Inner / Outer Leg Symmetry Of the Cornell Ranger

Semester Report – Fall 2007
MAE 690 - Master's of Engineering (M. Eng.) Project

Rohit Hippalgaonkar
rrh54@cornell.edu
+1-607-379-3529
3-4-1013/15, Barkatpura
Hyderabad, A.P., INDIA - 500027

Master's of Engineering Candidate, May 2008
Sibley School of Mechanical & Aerospace Engineering
Cornell University
Ithaca, NY, USA 14853

Bio-robotics and Locomotion Lab
Department of Theoretical & Applied Mechanics
Cornell University
Ithaca, NY, USA 14853

Foreword

This report outlines the work done by Max Wasserman (MAE '09, Cornell University) and myself in the Bio-robotics and Locomotion lab, while solving the problem of equalizing the mass properties of the two legs on the Cornell Ranger – a dynamic-walking bi-ped robot. This equalization will need to be done every time a major change is made to the mass properties of the robot and hence this report will also serve as a guideline on how to do this.

Chapter 1 introduces the reader to the problem at hand while also outlining relevant design details of the robot. Chapter 2 states the conditions on the mass properties that need to be equalized while Chapter 3 outlines the apparatus, experiments and calculations used to measure relevant properties of the robot. Chapter 4 gives an overview of the MATLAB programs used to achieve symmetry between inner and outer legs. Conclusions are in Chapter 5.

We worked approximately ten hours a week on the project – starting with planning and designing an apparatus for measuring the relevant mass properties of the robot, taking measurements, writing the required programs for the robot, keeping track of the changes made to the robot (in terms of these mass properties) and finally recommending a mass distribution for the robot in its current configuration that will equalize the mass properties of the two legs.

Contents

1. Introduction

The Cornell Ranger unofficially holds the world record for the largest distance traveled by a walking robot or walking toy without help, falling over or having to recharge its batteries. It can be aptly described as a 'four-legged bi-ped', having two inner legs that are joined at the bottom by a short steel bar, and two outer legs on the outside that are joined at the top by a long aluminum bar (Fig. 1). One description of the walking 'cycle' of the Ranger is as follows – First, say, the inner leg is the pivot or stance leg, while the outer leg is swung forward, by the hip-motor and a push-off mechanism on the outer leg (actuated by the foot motor on the outer leg). After the foot on the outer leg lands, it is now the stance leg and the inner leg is swung. Thus each leg swings once every cycle.

One area where there is room for improvement is the 'symmetry' of the walk - that is, currently, the size of the steps are not equal for the two 'half-cycles', on applying the same input (i.e. identical torque histories to hip motor in both half cycles, and same torque histories to individual feet motors).

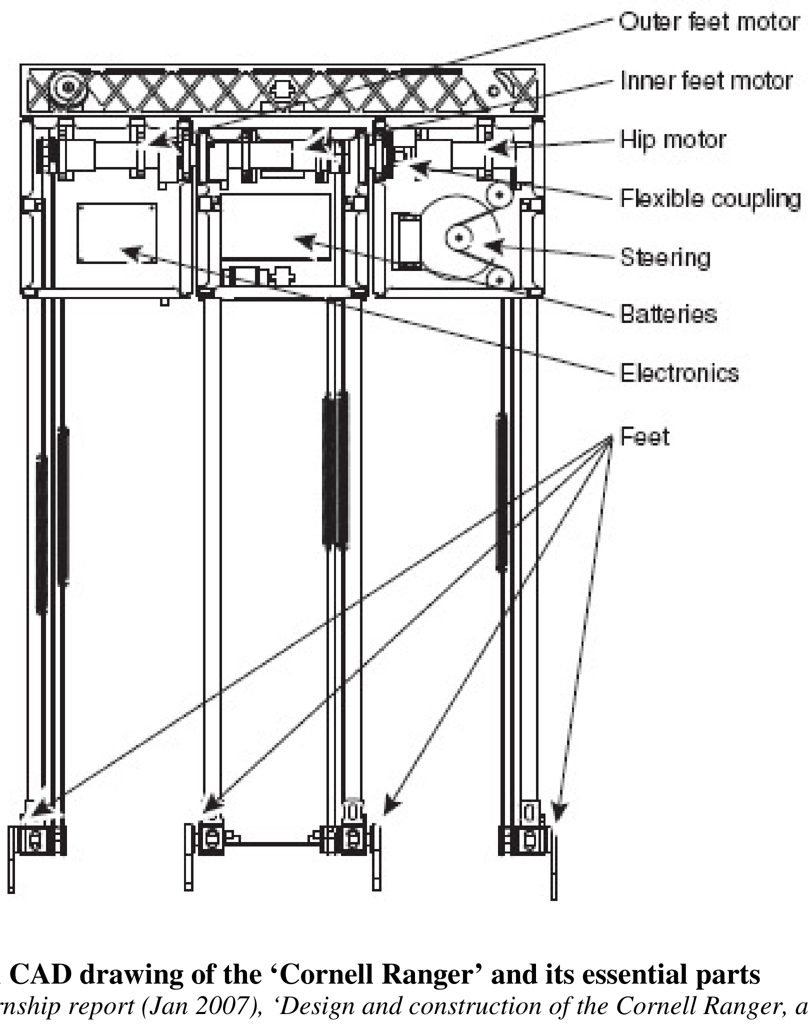


Fig. 1.1 – A CAD drawing of the 'Cornell Ranger' and its essential parts
(from Daniel Karssen's internship report (Jan 2007), 'Design and construction of the Cornell Ranger, a world record distance walking robot')

2. Conditions for Symmetry

Given that we apply the same control law and inputs to both feet motors, and the same control law and the same inputs to the hip motor in both half-cycles, to get a 'symmetric walk' or identical motion during both half-cycles, the two legs must be dynamically similar.

For the legs to be dynamically similar:

(1). the governing equations of motion of the legs must be the same in the two half-cycles, and

(2). additionally, all couplings between the two legs must be equivalent, for e.g. the springs joining the foot motor to the foot on each leg must have equal spring constants.

Condition (1) requires that the equations 2.1 – 2.3 (below) need to hold. Our project goals were to satisfy these conditions and to get a procedure in place so that in future, if changes to the robot are made, anybody can use this procedure to get the robot back into a symmetric configuration.

$I_{\mathrm{in/h}} = I_{\mathrm{out/h}}$ ...... Eqn. 2.1

$m_{\mathrm{in}} y_{\mathrm{in}} = m_{\mathrm{out}} y_{\mathrm{out}}$ ….. Eqn. 2.2

$m_{\mathrm{in}}\, z_{\mathrm{in}} = m_{\mathrm{out}}\, z_{\mathrm{out}}$ .... Eqn. 2.3

where,

y_{in} (y_{out}) = distance of the centre of mass of the inner leg (outer leg) from the hip-axis, as measured along the leg.

z_{in} (z_{out}) = distance of the centre of mass of the inner leg (outer leg) from the hip-axis as measured perpendicular to the plane of the leg.

m_{in} (m_{out}) = mass of the inner (outer) leg.

$I_{\mathrm{in/h}}$ ($I_{\mathrm{out/h}}$) = moment of inertia of the inner (outer) leg about the hip-axis.

Note that, perhaps surprisingly, we do not need $m_{\mathrm{in}} = m_{\mathrm{out}}$, nor do we need $I_{\mathrm{in/h}} = I_{\mathrm{out/h}}$. In particular adding *any* point mass to either leg at the hip does not affect dynamic similarity.

3. Measurements

It is not a co-incidence that the quantities that need to be equalized for dynamic similarity (ref. Eqns. 2.1 – 2.3) of the two legs are all measurable without disassembly of the robot. For example, m_{in} & m_{out} and $I_{\mathrm{in/com}}$ & $I_{\mathrm{out/com}}$ cannot be measured without disassembly. Experiments were devised to help us measure the relevant mass properties for each leg. Say that we want to calculate these quantities for the outer leg - the recipe for calculation of the relevant quantities (Eqns. 2.1 – 2.3) for each leg is as follows.

- First, the inner leg is clamped and the outer leg is left free to swing (**Setup I,** Fig. 3.1) and the deflection of the outer leg from the vertical is noted (θ_0) at equilibrium.

- Next, the outer leg is swung and the average time-period (and hence, ω) is noted over ten cycles of forced, damped, small-angle oscillations where the amplitude of oscillation is kept constant. This can be done in two ways – a) We could try the 'zero-torque method' wherein the hip motor is engaged to the outer leg and the gains on the hip-motor controller are tweaked such that we get approximately stable motion that does not grow or decay much. This means that the motor torque is roughly equal and opposite to the friction torque – Fig. 3.2 has graphs of hip-encoder output vs. time in a typical case. b) In the second method, where the motor is disengaged, the pendulum is left from a position slightly displaced from equilibrium and is given constant periodic forcing such that we get constant-amplitude small-angle oscillations – if the leg is left to swing naturally it will come to complete rest in about five oscillations.

- We next compute the quantity $m_{\mathrm{out}}r_{\mathrm{out}}$, where r_{out} is the distance of the centre of mass of the outer leg from the hip-axis. The procedure to find $m_{\mathrm{out}}r_{\mathrm{out}}$ is explained under **Set-up II** (Fig. 3.3) – the same set-up allows us to take measurements from which we calculate $m_{\mathrm{out}}y_{\mathrm{out}}$ and $m_{\mathrm{out}}\,z_{\mathrm{out}}$.

- Once we know the values of ω and $m_{\mathrm{out}}r_{\mathrm{out}}$ we calculate the value of $I_{\mathrm{out/h}}$ using Eqn. 3.2.

$$I_{\mathrm{out/h}}\alpha(t) = - m_{\mathrm{out}}r_{\mathrm{out}}g\sin(\theta(t)) = -m_{\mathrm{out}}r_{\mathrm{out}}g\theta(t) \qquad \text{…. Eqn (3.1)}$$

where, $\alpha(t)$ = angular acceleration
$\theta(t)$ = angular deflection from equilibrium position

On solving Eqn. 3.1, we get, $\omega = (m_{\mathrm{out}}r_{\mathrm{out}}g/I)^{(1/2)} = 2\pi/T_u$ …. Eqn (3.2)

We approximate the dynamics of swinging motion to that of an un-damped, unforced simple pendulum performing small-angle oscillations under gravity. Appendix 1 outlines a procedure for error analysis, where we tried to estimate the error in our ω calculations due to the above approximation and found that the approximations were indeed reasonable. A similar procedure is followed in the computation of the relevant mass properties of the inner leg, and here the experimental set-ups are much simpler.

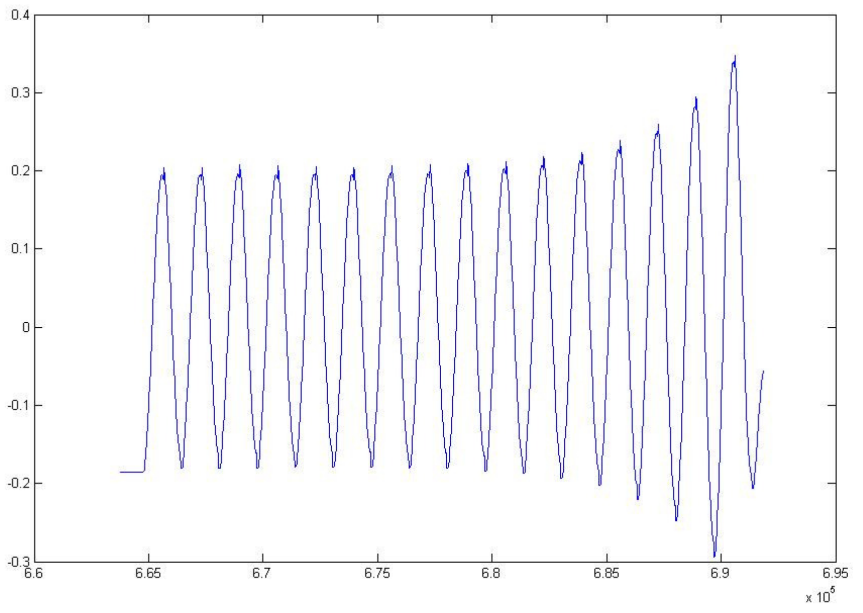


Fig. 3.1 – Typical position data from hip-encoder (‘Zero-torque’ experiment)

<u>Set-up (I) - For measurement of moments of inertia (I) about the hip</u>

Outer leg inertia measurement set-up

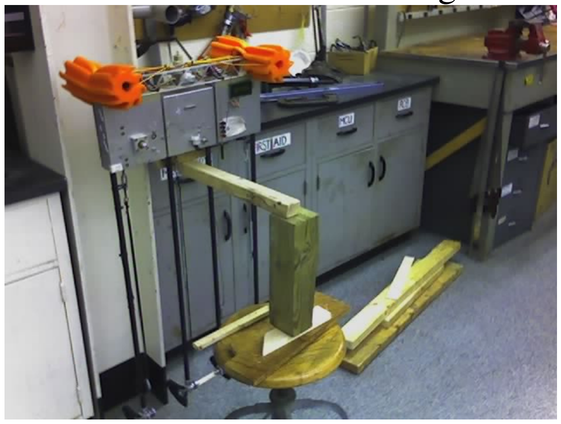


Fig. 3.2 (a) – Front view

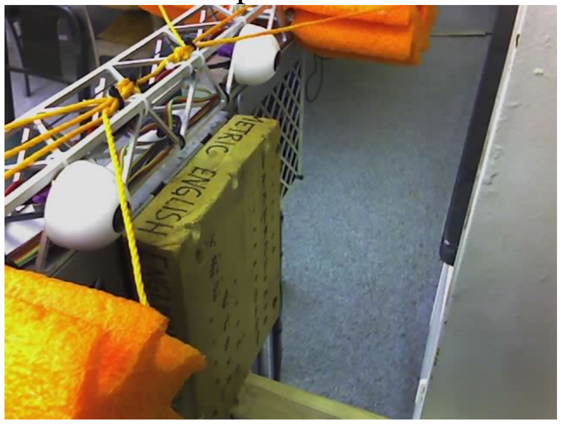
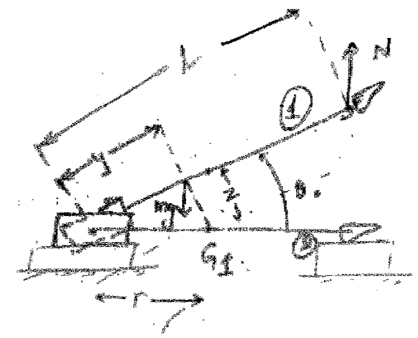


Fig. 3.2 (b) - Rear View

Figs. 3.2 (a) and (b) show front and back views of the set-up used to measure outer leg inertia. Not seen here is that the long bar is clamped at the back to the wall. Also if the long bar is clamped properly and is perfectly horizontal (unlike as in Fig. 3.2 (a)), we do not need to support it at its front end. The set-up for measuring moment of inertia of the inner leg about the hip axis is a lot simpler, wherein we just screw a large piece of wood onto one of the outer legs and clamp this to the wall. A similar procedure is followed to obtain measurements.

Set-up (II) - For measurement of my, mz values.

To measure my, mz



G_1 : centre of mass location of leg 1.

$(mg)y\cos\theta_0 = N\cdot(l\cos\theta_0)$

$(mr) = (my)/\cos\theta_0$.

$mz = (my)\tan\theta_0$.

θ_0 = equilibrium deflection (from vertical).

We know (N/g) from the digital scale reading

Fig. 3.3 – To measure my, mz values

Fig. 3.3 shows not only the set-up that allows eventual computation of $m_{out}y_{out}$ and $m_{out}z_{out}$ but also outlines a method that will allow for precise and fast computation of these values. We horizontally rest both ends of the inner leg on a set of 2-by-4's. The outer leg must be free to swing about the hip-axis, and using the value of θ_0 calculated from **Set-up I** we lift and hold the inner leg such that its centre of mass (G_1) lies on the horizontal passing through the hip axis. A force-measure is placed at a convenient place on the outer leg (say, at 'l' from the hip axis) and we compute $m_{out}y_{out}$, $m_{out}z_{out}$ values from the procedure shown in Fig. 3.3. Note that we can also compute mr from the experiment above, where r is the distance of the centre of mass from the hip-axis.

Also note that for the inner leg, $\theta_{0/in}$ was found to be 3.9 degrees and for the outer leg, θ_0 is approximately 0.7 degrees. Now, cos(3.9) = 0.9976, and hence, except for calculation of the values of $m_{out}\,z_{out}$ and $m_{in}z_{in}$, we can approximate $\theta_{0/out} = \theta_{0/in} = 0$ degrees for all practical purposes. Hence, for example, we need not bother lifting and holding the outer leg by exactly $\theta_{0/out}$ as shown in Fig. 3.3.

4. Equalizing Mass Properties

The robot properties were measured on 23rd October, 2007 and showed the following values **(Table 4.1)**:

	Inner Leg	Difference	Outer Leg
my (kg.cm)	53.76	- 1.60	55.36
mz (kg.cm)	4.17	3.53	0.64
$I_{/h}$ (kg.cm^2)	2480	-1606	4086

Table 4.1

Since then the following changes were made to the robot:

- Three boxes (~ 100gms each) have been added– one below each of the 3 steel cages of the robot, i.e., about 22 cm from the hip-axis. These house the satellite processors.
- A new, stiffer bar (33gms) joining inner legs has replaced the previous one (about 20gms), at around 95 cm from the hip.
- Hip-spring assembly moved down by 31cm (initially the top-ends of the higher pair of springs were at 23 cm, now they are at about 54 cm).
- Small electronics parts added inside the steel cages.
- An inertial measurement unit (IMU) was added below the computer box on the inner leg, which weighed about 60gms, and would be roughly located at 25 cm from the hip.
- Two identical pairs of hip-springs are being used currently – each spring of about 16gms each, and rest length 12cm. Earlier, we had only one of these pairs of springs along with a heavier pair of springs (45 gms each, rest length of 16cm and placed below the lighter springs). These heavier springs also had much higher pre-tension that the lighter springs.
- A small part was removed from the bottom of the outer leg (35gms, at about 87 cm from the hip axis).

The following values were obtained after the changes when the measurements were taken using the zero-torque experiment on December 5th 2007 **(Table 4.2)** – note that the $I_{/h}$ and *my* values of the inner leg have now increased above the values of the outer leg, as most of the changes were made to the inner leg.

	Inner Leg	Difference	Outer Leg
my (kg.cm)	70.3	4.14	65.86
mz (kg.cm)	4.17	3.51	0.66
$I_{/h}$ (kg.cm^2)	4584.5	279.8	4304.7

Table 4.2

So starting with the values in **Table 4.2** we must move to a symmetric configuration with the following constraints in mind:

- $I_{in/h}$ ($>I_{out/h}$) must not increase further by much (preferably should decrease). If $I_{in/h}$ increases by too much we would have to increase $I_{out/h}$ as well in order to match them up. In turn, cycle-time would increase and the robot would walk a lesser distance, as batteries would get exhausted much faster.
- The new mass distribution should be relatively easy to implement (e.g. adding masses too far out from the plane of the robot will be difficult; neither must we add masses above the hip axis).
- We must accommodate specific needs (e.g. another battery must be added and so on).
- Must not affect other projects on robot (e.g. hip-springs must still be able to give desired torque (6N.m) in new configuration as well).

Two MATLAB programs were used to in solving the problem of getting to a symmetrical configuration.
1) A program (equality.m) to which the user inputs the locations of the masses to be added (must be three masses or more) and the differences in the relevant mass properties of the two legs is made exactly zero by the program (code in Appendix 2(a)). This program however has limited applicability – every time a change is made to the robot one must calculate manually the changes in the mass properties. Also, it fails to work with the constraints as described above and does not give the user an intuitive feel for solving the problem.

2) Another program (equality_auto.m, Appendix 2(b)) overcomes both the disadvantages of the previous one and is also a guided-user interface (GUI). Essentially it requires the last set of properties as initial input, and prompts the user to add a mass wherever required. It then calculates the new differences (and individual values) in the mass properties while prompting if any further mass needs to be added. A sample session of this program is included in Appendix 2(c) – we have gone from the asymmetric configuration in **Table 4.2** to a symmetric configuration.

In both these programs any masses that was added was modeled as a positive point mass being added at the concerned location. To move a particular mass a negative mass is to be added by the user at the old location and an equal positive point mass is to be added at the new location. The program would then be calculate the new mass properties. For instance when the mount holding the hip-springs was down by 31 cm on the inner leg, this change accounted for in the following way by the program:

$$(m_{in}y_{in})_{new} = (m_{in}y_{in})_{old} + m_{mount}y_{mount,f} + (-m_{mount})y_{mount,i}$$

where $y_{mount,f}$ is the final y co-ordinate of the mount and $y_{mount,i}$ is the initial position of the mount. Similarly,

$$(m_{in}z_{in})_{new} = (m_{in}z_{in})_{old} + m_{mount}z_{mount,f} + (-m_{mount})z_{mount,I},$$ and

$$(I_{in/h})_{new} = (I_{in/h})_{old} + m_{mount}[(y_{mount,f})^2 + (z_{mount,f})^2] + (-m_{mount})[(y_{mount,i})^2 + (z_{mount,i})^2]$$

To minimize the values in column 2 of **Table 4.2,** it is found using the GUI that we need to place the second battery (split into two equal halves of about 375 gms each) at y = 7 cm, z = 4.5 cm from the hip-axis on the outer legs and move the whole hip-spring assembly up by about 10 cm from its current configuration.

	Inner Leg	Difference	Outer Leg
my (kg.cm)	68.77	0.12	68.65
mz (kg.cm)	4.17	0.155	4.035
$I_{/h}$ (kg.cm^2)	4384.16	27.52	4356.64

Table 4.3

The differences in the mass properties that we are interested in are small percentages of the individual values.

5. Conclusions

Clearly, through the various changes made to the robot since measurements made on October 23rd 2007, the robot is closer to symmetry. By the addition of an extra battery near the hip-axis on the outer legs and movement of we were really able to drive the differences down.

Through the course of this project we also learnt that changes which are seemingly negligible, can make a large difference in the mass properties. For instance, the hip spring assembly is relatively light (overall weight is less than 200 gms), but since it is located far away from the axis and was moved down by a large distance (31 cm), it caused a large change in the inertia values of the inner leg.

To solve this problem next time, we also have a convenient MATLAB program (equality_auto.m) which is essentially a GUI and helps the user intuitively decide what masses to add and move so as to obtain a more symmetric configuration.

What remains is to implement the mass-distribution recommendation for symmetry and verify that the robot actually has identical half-cycles with the extra battery added and the hip springs moved up. When all the changes listed below **Table 4.1** were being made to the robot, they were tracked by extrapolations made using equality_auto.m instead of measuring properties every time a change was made. The recommendations based on such extrapolations were very similar to the recommendations made based on measured values (**Table 4.2**) and hence one expects that when the changes recommended are actually implemented we will get a configuration that is reasonably close to what is predicted in **Table 4.3.**

Appendix 1.a)

When the outer legs were swung without any forcing (with motor disengaged), they came to complete rest in just about five cycles – this complete stopping suggested the presence of solid friction. We want to get a sense of the approximation made when the swing motion in **Setup I,** was equated to un-damped, unforced motion of a simple pendulum under gravity. The actual motion was under the action of a gentle, periodic force which made sure that the damping in the system is balanced and the motion was at constant amplitude.

Error Analysis

My approach was to compare **damped, unforced motion** to **un-damped, unforced motion** and compare the frequencies in both cases. If the difference in these frequencies is negligible compared to the un-damped frequency, then it would be reasonable to assume that the difference in frequency of the **actual motion** (damped and forced) and the **un-damped, unforced motion** will also be negligible compared to the un-damped frequency. Thus we could just use the frequency of the actual motion, as calculated from Eqn. (3.2) – which applied for un-damped motion - and calculate inertia from this. Also, if this approximation was a reasonable one for the outer leg, then it would be also be reasonable for the inner leg since it was observed that the inner leg only came to a stop in around 10 cycles when left to oscillate naturally as compared to 5 cycles for the outer leg.

The idea was to simulate and compare damped, unforced and un-damped, unforced motions in MATLAB and thus we needed to have are the same set of values of mr and I to compare the two frequencies in the damped, unforced and un-damped, unforced cases. For want of another way of doing it, we use the $m_{\text{out}}r_{\text{out}}$ as calculated from **Set-up II**, and use this $m_{\text{out}}r_{\text{out}}$ value and the experimentally observed ω value to compute the $I_{\text{out/h}}$ value we need for comparison (again from Eqn (3.2)). The whole procedure could be thought of as having some physical object which we know has the corresponding mr and $I_{/\text{h}}$ values and we're comparing its natural oscillations, with and without damping.

The following equations of motion describe the damped, unforced case:

$$I_{\text{out/h}}\alpha(t) = - c*sign(d(\theta(t))/dt) - m_{\text{out}}r_{\text{out}}g\theta(t) \qquad \text{…. Eqn (6.1)}$$

The code for the simulations is in Appendix 1 b). MATLAB has trouble solving the above differential equation, basically due to the discontinuity of the signum function at '0'. The logistic function (Eqns. 6.2, 6.3) was used to get roughly the same behavior for friction but with a continuous nature at '0'. The value of c was tuned until the motion came to a stop in about five cycles – as observed.

$$I_{\text{out/h}}\alpha(t) = - c(1 - y)/(1 + y) - m_{\text{out}}r_{\text{out}}g\theta(t) \qquad \text{…. Eqn (6.2)}$$

$$y = exp(-\varepsilon d(\theta(t))/dt),\ \varepsilon = 100. \qquad \text{…. Eqn (6.3)}$$

The plot for friction torque in the two cases is shown in Fig. 6.1.

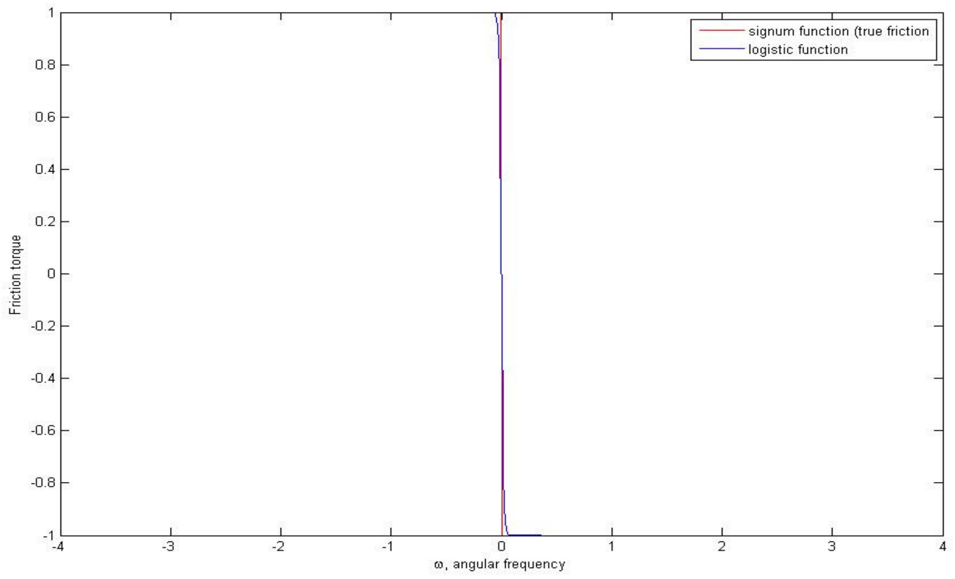
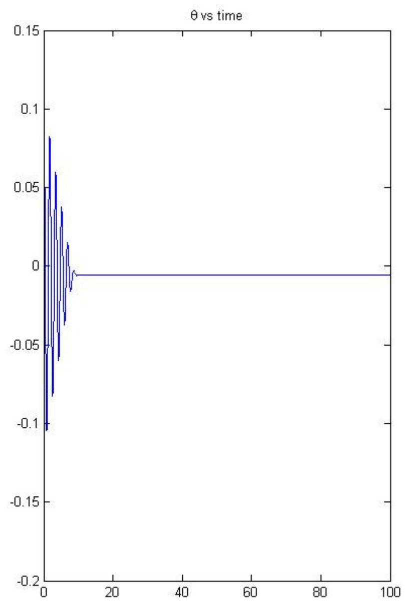


Fig. 6.1 – Solid friction Torque vs. Angular Velocity



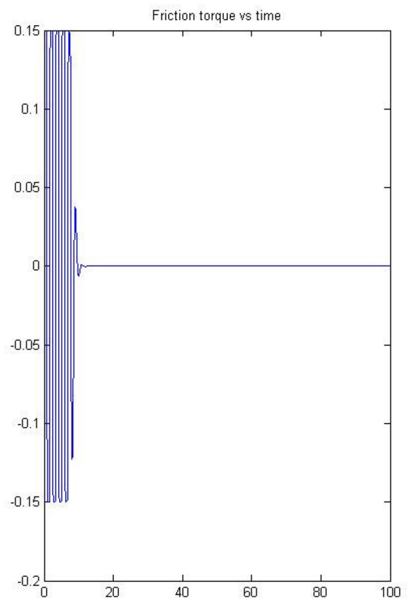


Fig. 6.2.1 – θ vs t **Fig. 6.2.2 – f vs t**

Figs. 6.2.1 and 6.2.2 show the results of the simulation with c = 0.15 – the pendulum comes to rest in about 5 cycles and has about the same frequency as the un-damped case. It was found that:

T_u - T_d < 0.01sec < .6% of T_u, Where **T_u = 1.721 seconds** (from Eqn. 3.2).

Hence our approximation to find $I_{/\mathrm{hip}}$ (from Eqn. 3.2) seems like a reasonable one to make.

Appendix 1.b)

This appendix has the MATLAB code that simulates the motion of a forced, damped pendulum where only solid friction is assumed to be the dissipating torque and the solid friction term is approximated by the logistic function.

```
function pend_coul_frixn
% Author : Rohit Hippalgaonkar - 10/29/2007
clc;

% parameters - all in MKS units
mr = 0.6125; % m - mass, r - distance of COM from hip-axis
g = 9.81;
I = 0.4503; % inertia about the hip axis
c = 0.15; % co-efficient of solid friction
eps1 = 100;

% initial conditions on position
theta0 = pi/30;
% initial conditions on velocity
theta0dot = 0;
zzero = [theta0 theta0dot];

tspan = linspace(0, 100, 100000);
tol = 1e-6;
options = odeset('reltol',tol,'abstol',tol);
[t zarray] = ode45(@rhs, tspan, zzero, options, mr, g, I, c);

% Unpacking the solution
theta = zarray(:,1);
omega = zarray(:,2);

r = - c*(1 - exp(-eps1*omega))./(1 + exp(-eps1*omega));

subplot(121)
plot(t, theta);
title('\theta vs time');

subplot(122)
plot(t, r)
title('Friction torque vs time');

end

% RHS file in ode45 (ode solver)
function zdot = rhs(t, z, mr, g, I, c)
eps1 = 100; theta = z(1);omega = z(2);

alpha = - (mr*g/I)*(theta) - c*(1 - exp(-eps1*omega)./(1 + exp(-
eps1*omega))); zdot = [omega alpha]';
end
```

Appendix 2.a)

This appendix has the MATLAB code for the first program we used in trying to solve the problem of getting to a symmetric configuration from an asymmetric configuration. It essentially takes in the locations of masses that you would like to add at various places on the robot and outputs the values of each mass to be added while driving the new difference in the mass properties to zero. We must add three or more masses for the difference to go down to zero.

```
function masses = equality_max_batt(locs)

%valsIn and valsOut are arrays of four elements
%vals(1)  = m*y
%vals(2)  = m*z
%vals(3)  = I
% these values must be known from experiment

% masses in kg, distance in cm, inertia in kg.(cm^2)
valsIn(1) = 61.16; % m*y of inner leg with battery
(0.64*96*(cos(3.9*pi/180))^2 -> measured
valsIn(2) = 4.17; % m*z of outer leg with battery (m*y*tan(3.9*pi/180))
valsIn(3) = 4147.9; % inertia (from measured time period and m*y values)

valsOut(1) = 61.24; % m*y of outer leg
valsOut(2) = .64; % m*z of outer leg
valsOut(3) = 4365.4; % inertia of outer leg about hip

%Each row of locs represents the position for each individual added mass
%locs(:,1) = which leg? (0 = outer, 1 = inner)
%locs(:,2) = y-location
%locs(:,3) = z-location

eqIn = [valsIn(1); valsIn(2); valsIn(3)];
eqOut = [valsOut(1); valsOut(2); valsOut(3)];
eqDiff = eqOut - eqIn;

b = eqDiff;
ck = (2*locs(:,1)-1)';
A = [locs(:,2)'.*ck;locs(:,3)'.*ck;(locs(:,2).^2+locs(:,3).^2)'.*ck];

m = A\b;

newIn = eqIn + A*(locs(:,1).*m);
newOut = eqOut + A*((locs(:,1)-1).*m);
newDiff = newOut-newIn;

masses = [m; newDiff];
```

Appendix 2.b)

This section includes the code for the second program we used in going from an asymmetric distribution to a symmetric distribution. It is essentially a GUI that computes the new differences in the mass properties every time the user inputs an old set of values of the properties and a mass at a particular location on a particular leg. It is a tool that helps the user intuitively reach the symmetric configuration.

```
function [masses,finIn,finOut] = equality_auto(eqIn,eqOut)

%valsIn and valsOut should be arrays of four elements
%vals(1) = m
%vals(2) = l_y
%vals(3) = l_z
%vals(4) = I

%Each row of locs represents a different position for an added mass
%locs(:,1) = which leg? (0 = outer, 1 = inner)
%locs(:,2) = y-location
%locs(:,3) = z-location

%eqIn = [valsIn(1)*valsIn(2);valsIn(1)*valsIn(3);valsIn(4)]
%eqOut = [valsOut(1)*valsOut(2);valsOut(1)*valsOut(3);valsOut(4)]

eqDiff = eqIn-eqOut;

fprintf(1,'The differences in the leg measurents are:\n');
fprintf(1,'my: %5.5f   mx: %5.5f   I: %5.5f\n\n',eqDiff);

cont = 'Y';
n = 0;

while strncmpi(cont,'Y',1)

    leg = input('To which leg do you want to add mass? (0 = outer, 1 =
inner)');
    while leg ~= 0 && leg ~= 1
        fprintf(1,'That is not a valid input.\n');
        leg = input('To which leg do you want to add mass? (0 = outer, 1
= inner)');
    end

    m = input('What is the mass you want to add?');
    y = input('What is the y-coordinate of the mass?');
    z = input('What is the z-coordinate of the mass?');

    if leg == 1
        newIn = eqIn + m*[y;z;y^2+z^2];
        newOut = eqOut;
        fprintf(1,'\nThe new inner leg measurents are:\n');
        fprintf(1,'my: %5.5f   mx: %5.5f   I: %5.5f\n\n',newIn);

    elseif leg == 0
```

```
        newIn = eqIn;
        newOut = eqOut + m*[y;z;y^2+z^2];
        fprintf(1,'\nThe new outer leg measurents are:\n');
        fprintf(1,'my: %5.5f   mx: %5.5f   I: %5.5f\n\n',newOut);
    end

    newDiff = newIn-newOut;

    fprintf(1,'The new differences in the leg measurents are:\n');
    fprintf(1,'my: %5.5f   mx: %5.5f   I: %5.5f\n\n',newDiff);

    keep = input('Do you want to keep this mass? (Y/N)  ','s');
    if strncmpi(keep,'Y',1)
        eqIn = newIn;
        eqOut = newOut;
        n = n+1;
        mss(n,:) = [leg m y z];
    end

    cont = input('Do you want to add another mass? (Y/N)  ','s');
end

masses = mss;
finIn = eqIn;
finOut = eqOut;

end
```

Appendix 2.c)

This section contains a sample session of the GUI (code in Appendix 2.b) and shown here is how we went from the asymmetric configuration measured on Dec. 5th **(Table 4.2)** to a symmetric configuration while keeping all the constraints in mind.

>> equality_auto([70.3; 4.17; 4584.5],[63.4; 0.66; 4304.7])
% The first matrix is the *my, mz* and $I_{/h}$ values respectively of the inner leg as a column vector and the second matrix signify the same values in the same order for the outer leg.

The differences in the leg measurents are:
my: 6.90000   mz: 3.51000   I: 279.80000

% Positive differences signify that the value is greater for the inner leg. Clearly we must
% try and reach at a symmetric configuration while keeping the constraints (Sec. 4, page 10)
% in mind. We could try reducing the *my* and $I_{/h}$ values of the inner leg by moving the hip-
% spring assembly up and try to equalize the *mz* values simply by using the second battery
% appropriately.

% First, moving the hip-spring assembly up the inner leg
% Note that the hip-spring assembly contains two mounts of 37.5 grams (both on inner

```
% leg), two hooks of six grams each, and two pairs of springs, that is each pair of springs
% has a rest-length of 10 cm, and each spring has a mass of 16.5cm.

% First, moving the mount :

To which leg do you want to add mass? (0 = outer, 1 = inner)1
What is the mass you want to add? 0.075 % adding a positive mass at new location
What is the y-coordinate of the mass? 67 % new location of the mount
What is the z-coordinate of the mass? 0

The new inner leg measurents are:
my: 75.32500   mx: 4.17000   I: 4921.17500

The new differences in the leg measurents are:
my: 11.92500   mx: 3.51000   I: 616.47500

Do you want to keep this mass? (Y/N)  Y
Do you want to add another mass? (Y/N)  Y
To which leg do you want to add mass? (0 = outer, 1 = inner)-0.075
That is not a valid input.
To which leg do you want to add mass? (0 = outer, 1 = inner)1
What is the mass you want to add?-0.075 % adding a negative mass at old location
What is the y-coordinate of the mass?77 % old location of the mount
What is the z-coordinate of the mass?0

The new inner leg measurents are:
my: 69.55000   mx: 4.17000   I: 4476.50000 % measurements of the inner leg after moving
mount

The new differences in the leg measurents are:
my: 6.15000   mx: 3.51000   I: 171.80000 % new differences

Do you want to keep this mass? (Y/N)  Y
Do you want to add another mass? (Y/N)  Y

% moving the hook up by 10 cm from its old location
To which leg do you want to add mass? (0 = outer, 1 = inner)1
What is the mass you want to add?.012
What is the y-coordinate of the mass?63.5
What is the z-coordinate of the mass?0

The new inner leg measurements are:
my: 70.31200   mz: 4.17000   I: 4524.88700

The new differences in the leg measurements are:
my: 6.91200   mz: 3.51000   I: 220.18700

Do you want to keep this mass? (Y/N)  Y
Do you want to add another mass? (Y/N)  Y
To which leg do you want to add mass? (0 = outer, 1 = inner)1
What is the mass you want to add?-0.012
What is the y-coordinate of the mass?73.5
What is the z-coordinate of the mass?0

The new inner leg measurents are:
my: 69.43000   mz: 4.17000   I: 4460.06000

The new differences in the leg measurements are:
my: 6.03000   mz: 3.51000   I: 155.36000

Do you want to keep this mass? (Y/N)  Y
Do you want to add another mass? (Y/N)  Y
To which leg do you want to add mass? (0 = outer, 1 = inner)1

% moving the springs up by 10 cm from their old location
What is the mass you want to add?0.066
What is the y-coordinate of the mass?52.5
What is the z-coordinate of the mass?0

The new inner leg measurements are:
my: 72.89500   mx: 4.17000   I: 4641.97250

The new differences in the leg measurements are:
my: 9.49500   mx: 3.51000   I: 337.27250

Do you want to keep this mass? (Y/N)  Y
Do you want to add another mass? (Y/N)  Y
To which leg do you want to add mass? (0 = outer, 1 = inner)1
What is the mass you want to add?-0.066
What is the y-coordinate of the mass?62.5
What is the z-coordinate of the mass?0

The new inner leg measurements are:
my: 68.77000   mx: 4.17000   I: 4384.16000

The new differences in the leg measurements are:
my: 5.37000   mx: 3.51000   I: 79.46000
Do you want to keep this mass? (Y/N)  Y

% Adding a second battery (750 gms) on the outer leg, split in two equal halves on
% opposite sides of the outer leg.
Do you want to add another mass? (Y/N)  Y
To which leg do you want to add mass? (0 = outer, 1 = inner)0
What is the mass you want to add? 0.75
What is the y-coordinate of the mass? 7
What is the z-coordinate of the mass? 4.5
```

```
The new outer leg measurements are:
my: 68.65000   mx: 4.03500   I: 4356.63750

The new differences in the leg measurements are:
my: 0.12000   mx: 0.13500   I: 27.52250

% really low differences! Hence this configuration works.

Do you want to keep this mass? (Y/N)  Y
Do you want to add another mass? (Y/N)  N

ans =

    1.0000    0.0750   67.0000         0
    1.0000   -0.0750   77.0000         0
    1.0000    0.0120   63.5000         0
    1.0000   -0.0120   73.5000         0
    1.0000    0.0660   52.5000         0
    1.0000   -0.0660   62.5000         0
         0    0.7500    7.0000    4.5000
```