 240 cm
GW = 240 cm

Goalie robot can be placed within 80 cm from the goal line.

H-120 Class:

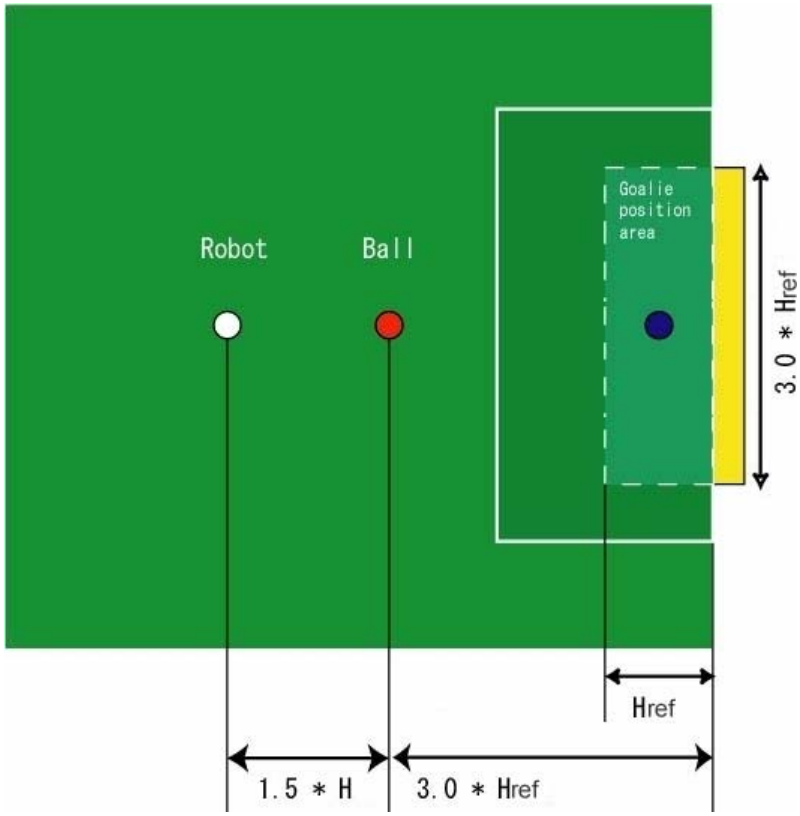
D1 > 60 cm (Assuming as an example that the humanoid's height is 120 cm)
D2 = 360 cm
GW = 360 cm

Goalie robot can be placed within 120 cm from the goal line.

A session will finish, once the goalie robot (Team B) has touched the ball, or, as soon as the ball has stopped within the marked goal field. If the ball is free (not touched by Team B's robot), 60 seconds is allowed for the striker robot to attempt to score the goal. During this period, the session will finish whenever the goalie robot touches the ball. The goalie robot is not alowed to move out of the goalie position area until after 5 seconds after the ball was initially touched by Team A's robot.

One game consists of 5 sessions for each team. If both teams have the same number of scores after 5 sessions, the session will continue until one team scores more goals than the other team. The roles between the teams are exchanged after each kick (e.g. striker and goalie).

Fig. E.4, PK Shoot Field.



2.3 Free Style

Five (5) minutes will be given to each team for them to show any demonstration with their humanoid robot/s. Evaluation will be given by a panel that consists of seven independent jury members. Each jury member shall rate each demonstration, within a scale from 1 to 10 points, for (A) technical merits, and (B) artistic impression. One highest score and one lowest score is discarded, and the total points from the remaining jury are assigned as a over-all score for the team.

2.4 Exhibition(no award)

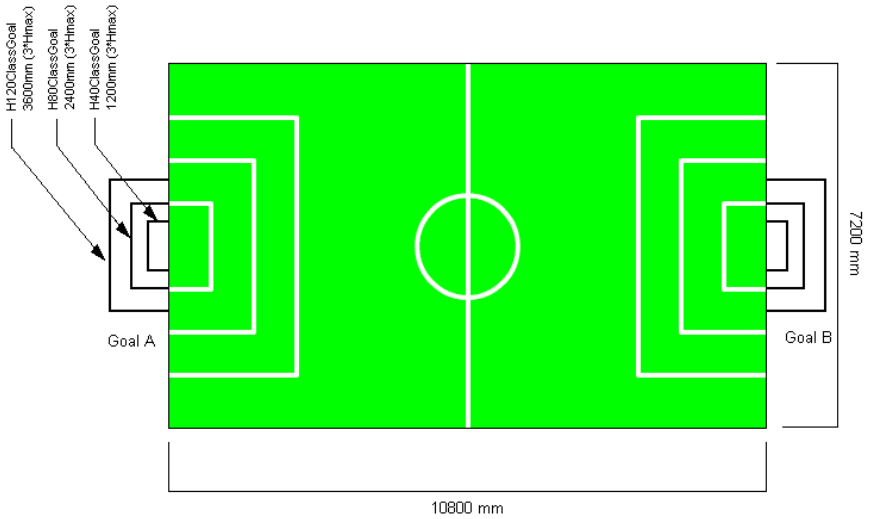
All the robots walk together

3. Environment

3.1 The Field

For the RoboCup 2002 and 2003, the main stage was used as filed for the Humanoid League. As suggested by the organizing committee of the RoboCup 2004, HL will have its own field in 2004.

-The field size is 7.2m by 10.8m
-Green carpet for the MSL will be used.
-The setting will be the same as the filed used in Padova



3.2 Type of robots

Definitely, we should eventually move towards fully-autonmous humanoids.
External power will not be allowed from 2004.

3.3 Lighting Condition

The organizer will simply ensure there is adequate ambient lighting (~500 lux).
Uniformity of the lighting conditions throughout the length of a match will be guaranteed.
This means teams must be prepared for potentially uneven lighting, shadows, and other challenges that arise due to the lack of the traditional spotlights.

3.4 Performance factors

External power will not be allowed.

remote brain	1.5
human remote control	3.0
commercial platform	1.2

4 Awards

(1). "Humanoid walk" classless 1st, 2nd, 3rd

(2). "Penalty Shoot" only 1st for each class

(3). "Technical Challenges (Balancing + Obstacle avoidance + Passing)" only 1st for each class

(4). "Free styles" classless 1st, 2nd, 3rd
This competition is to encourage teams to focus on any basic research issues for the roadmap to 2050 which are not belong to the current Technical Challenges, e.g. throwing ball, catching ball, 1 vs. 1, 2 vs. 2 and so on. We should encourage all the team to look at some basic research issues for the roadmap to 2050.

(5). "The best humanoid" The Best One.

(6). "Exhibition " no award

Roadmap

2004: more challenges in the Free Style competition, e.g., balancing, passing and obstacle walk.
2005: one versus one game, fully autonomous robots.
2006: two versus two game, challenges on multiple objects tracking and collision avoidance.