

The Biorobotics and Locomotion Lab

Under Professor Andy Ruina

Dynamic Balance of a RC Biped Robot

Final Report T&AM 492
Spring/Summer 2007
August 15, 2007

Denise Wong
dw229@cornell.edu

Abstract

The goal of the Radio Controlled (RC) Biped Robot project is to make a small commercial robot balance and walk on small feet; feet that are proportionally similar to human feet and contrary to the usual flat feet that walking robots have. The approach used is dynamic balance, the ability to move while remaining in control and balanced. Gyroscopes and accelerometers are used to detect the position and dynamics of the robot; this data is used to determine where the robot must move its foot and how this movement should take place in order to balance or catch itself from falling. This paper details the motor testing, design and fabrication of a support structure and programming in the C programming language which was carried out as part of this project.

1 Introduction

There are a number of methods that can be adapted to make a walking bipedal robot and a wide range of them exist already. A common approach is to use static balancing to achieve a walking motion, this involves constantly shifting the centre of balance from one foot to another as the robot takes steps and advances and is common in walking robots. The approach that is taken with the RC biped is walking through the means of dynamic balance, remaining in control and balanced whilst moving. This is more similar to the way humans walk. Another way to visualize this is to imagine constantly catching yourself when you walk by taking a step.

Areas involved in achieving this goal that are discussed in this report are motor testing, support system design and programming. Through this project we hope to successfully demonstrate dynamic balancing in robots. Ultimately, this project will increase our knowledge on the kinematics of walking, so we can better understand how to build more efficient robots with better gait.

Figure 1.1 shows the anatomy of the RC Biped Robot. In this paper, the upper part of the robot where the circuit boards are located is referred to as the torso of the robot.

2 Servo Motor Testing

2.1 Introduction

The goal is to calculate the maximum angular velocity of the two different types of servo motors used on the RC Biped robot, HS-475HB and HSR5995TG. HS-475HB is used to control rotation of the hip about the y-axis and also rotation of the torso about the z-axis. HSR5995TG is used to control rotation of the hip about the x-axis and knee rotation (about the x-axis). The maximum velocity data for the motors will be used to calculate the time it will take for any particular movement (or step). Thus enabling the optimum position of the leg required for the robot to balance to be calculated, taking into account

that there is a time delay between the command for a movement and when the robot will finish executing this command.

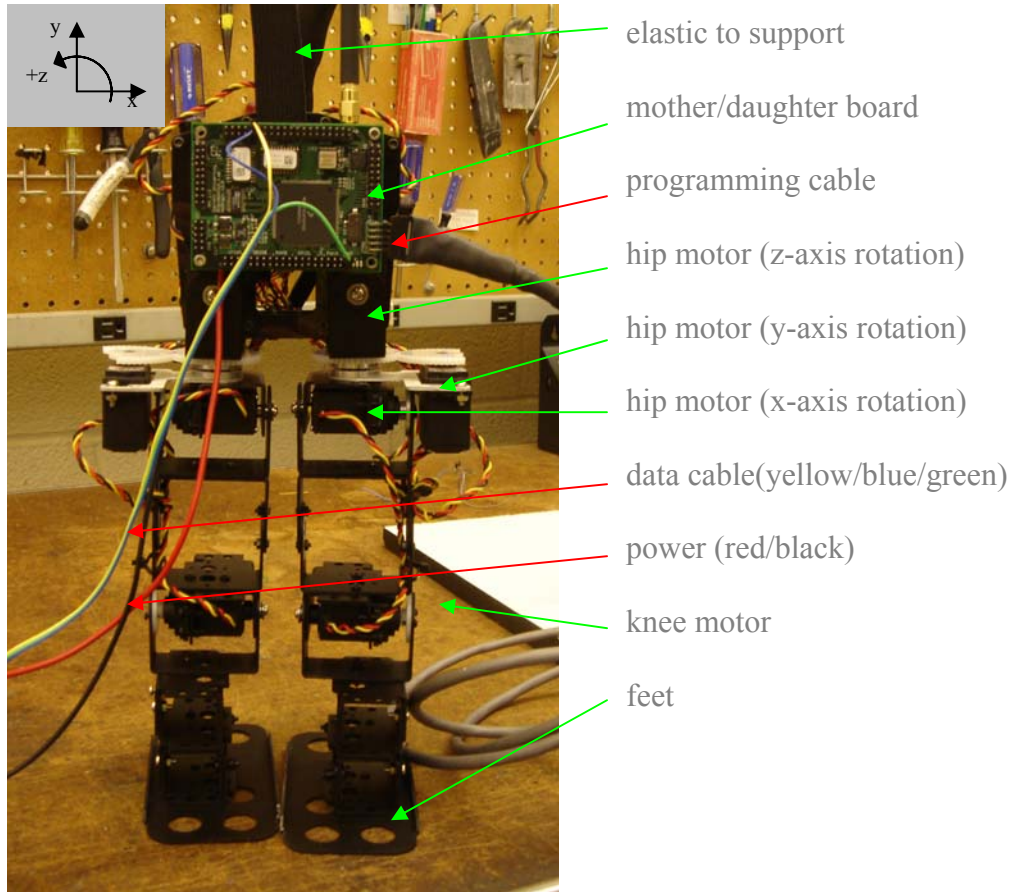


Figure 1.1

2.2 Method

For the HS-475HB motor, the motor was disconnected from the robot and connected to a power supply and the computer. The LabVIEW program was used to send signals to the control the rotation of the motor. A potentiometer was attached to the output shaft of the motor. The potentiometer was calibrated before testing to obtain a relation between the voltage change measured by the potentiometer to the angle the output shaft of the motor rotated, enabling the average angular velocity to be calculated.

The HSR5995TG motor did not respond to LabVIEW commands when connected to the computer, therefore a motor test function was written into the robot code and the code was downloaded onto the robot to rotate the motor 90°. Figure 2.1 is an excerpt of code from the main loop that produced the repeated rotation of the motor output shaft. A

similar approach with the potentiometer connected to the output shaft of the motor and LabVIEW was used to measure the average angular velocity.

```

err_biped();
sensor_read();
fall_state();

++stepcounter;
if (stepcounter>0)
{
speed[m_rhipz]=6;
motor_function(m_rhipz,90);
  if(stepcounter>1000)
  {stepcounter=-1000;}
}
if(stepcounter < 0)
{
speed[m_rhipz]=6;
motor_function(m_rhipz,0); //each cycle 1 millisecond
}

```

Figure 2.2

The response of the motor was collected in LabVIEW in the form of a data table with time and voltage amplitude and plotted on a graph of voltage against time. A signal of varying pulse width from 0.5 to 1.5Hz was used, data was collected every 1 millisecond.

2.3 Results

The graph of voltage against time was used to collect data for the time taken for the rotation and the voltage change which is converted using the calibrated constant of degrees of rotation per volt.

The calculation is as follows:

- ξ conversion constant [rad V⁻¹]
- $\Delta V = V_2 - V_1$ change in voltage amplitude [V]
- $\Delta\theta = \theta_2 - \theta_1$ change in angle [rad]
- T time [s]

Angle change [rad] $\Delta\theta = \xi\Delta V$

Angular velocity [rad s⁻¹] $\omega = \frac{\text{change_in_angle}}{\text{change_in_time}} = \frac{\theta_2 - \theta_1}{T_2 - T_1} = \frac{\Delta\theta}{\Delta T}$

The values are depicted in the graph in Figure 2.1 obtained from LabVIEW of a typical data set.

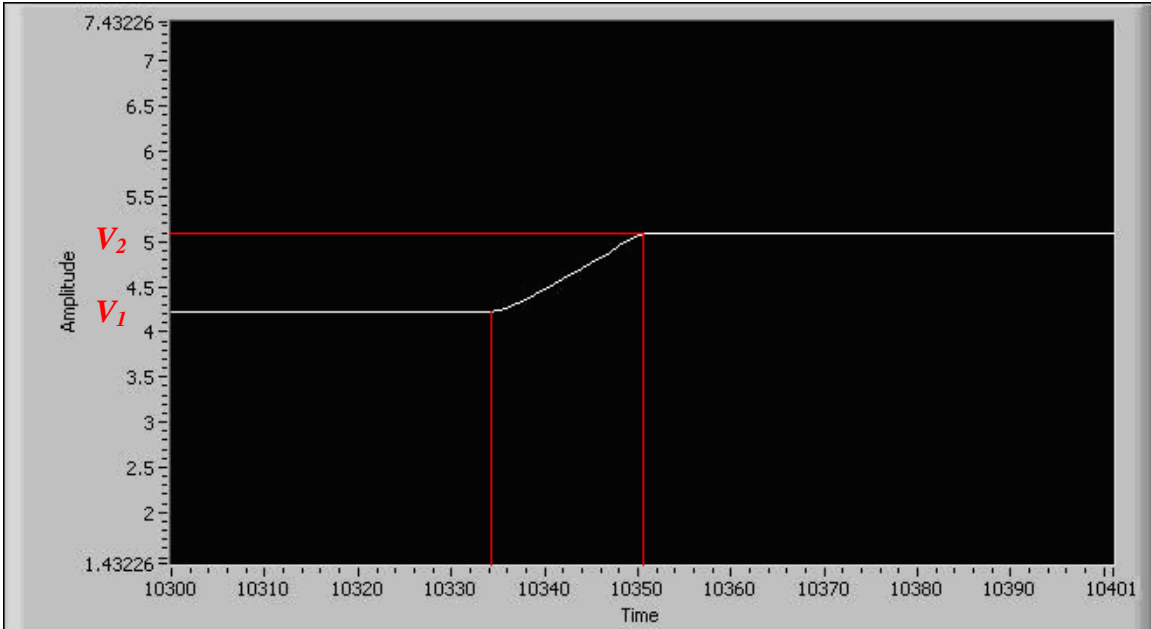


Figure 2.3

T_1 T_2

The following are the angular velocities for each motor. These are the maximum angular velocities of each motor are summarized below:

Motor	Angular Velocity					
	Average		Minimum		Maximum	
	degrees/s	rad/s	degrees/s	rad/s	degrees/s	rad/s
HS-475HB	92.3	1.61	79.8	1.39	98.5	1.72
HSR5995TG	50.7	0.88	46.6	0.81	57.5	1.00

Table 2.1

Extensive primary data tables can be found in Appendix A.

2.4 Discussion

The results represent the maximum velocity of the motors because with when the signal is transmitted to the motors the motors will move as fast as possible to move to that specific orientation. Figure 2.1 shows that while the slope of the graph of voltage amplitude against time is straight, the beginning (at time T_1) and end (at time T_2) of the response has a slightly slower angular velocity than throughout the motion from θ_1 to θ_2 . Therefore, the values obtained are the average angular velocity of the entire response to the command for the motor to move 90° . Over a large angle movement, the effect of the slower start and stop of the motor will be negligible to timing calculations for how long it was take for the robot to take a specific sized step. However, for small angle movements, this variation in angular velocity will result in a larger error in the calculation of the predicted duration of a motor movement or step.

Further motor testing may involve collecting data for time durations for a range of angle movements and comparing them to predicted time values from the average maximum velocity for each motor. This would allow the percentage error to be calculated, which may show that an error correction would be needed to for different ranges of angle movements to allow for a more accurate estimate of the duration it would take to make that movement. Moreover, angular velocity data could be taken for each motor on the robot with the motors connected to the limbs of the robot as this load that will be supported by the motor may affect the angular velocity.

3 Biped Robot Support Structure

3.1 Introduction

A support structure is needed to hold the robot while programs for taking steps and balancing are tested. While testing stepping functions, the robot frequently loses balance or trips over its own feet, causing the robot to fall. This could possibly damage the electronics, particularly the circuit board. It was important to therefore create a support system which would protect the robot from high impact falls. Moreover, the support would also add convenience to the programmer as he/she would no longer need to hold the robot upright in one hand whilst operating the computer in the other.

The following are the needs considered prior to designing the support system.

Needs:

- Stop or slow down the robot's falling motion
 - allow a longer time for program to react and act to the falling
- Prevent the electronics from becoming damaged when falling
- Constrict the movement of the robot to one degree of freedom
 - first step of our approach to dynamic walking/balancing is to focus on balancing in the forward and backward direction and then applying the same method to left and right balancing
- Portable support for the robot
 - allow robot to be taken and demonstrations to take place at poster sessions and conferences

3.2 Methods

Firstly, a morphological chart (Figure 3.1) was created to explore possible ways of approaching this design problem. The needs are listed on the left and on the right are different solutions to the problem or need that is to be fulfilled.

The two clamp stand support structure, from the first row, was selected because of the portability and ease of manufacture. The pulley allows the same support to be provided along the length of the upper bar as the pulley follows the robot wherever it goes. Figure 3.2 is the initial sketch of the support system.

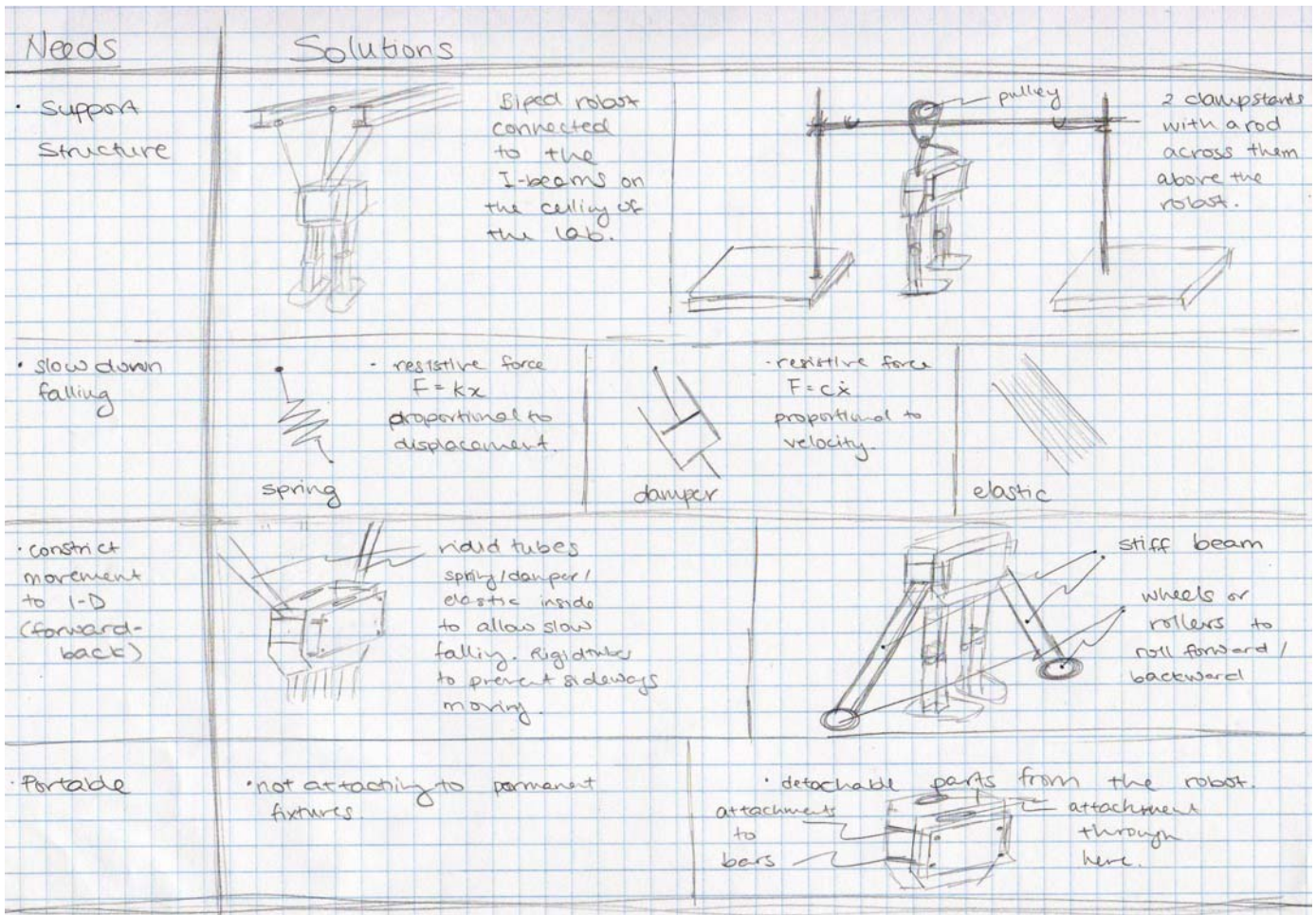


Figure 3.1

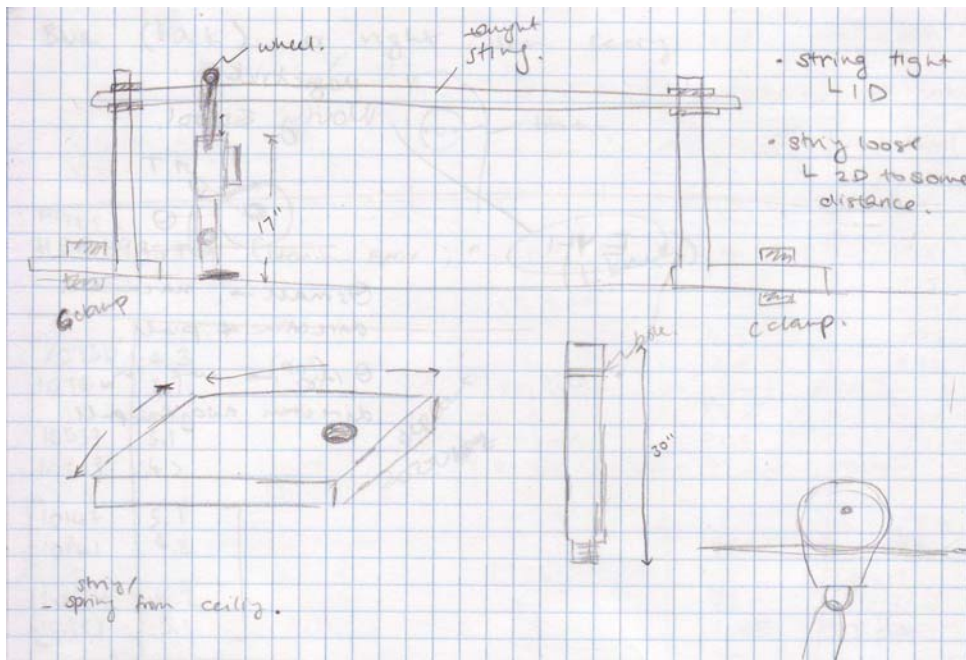


Figure 3.2

3.3 Results



Figure 3.3

Figure 3.3 is a picture of the final design. CAD drawings of the base and pipe flange can be found in Appendix B. Manufactured chipboard was used for the base of the clamps while custom aluminum pipe flanges were machined on the centre lathe and four machine screws were used to attach the pipe flange to the chipboard base. Two purchased clamps were used to hold the support bar up. 4130 high strength steel tubing was selected for the support bar even though it was rather pricey because it is light and stiff – it will not bend in the centre when hung between the two clamp stands (unlike solid and tube aluminum bars.) Two pulleys, instead of one, are used to provide vertical stability because the two pulleys push against each other whereas one pulley will swing forward and backward more. An elastic cord with a large spring constant (low elasticity) is used to connect the torso of the robot to the pulleys so that the robot will fall slowly but not bounce up and down as a result of the spring.

3.4 Discussion

The support worked well in keeping the robot from hitting the floor, thus protecting the electronics, mainly the circuit boards, from damage due to impact with the ground. The elastic used is mainly there to restrict how far the robot can fall down. The alignment of the elastic slows down the forward and backward falling of the robot but does little to

restrict movement in the sideways direction. Further development of the support could involve adding a rigid side support which would prevent sideways movements. Moreover, once the robot is able to stay upright for more steps, this support system may become too small. A longer support bar could be used, but this may bend in the middle leading to inadequate support.

4 Programming

4.1 Introduction

Dynamic walking is not a repetitive action that can be defined and executed as a sequence of exact motor angles. This concept is hard for an individual to conceptualize since after learning how to walk, the task becomes thoughtless and slight changes in the sequence of an individual's walking goes unnoticed but is definitely happening in order to adjust to the terrain. The approach that is used with the RC biped to achieve dynamic balancing is by using gyroscope data which describes the orientation of the robot torso to determine the appropriate foot placement and hence motor movement that would be needed to “catch” the robot from falling.

4.2 Method

Initially, the feet on the robot were narrow two-point feet (Figure 4.1) and the robot was unable to stay upright without support. The lack of ankle stiffness and actuation made it hard to control the position of the ankle and therefore the robot was very unstable in all directions. Larger, flat feet (Figure 4.2) were implemented as a stepping stone towards achieving dynamic balancing on the narrow, 2-point feet. The flat feet eliminate the problem of the flimsy ankles as the feet are fixed at 90° from the lower leg.

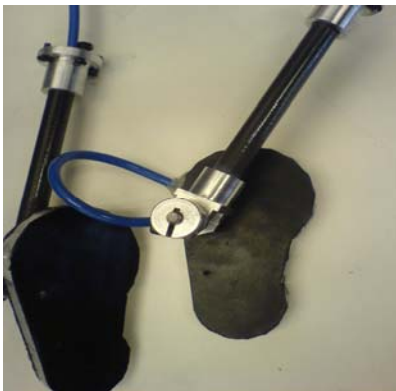


Figure 4.1

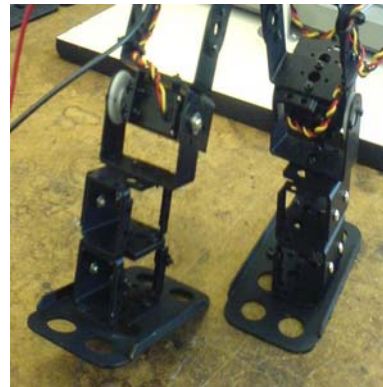


Figure 4.2

When I joined the team, the robot was already built and the backbone of the code to program the robot existed already. The function *pend_step* uses gyroscope data to calculate the angle that the motors need to rotate so the feet are flat on the ground and the

motherboard on the torso is upright. This function is an adaptation from Steven Bagg's pendulum balance (Appendix C) which keeps the torso of the robot upright when the robot is moved (e.g. rotated when holding on to the legs) Figure 4.3 shows the sequence of movements that occur in one fall and step cycle for 1-D starting from a falling state – when the torso is not vertical. First the left hip motor adjusts to move the torso upright, at the same time the knee motor rotates so that the foot is flat on the ground. The right leg then does the same thing in the opposite direction so that the two feet of the robot are flat on the ground and the position of the robot is symmetric about the torso.

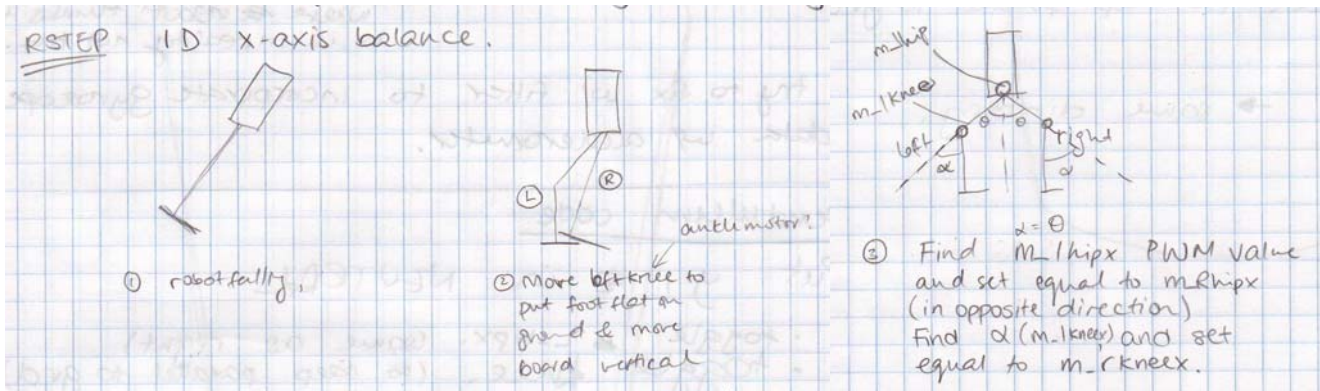


Figure 4.3

4.3 Results

The excerpt of code shown in Figure 4.4 is the *pend_step* function. Gyroscope data is translated from fixed point to a floating point number which corresponds to the PWM change that the motor needs to rotate to counter the current state of the torso so that it will be upright. The function is set up for a 1-D case and only responds to forward and backward movements (about the x-axis.)

```

void pend_step(void)
{
    Uxxi=FFloatRnd2S16int(Uxx);

    LHipx=Lhipxlast+Uxxi;
    RHipx=Rhipxlast+Uxxi;
    RKnee=Rkneelast-Uxxi;
    LKnee=Lkneelast-Uxxi;
    RHipy=NEUTRAL;
    LHipy=NEUTRAL;
    RHipz=NEUTRAL;
    LHipz=NEUTRAL;

    Rhipxlast=RHipx;
    Lhipxlast=LHipx;
    Rkneelast=RKnee;
    Lkneelast=LKnee;

    RHipz=RHipz-50;
    LHipz=LHipz+50;

    setPWMval();
}

```

Figure 4.4

The code is tested by holding the leg just under the hip and tilting the robot slightly forward or backward. The function works well for the first two or three tilts, but as more time passes the accuracy of the rotation worsens and eventually the PWM values of the robot are set to the extreme positions. The gyroscope only collects information discretely and integration is used to complete the data, therefore causing a drift in the gyroscope data with respect to the actual location of the robot. (Noise is another factor that contributes to this error.) It was difficult to deduce the error as variables in the program were not recorded, therefore it was unclear what the problem was and how it could be eliminated. To overcome this problem, LabVIEW¹ was implemented by connecting the motherboard to the computer so that data could be sent via wires to track global variables. The LabVIEW program used to keep track of the parameters of the Ranger² robot was used to track the variable on the RC Biped robot.

4.4 Discussion

The RC Biped robot models a human leg with a ball and socket hip joint that allows for three degrees of freedom and a hinge joint at the knee which allows for one degree of freedom. The idea seems feasible and tests revealed that the motors at least initially responded well and as expected to a tilt in the torso. However, the sequence of motions is not an accurate representation of the way humans balance. Firstly, the RC Biped has a reverse knee, meaning that it bends both forward and backward from the neutral straight position; contrary to the human knee which only bends backwards. Secondly, the robot does not have ankle actuation. The human leg from the hip has 7 degrees of freedom whereas the RC biped robot has only 4. Moreover, the sequence does not mimic that stages that a human is likely to go through when falling.

Further development of a balancing function could implement the accelerometer data in addition to the gyroscope data so that the two sets of data can be collaborated to obtain more accurate calculations of the orientation of the robot. Furthermore, implementation of a filter to eliminate the drift in the gyroscope data would also enable more accurate calculations of the position of the robot. In addition, an error state could be incorporated to correct the drifting gyroscope data.

Moreover, ankle actuation would also be useful for balancing with smaller feet to make finer adjustments to stay balanced. In the future, it would also be helpful to be able to track the parameters as well as the global variable in the program. In this case, it would have been very beneficial to be able to track the LHipx, RHipx etc. values.

5 Conclusion

Motor testing shows that the angular velocity of the two types of servo motors for 90° rotation is 92.3°/s for the HS-475HB model and 50.7°/s for the HSR5995TG model.

¹ Same LabVIEW programme as that used for the Ranger Robot - written by Jason Cortell

² Cornell University Biorobotics and Locomotion Lab, Ranger Walking Robot project

These values can be used to approximate the duration of a particular step and hence used in calculations to compute the optimum step size and foot placement. A simple, portable, robot support system was designed and fabricated to allow the robot to be restricted from falling and protected from damage during testing. The support system allows for free, protected movement in the z-direction (forwards and backwards) and restricted movement in the x-axis (left and right.) Ideally, only one dimensional movement would be allowed. A stepping function, `pend_step`, developed from Steve Bagg's pendulum balance (Appendix C), uses gyroscope data to determine how much each motor must move in order to keep the torso of the robot upright and the feet flat on the ground. The current feet could be cut down so the robot's proportions are more similar to the proportions of the human, this would also mean that the robot is less balanced with not moving which increases the need for a balancing algorithm. The ultimate goal would be to balance and walk on narrow two point feet which are of similar proportions to the robotic leg as the human leg is to the feet.

Acknowledgements

I would like to thank my talented peers from the Biorobotics and Locomotion Lab, Spring 2007, for creating the positive lab environment. Special thanks to Steve Bagg for starting this project and for his patience in guiding me through all the code; and to Alex Veach for being a great teammate and of course to Jason Cortell for always being there to answer my endless questions.

References

- [1] Barr, M. and Massa, A. 2006. *Programming embedded systems: with C and GNU development tools*. Sebastopol, Calif. : O'Reilly
- [2] Senior, A and Tosunoglu, S. 2006. *Design of a Biped Robot*. 2006 Florida Conference on Recent Advances in Robotics