Balancing of a Robotic Unicycle

By Neil D'Souza-Mathew (PEM)

Fourth-year undergraduate project in Group F, 2007/2008

Supervised by Prof. Jan Maciejowski

I hereby declare that, except where specifically indicated, the work submitted herein is my own original work.

Contents

1 Technical Abstract

The problem of making a robotic unicycle respond to a reference velocity demand while balancing itself in the pitch direction, was investigated. The dynamics of the problem were considered and a linear controller was implemented on it.

Virtual Work was used to consider the work done due to infinitesimal changes in each of the unicycle's coordinates, and the *Lagrangian Equations* were then used to derive its trajectory.

The resulting equations were linearised and *infinite-horizon* LQR was used to obtain a stable state-feedback gain vector. Performance constraints were specified, and a loop-shaping method was used to make the closed-loop system satisfy them. The resulting closed-loop system of the controller and nonlinear pitch-dynamics were simulated in Simulink and its stability verified.

LabView software was developed to calibrate all sensors to *SI units*, as well as to implement the controller. The IR transceivers for the wheel tachometer were replaced by a rotary encoder. A test rig was designed and constructed, and tests were carried out demonstrating that the system was in a limit cycle. Describing-function analysis was used to propose solutions to the limit cycle problem.

Due to necessity, a new testing strategy was devised and constructed requiring less resources than the previous design.

2 Introduction

2.1 Background

A robotic unicycle was built by Mark Mellors and Andrew Lamb [?] in the academic year 2004/2005. The design was chosen to emulate an actual unicyclist, with an inertial disc to represent the unicyclist's arms. Using the state-space convention, the inputs to the unicycle are the *PWM signals* to the motors for the inertial disc and for the main drive wheel. The measured outputs are the *tilt* which is triangulated via IR range-finders, the *rate of tilt* from rate gyroscopes and the *velocity* of each of the motors measured by IR and encoder discs. The inputs and outputs are connected to the I/O modules of a National Instruments CompactRio device, on which the controller may be implemented.

2.2 Aims

This project was intended to design a software control system to achieve a fast response to a reference velocity on the unicycle, while balancing it about its vertical equilibrium point, in the pitch case only. A secondary aim was to gain an understanding of the dynamics and control with regard to balancing the unicycle in the pitch direction.

The practicalities were intended to be achieved with the back-up of dynamics and control theory from across the Engineering Tripos. Alternative routes to solving the problem without linear methods are also possible, such as Reinforcement Learning in the form of Q-Learning. Appendix A provides a brief explanation of this method. The main issue with this is that trials need to be repeated a number of times depending on the convergence of the algorithm and with large systems it may not be feasible or cost-efficient to test until convergence.

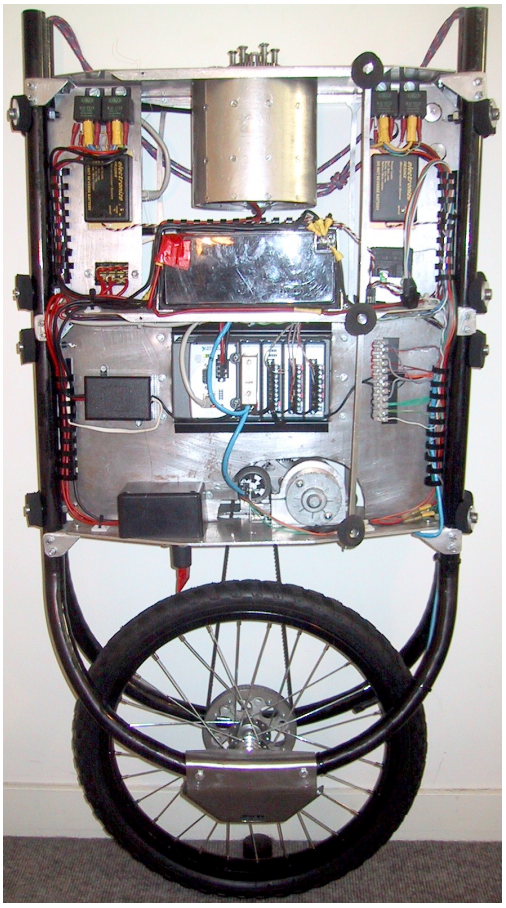

Figure 1: Photograph of Unicycle front

2.3 Method

This section is intended to give the follow-on student in this project a brief overview of the main steps taken in order to arrive at the testing stage. A separate handover document will be created going into the practical detail of the electronics, LabView software and Simulink tips. During the course of this report, brief mentions will be made of commands used in Matlab that will be time-saving for the project.

1. Deriving the equations of motion: The Lagrangian method was used for this which is well suited to the pitch case. It is similarly likely to be useful in the lateral balancing case. For purposes of understanding, the notes from the 3rd year dynamics course (3C5) [**?**] are useful.

2. Linearised state-space form: The linearisation can be achieved easily by hand as described in the 3F2 notes [**?**] and verified quickly from the nonlinear Simulink model by using the Matlab command 'linmod'.

3. LQR: The Linear Quadratic Regulator is an optimality procedure which minimises a quadratic cost function of the inputs and outputs, and can be used to return the optimal stabilising state feedback vector for a chosen set of weightings. The Matlab command used is 'lqr' and takes as argument the linearised state space matrices, the weighting matrix Q and in this case the scalar input weighting R.

4. Simulink; is very useful for testing the validity of the controller on the non-linear model. The Matlab command 'simulink' brings up a GUI that allows the construction of the non-linear plant via graphical building blocks. The controller can be placed in a feedback loop with this plant and a simulation of the time response at any point along the loop may be obtained.

5. Loop shaping: The material in the 4F1 course [**?**] on loop shaping is particularly useful for satisfying performance constraints (for example, constraints on bandwidth, sensitivity or complementary sensitivity). In particular, the notes on loop gain crossover-frequency limitations due to RHP poles and zeros, and the material on lead-lag compensators were useful for this project.

6. LabView software: LabView and the CompactRio device are made by the same company, National Instruments. Section 7 gives an overview of the programming system for the unicycle. National Instruments provide a few good programming tutorials [**?**] available on the net. The next stage of the project should not require a heavy element of coding and most new requirements should be achievable via modifications of the existing code.

7. Testing: The unicycle is a powerful and heavy piece of equipment and requires sturdy rigging. While it was possible to test the pitch case balancing with zero reference velocity, it is likely that for the lateral balancing case the reference velocity constraint may need to be lifted during testing, and therefore a larger testing space would be necessary.

3 Equations of Motion

3.1 Lagrangian derivation

The derivation of the dynamics equations is described below, with only key results detailed. Figure 2 shows the model being used for the derivation of the unicycle trajectory. The system is considered as being composed of the frame and the wheel, joined at the centre of the wheel.

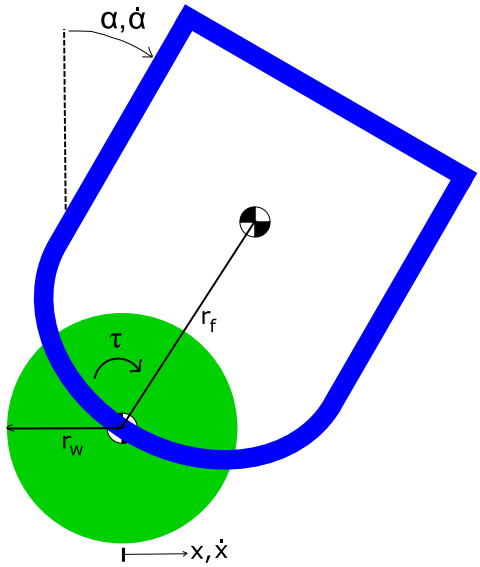

Figure 2: Unicycle Model

where:

τ is the applied torque

α is the angle of tilt of the frame w.r.t. the vertical

$\dot{x}$ is the translational velocity of the centre of the wheel

r_w is the radius of the wheel

r_f is the distance from the centre of the wheel to the frame's centre of mass

I_{fg} is the moment of inertia of the frame w.r.t. its own centre of mass

I_{wg} is the moment of inertia w.r.t. its own centre of mass

To derive the Lagrangian equations, consider Virtual Work applied to each of the two coordinates. Consider first the case where x is held fixed, and α is varied infinitesimally. Figure 3, shows that for an infinitesimal rotation of the frame $\delta\alpha$, the work done is $\tau \cdot \delta\alpha$

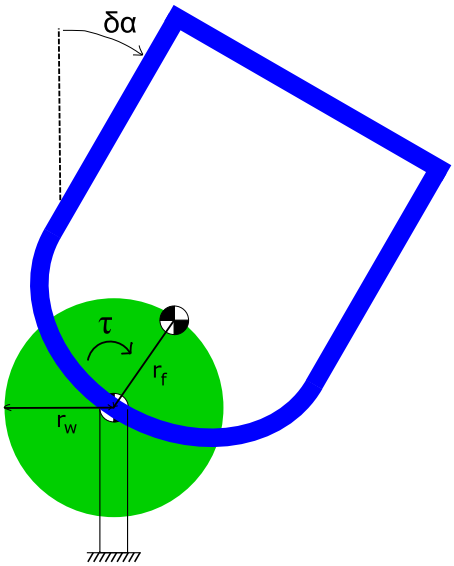

Figure 3: Virtual Work for α

Next, for the case where α is held fixed, Figure 4 illustrates that for an infinitesimal translation δx of the unicycle as a whole, the work done is $\frac{\tau}{r_w} \cdot \delta x$

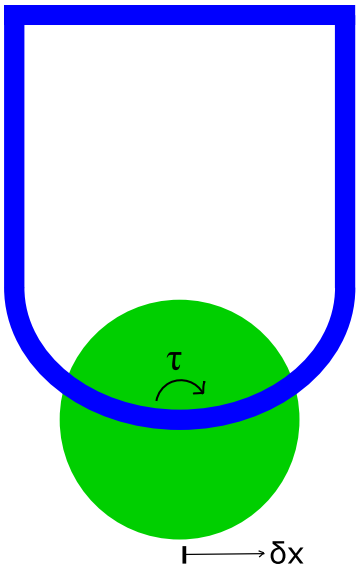

Figure 4: Virtual Work for x

The Virtual Work equation is thus:

$$\delta W = \frac{\tau}{r_w} \cdot \delta x + \tau \cdot \delta\alpha$$

And the Lagrangian L is:

$$L = T - V$$

where T is the Kinetic Energy of the system and V is the Potential Energy of the system. The Lagrangian equations are thus:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \left(\frac{\partial L}{\partial x}\right) = \frac{\tau}{r_w}$$
$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\alpha}}\right) - \left(\frac{\partial L}{\partial \alpha}\right) = \tau$$

$$T = \frac{1}{2}\left(m_w \dot{x}^2\right) + \frac{1}{2} m_f \left(\dot{x}^2 + 2\dot{x} r_f \dot{\alpha} \cos\alpha + r_f^2 \dot{\alpha}^2\right) + \frac{1}{2}\left(\frac{I_{wg}\dot{x}^2}{r_w^2}\right) + \frac{1}{2}\left(I_{fg}\dot{\alpha}^2\right)$$
$$V = m_f g r_f \cos\alpha$$

Putting the above equations together gives us, in nonlinear state-space form:

$$\frac{d}{dt}\begin{bmatrix} \dot{x} \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} \left(\frac{1}{ac - b^2\cos^2\alpha}\right)\left[bc\dot{\alpha}^2 \sin\alpha - b^2 g \sin\alpha\cos\alpha + \tau\left(\frac{c - b r_w \cos\alpha}{r_w}\right)\right] \\ \left(\frac{1}{ac - b^2\cos^2\alpha}\right)\left[bga\sin\alpha - b^2\dot{\alpha}^2\sin\alpha\cos\alpha + \tau\left(\frac{a r_w - b\cos\alpha}{r_w}\right)\right] \end{bmatrix}$$

where

$$a = m_f + m_w + \frac{I_{wg}}{r_w^2}$$
$$b = m_f r_f$$
$$c = m_f r_f^2 + I_{fg}$$

These equations were put into a Simulink block as in Figure 5.

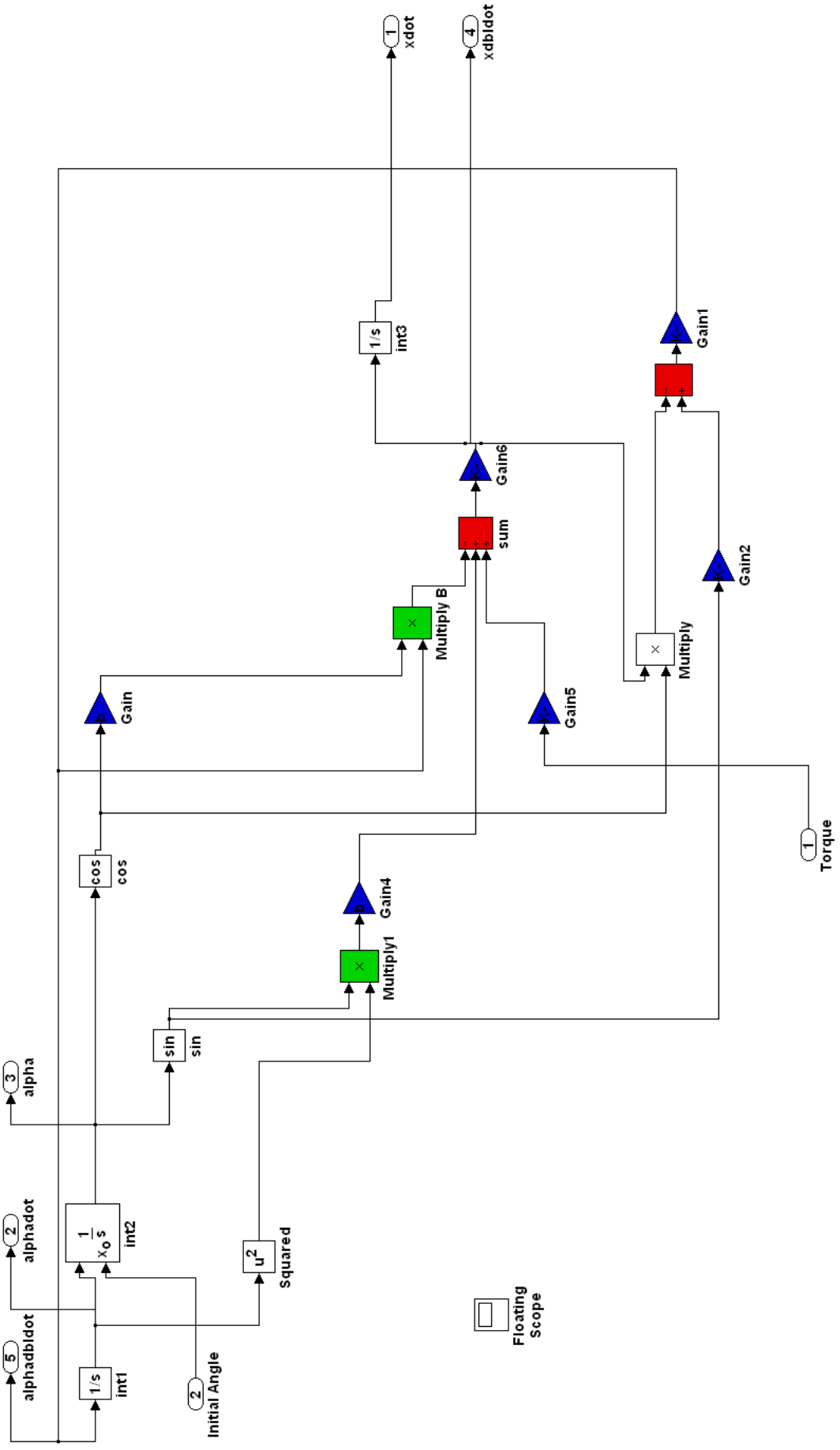

Figure 5: Nonlinear pitch-dynamics block

3.2 Linearisation and State-Space

The linearisation was obtained by hand as shown in Appendix B. It has also been verified by the 'linmod' command on the Simulink model that will be discussed in the following chapter. So in linear state-space form where a, b and c are as defined in the previous subsection,

$$\frac{d}{dt}\begin{bmatrix} \dot{x} \\ \dot{\alpha} \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 & 0 & \left(\frac{-b^2 g}{ac-b^2}\right) \\ 0 & 0 & \left(\frac{abg}{ac-b^2}\right) \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{\alpha} \\ \alpha \end{bmatrix} + \begin{bmatrix} \left(\frac{c-br_w}{r_w(ac-b^2)}\right) \\ \left(\frac{ar_w-b}{r_w(ac-b^2)}\right) \\ 0 \end{bmatrix} \tau$$

3.3 Constants and inertial calculations

The known terms in the derivation above are:

$$m_f = 24\,kg$$

$$m_w = 1\,kg$$

$$r_f = 0.68\,m$$

$$r_w = 0.225\,m$$

The values for moment of inertia were calculated via standard approximations found in the Mechanics Databook [?].

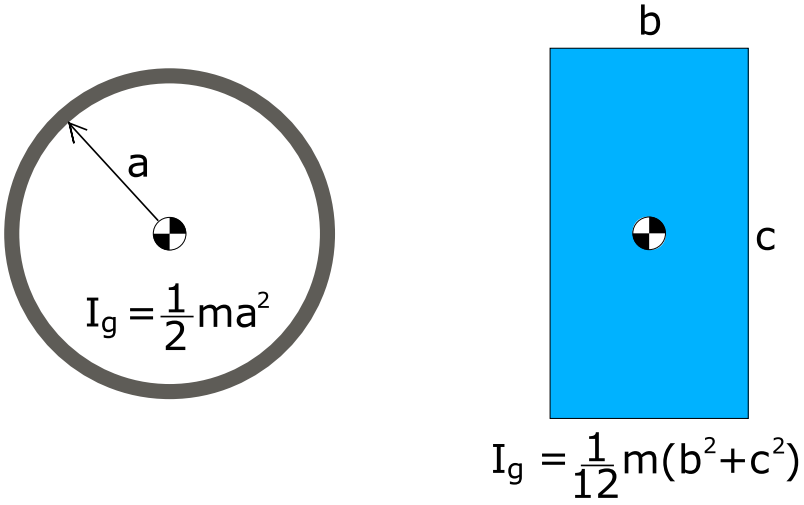

Figure 6: Moment of Inertia Approximations

The wheel was modelled as a thin hoop as in Figure 6, with all the mass distributed on the circumference of the hoop. This is a reasonable approximation as the mass of the spokes is negligible as compared with that of the rim. Thus its moment of inertia about its centre of mass,

$$I_{wg} = 1 \times 0.225^2 = 0.051\, kg.m^2$$

Also shown in Figure 6, the frame is modelled as a thin rectangular $56 \times 30 cm^2$ plate. Again, this is a reasonable approximation to make as the battery and controller lie very close to the centre of mass, and the motors are approximately equidistant from it. The rest of the mass is made up from sheet metal. Thus.

$$I_{fg} = \frac{24 \times \left(0.56^2 + 0.3^2\right)}{12} = 0.807\, kg.m^2$$

Finally, the linearised state-space form for this system is:

$$\frac{d}{dt}\begin{bmatrix} \dot{x} \\ \dot{\alpha} \\ \alpha \end{bmatrix} = \begin{bmatrix} 0 & 0 & -60.72 \\ 0 & 0 & 96.69 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{\alpha} \\ \alpha \end{bmatrix} + \begin{bmatrix} 1.23 \\ -1.69 \\ 0 \end{bmatrix} \tau$$

4 Linear Quadratic Regulator

A control system with state feedback was implemented. Linear Quadratic Regulation [?] was chosen to incorporate optimality conditions for calculating the state feedback vector. According to LQR, the cost function to minimise is:

$$J = \int_0^\infty \left(x^T Q x + u^T R u\right)\, dt$$

where x is the state vector and u is torque input to the wheel motor, Q and R are set to assign the relative importance of the various states and the input.

In this case, as we have 3 states, Q is a 3x3 matrix and R is a scalar. In the general case, the minimisation is solved by a continuous time recursive dynamic programming problem, the solution of which gives us an optimal (stabilising) input in the form $u = Kx$. For the case of the infinite-horizon cost function which is used here, a simple closed-form solution exists. An explanation of the infinite-horizon continuous-time LQR solution can be found in the 4th year 4F2 notes [?], however an understanding of this is not necessary to use the 'lqr' command in Matlab. LQR just gives a starting point for the controller and further work is required for performance specifications, non-linearities etc.

A diagonal Q matrix was chosen, with a greater weighting towards the angular states than the translational position. The scalar R was given the same weighting as the angular states, reflecting the fact that at least initially we are mainly interested in keeping the unicycle upright while remaining within the bounds of the motor's capabilities. A number of settings of these constants were evaluated using Bode, and the best combination of these is given below.

$$Q = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}$$

$$R = 0.1$$

which results in

$$K = \begin{bmatrix} -10 & -22.3 & -144.9 \end{bmatrix}$$

5 Performance

5.1 Performance issues

Having stabilised the linearised closed-loop system using LQR, the performance of the unicycle was considered next, using frequency response methods. A transfer function was required to describe the open loop of the Single Output system with feedback, which was most conveniently taken from the broken loop at the point after state feedback, as in Figure 7.

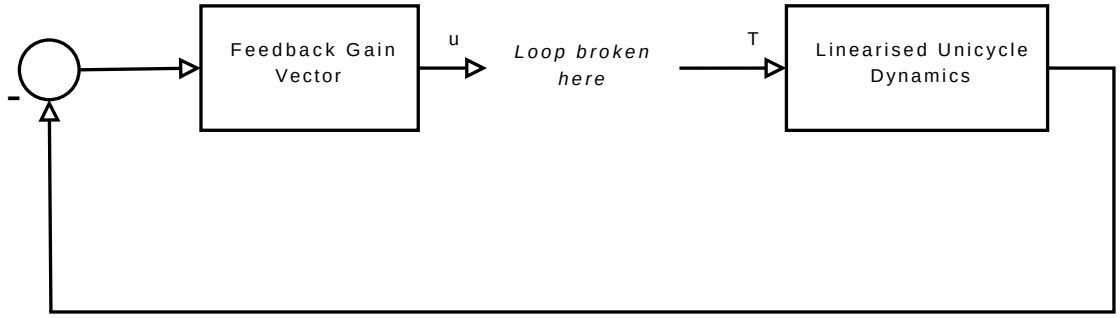

Figure 7: Broken Loop

After rearrangement, the transfer function between u and T is:

$$G = -\left[\frac{s^2\left(a_1k_1 + a_2k_2\right) + s\left(a_3k_3\right) + b_1k_1}{s^3 + ds}\right]$$

where $a_1 = 1.23$, $a_2 = -1.69$, $a_3 = -1.69$, $b_1 = -16.53$, $d = -96.69$ corresponding to the unicycle dynamics,

and $k_1 = -10$, $k_2 = -22.3$, $k_3 = -144.9$ from the state feedback vector derived via LQR.

The Bode diagram of this transfer function is shown in Figure 8. The plateau region on the magnitude plot between $0.5 rad\, s^{-1}$ and $10 rad\, s^{-1}$ is unattractive as the loop gain is only around $10dB$. This also translates to low rejection of disturbances (i.e. poor sensitivity) at these frequencies. For low loop gains, a continuously falling magnitude characteristic would be better. Furthermore, the RHP pole in the unicycle dynamics at $9.8 rad\, s^{-1}$ imposes a lower bound on the bandwidth of the system to ensure a counterclockwise encirclement of the -1 point in the Nyquist locus.

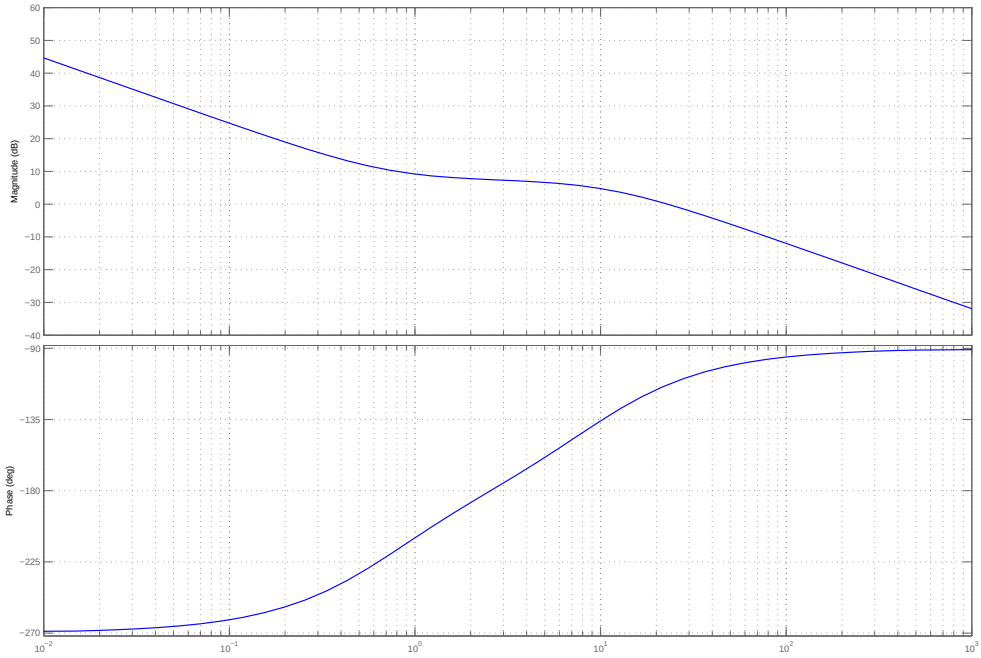

Figure 8: Plant Bode diagram

5.2 Performance Specification

Three performance specifications were thus derived.

1. A crossover frequency of $20rad\,s^{-1}$ is desired for the loop gain. This frequency was chosen for being greater than the RHP pole, and has the consequence that disturbances (generally low frequency) of frequency less than $3Hz$ are attenuated, as is sensor noise of frequencies greater than $3Hz$.

2. A Phase Margin greater than 40^o and a Gain Margin greater than $6dB$ is required.

3. Elimination of the 10dB plateau region, in the magnitude plot of the loop gain.

5.3 Feedback Solutions

With purely proportional gain, we can achieve the first two requirements but not the third. Extra gain is required for frequencies below $10rad\,s^{-1}$ in order to eliminate the plateau region. Figure 9, shows the magnitude characteristic of a lag compensator (with transfer function of the form $\left(\frac{s+a}{s+b}\right)$ which gives a low frequency gain that falls off between frequencies b and a.

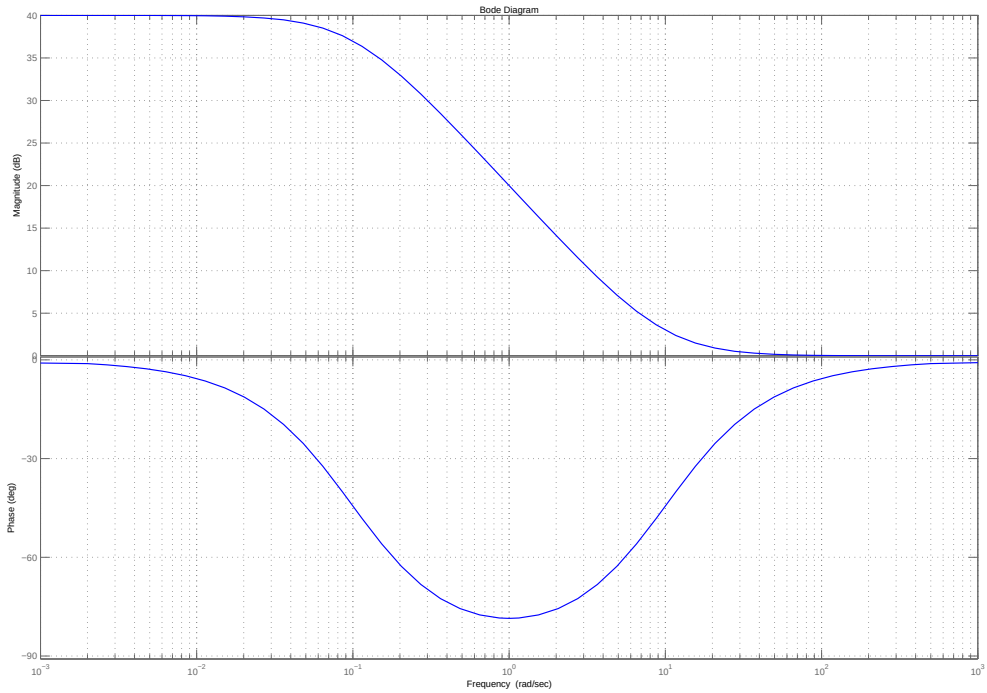

Figure 9: Characteristic of a Lag Compensator

So a lag compensator $\left(\frac{s+10}{s+0.1}\right)$ was incorporated into the open loop and a proportional gain of 0.85 was obtained with the visual aid of Matlab's SISO Design Tool. The resulting Bode plot of the transfer function is shown in green in Figure 10, meeting all the performance specifications. The original characteristic is shown in blue for purposes of comparison.

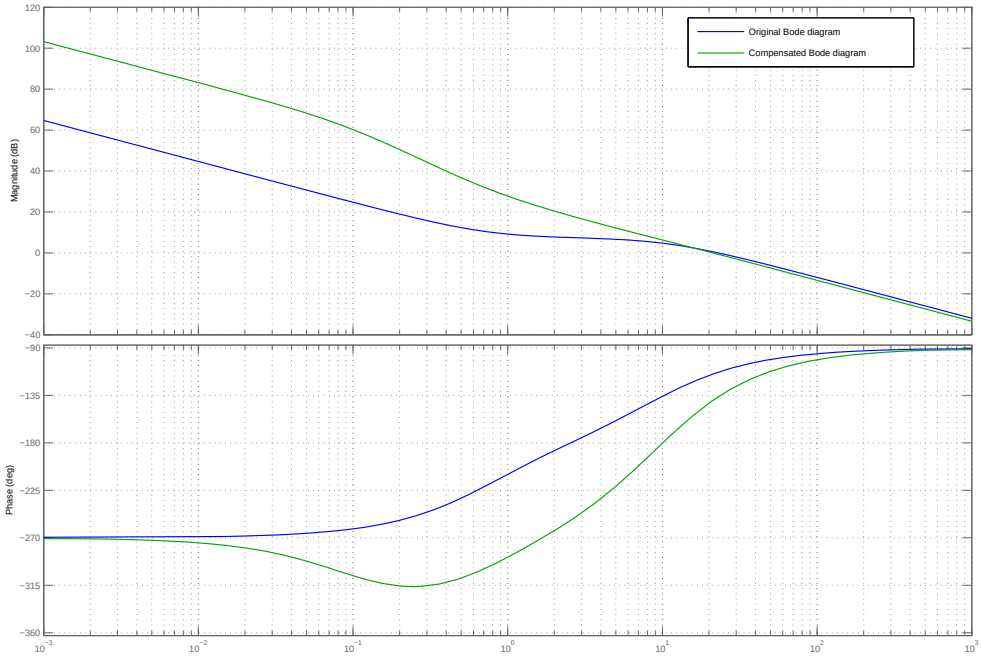

Figure 10: Compensated Plant Bode diagram

6 Simulated time response

The controller was incorporated in Simulink, around the nonlinear pitch dynamics block created at the end of Section 3.1. This was used initially to examine the stability of the LQR controller in the closed-loop with an initial angular position offset. Subsequently, after the loop-shaping process the compensator was incorporated into the closed-loop and the time response of the system was again simulated. Figure 11 shows the closed-loop block diagram with the compensator included. The contents of the Simulink block 'Non-linear Unicycle Block' was shown earlier in this report, in Figure 5.

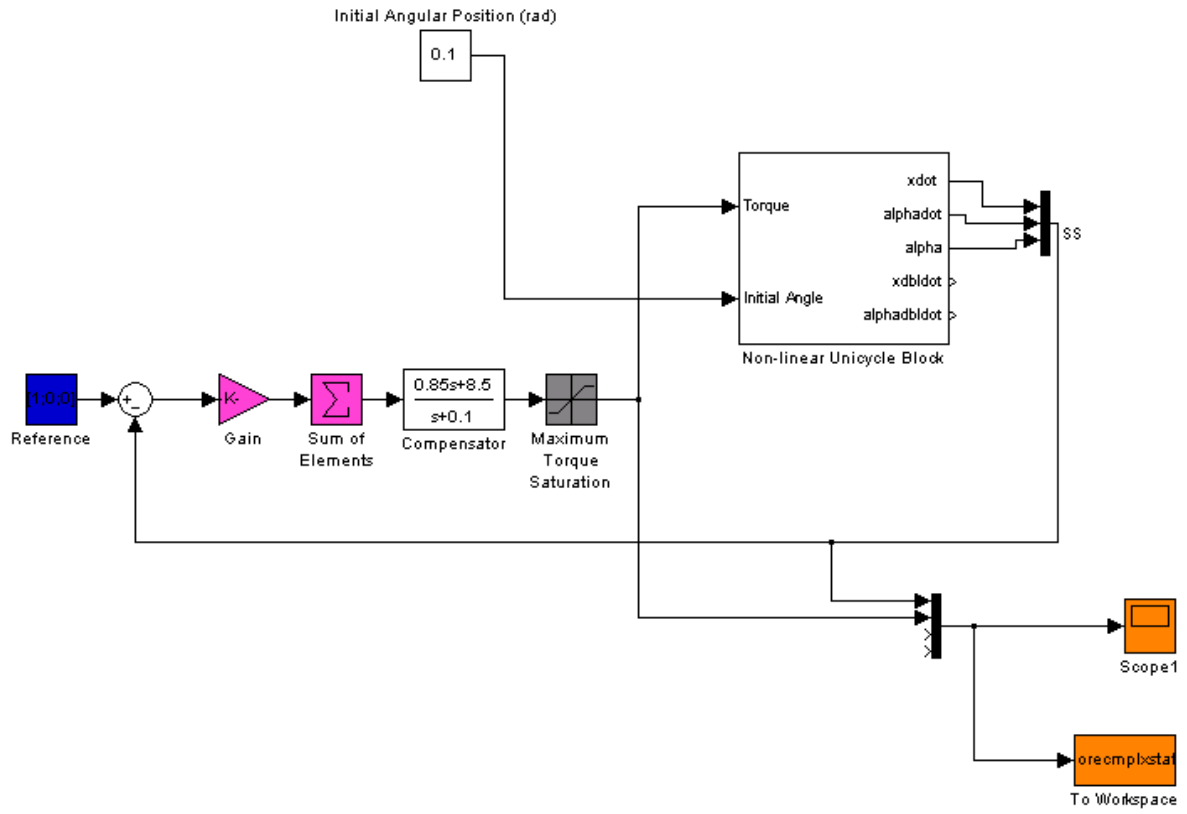

Figure 11: Closed-loop system in Simulink

An initial angular position of 0.1 radians (6^o) was used to obtain the responses, as this would be an expected degree of human error. The responses were stable for an initial position of up to 0.25 radians (15^o) after which it appears that the small angle approximations break down and the system goes unstable.

Figure 12 shows a 5 second simulation of the LQR feedback, uncompensated system, to a $1\,ms^{-1}$ translational velocity reference. The greater weighting given to the angular states can be immediately observed by the fact that the modulus of the angular position (orange) is always decreasing, and the angular velocity (green) is also small with a maximum of

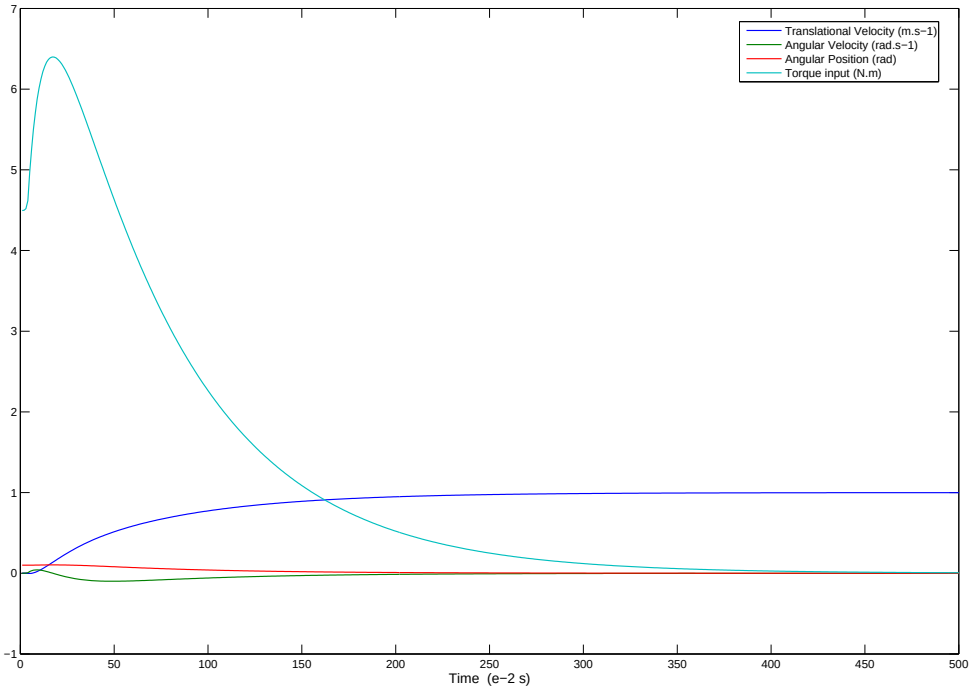

Figure 12: Simulation of time responses with LQR feedback gains

approximately $6^o s^{-1}$. The torque (purple) has a maximum of $6.5Nm$ which is well below its saturation. The translational velocity reaches its reference within 2 seconds.

With the addition of the compensator as in Figure 13, the angular position is very similar. The maximum angular velocity however, is higher, as is the Torque input which now peaks at $7.5Nm$. The response isn't as smooth as the uncompensated version, which is likely to be a result of only just meeting the Gain and Phase Margin specifications. Increasing the proportional gain leads to better Gain and Phase Margin as well as simulated time response, but also increases the crossover frequency, which is bad for the sensitivity to sensor noise.

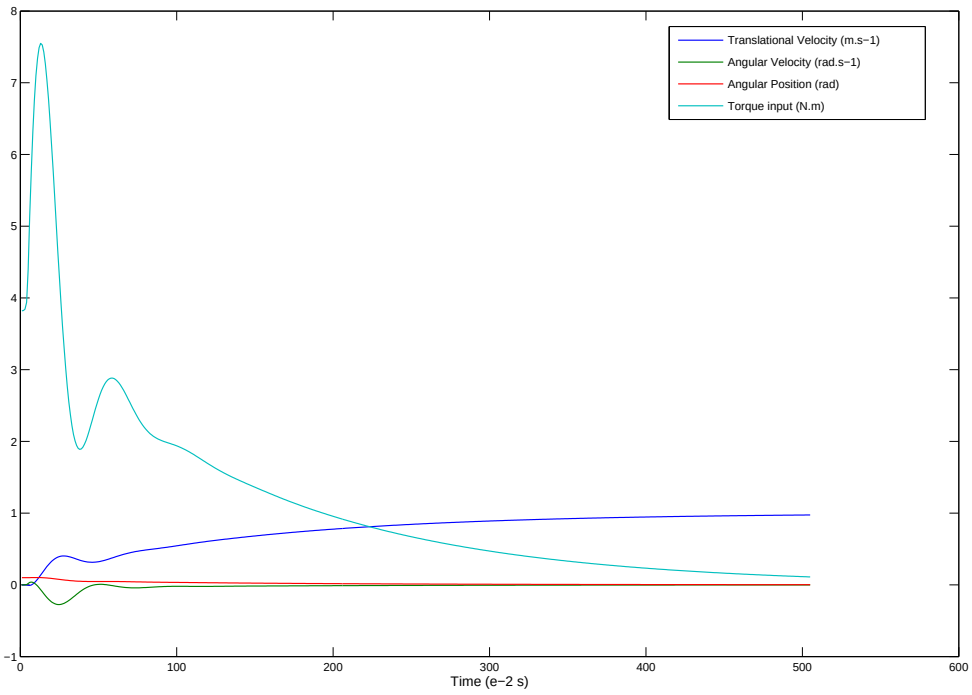

Figure 13: Simulation of time responses including LQR feedback gains and Compensator

7 Implementation

7.1 Hardware and software setup

The CompactRio device on the unicycle consists of two key components - an FPGA module, and a module containing a microprocessor. The software for the device is written and compiled in LabView 7, and a file each is created for the FPGA module and for the host (which can either be the onboard microprocessor, or a remote computer). The host file is responsible for executing the control strategy and can perform floating point operations. During development of the software, it was sufficient to keep the host on the computer and communicate via WiFi, however a faster control loop was needed during the testing phase. The host file was thus downloaded to the microcontroller, allowing a theoretical delay of only $1\,ms$. Figures 14 and 15 show graphical representations of the software setup for the development stage and the testing stage respectively.

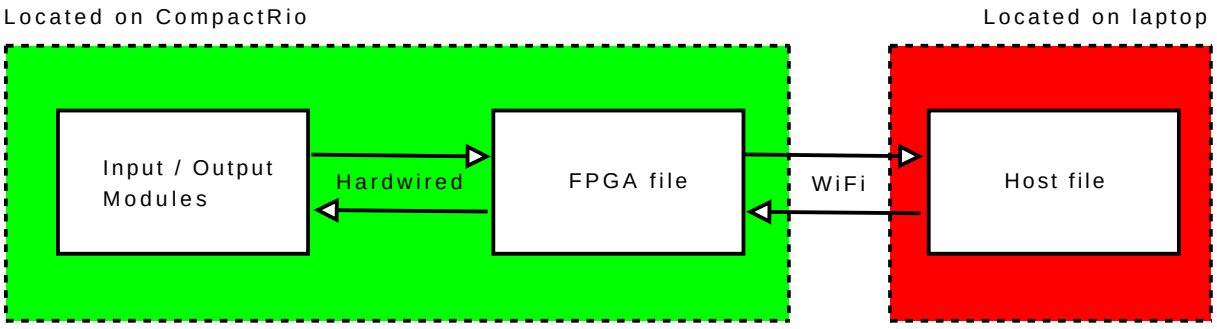

Figure 14: Software system during development

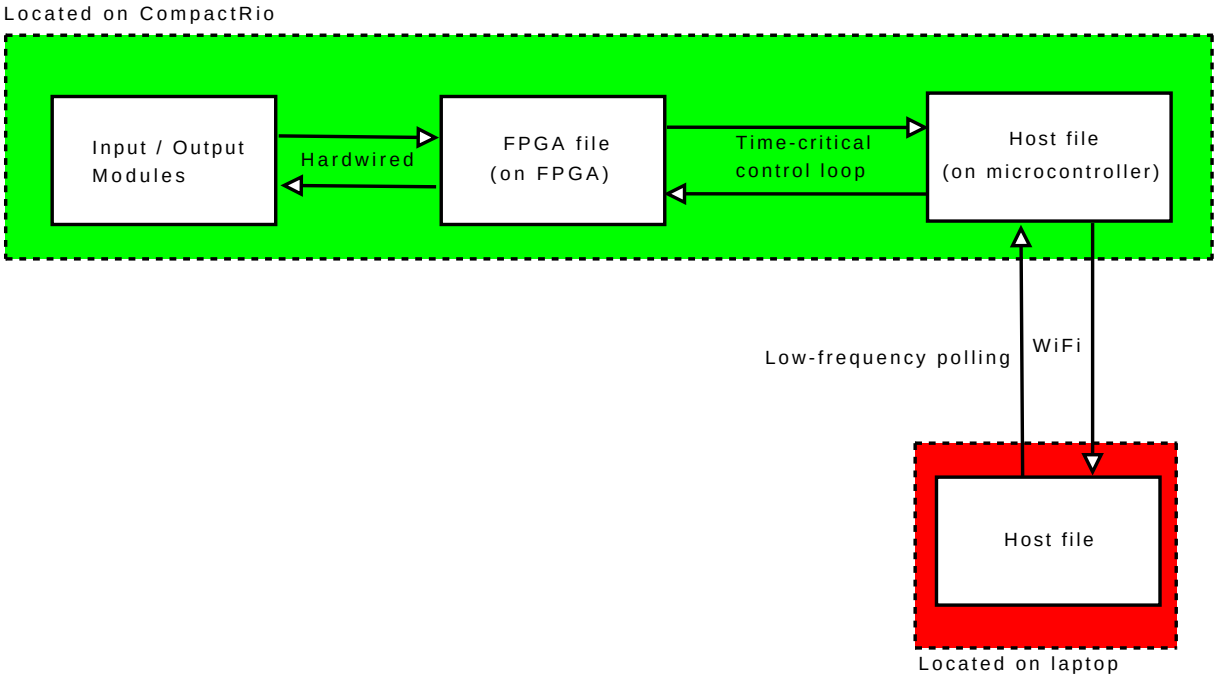

Figure 15: Time critical software loop

With the exception of the interface with the tachometer, the FPGA file provided by Mellors and Lamb [?] performed its function well.

Figure 16 shows the front-end of the Host file which resides on the laptop during testing, reading sensor data from each sensor on the unicycle like an oscilloscope, and controlling the reference velocities and feedback gain. In the Time Critical version during testing, the front end of this file is not necessary for the controller to run as it is just a low-frequency-polling bystander.

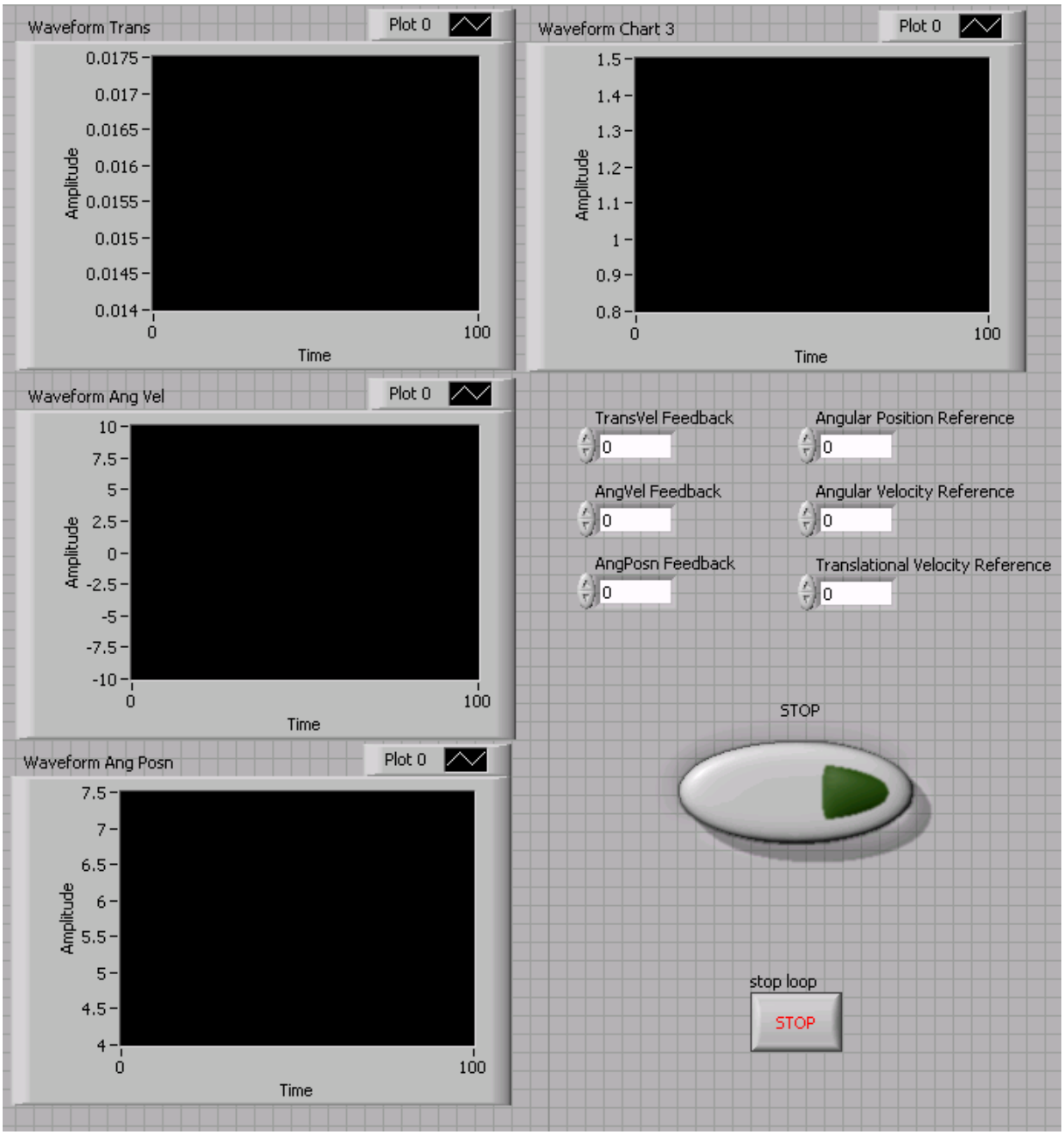

Figure 16: LabView GUI

7.2 Input / Output Measurement and Calibration

Calibration was required for each sensor to convert the voltage levels into SI units. The demanded torque also needed conversion into a PWM signal. Aside from a DC offset, the program for calibrating the rate-gyro sensors from $Volts$ to $rad\,s^{-1}$ had already been designed by Lamb[**?**].

7.2.1 Infrared Range finders

The angle (in degrees) for the range finders was computed from the Voltage input using an empirical formula created by Lamb [**?**]. For pitch control it seemed sensible to take the average of the forward and backward range-finders in order to obtain a more accurate estimate of tilt, which was then converted into radians.

7.2.2 Tachometer

The translational velocity of the unicycle is required in units of $m\,s^{-1}$, so both magnitude and sign are important in this one dimensional case. The original hardware consisted of two infrared transceiver assemblies positioned to face the surface of one of the shafts on the gear assembly, with each transceiver's focus tracing a unique circle around the shaft. In order to obtain the speed and direction, a design for the encoder disc was devised as in Figure 17.

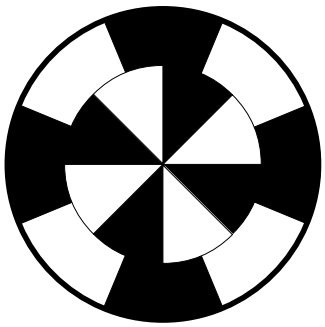

Figure 17: Encoder disc

The original software strategy as implemented in the FPGA file did not give accurate readings and this was redesigned as shown in the flowchart of Figure 18. Both the inner and outer sets of radial segments can be used to provide speed measurements. The direction of rotation is then obtained by considering the black-to-white transitions of the outer loop as the reading from the inner loop at the point of the transition depends on the direction, i.e. we are exploiting a rotational asymmetry. The decision nodes on the flowchart contain bitwise operators *XOR* and *AND*, which each evaluate to either 1 (True) or 0 (False).

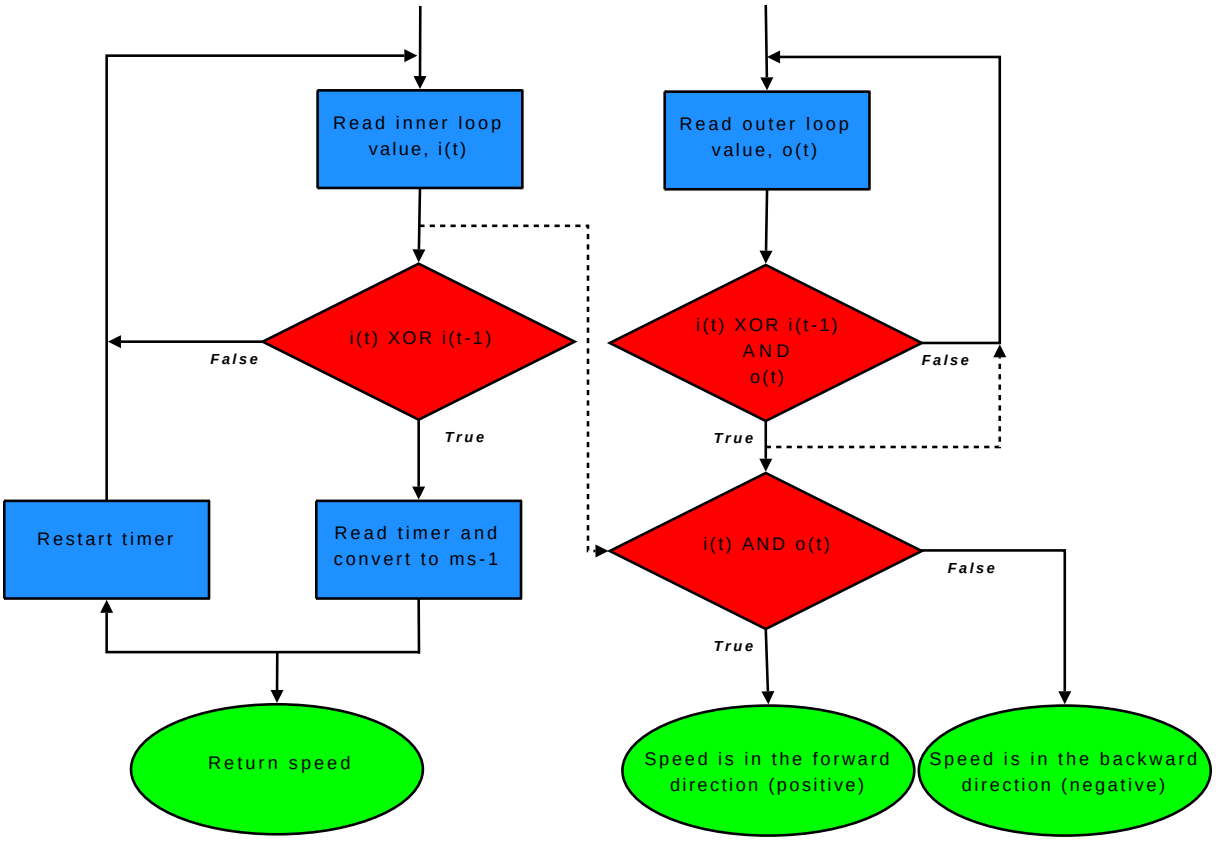

Figure 18: Tachometer Software Flowchart

In testing however, the IR assembly did not give a consistently correct direction even with the hysteresis function added in software. This was a hardware problem and was most likely due to the size of the shaft relative to the accuracy of the IR assemblies. After verifying this, the IR assemblies were removed, and replaced by a rotary encoder of the form shown in Figure 19.

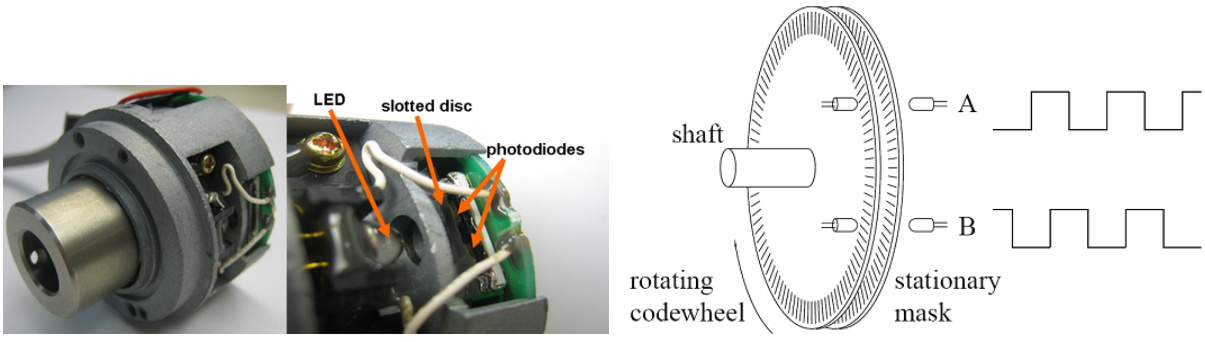

Figure 19: Diagram of a Rotary Encoder [?]

The rotary encoder was mounted on a bracket and its (frictionless) shaft was fixed onto the shaft of the motor. Its functionality is exactly the same as the IR assembly and encoder disc, but integrated into one package with far greater precision - the encoder chosen had

128 transitions per revolution, compared to the 8 transitions shown in Figure 17. The package has 4 wires, 2 for the positive and negative terminals of the power supply, and 2 digital output lines for the signals from the inner and outer radial loops. Setting the positive supply to +5V and the negative supply to Ground, the data lines are at standard TTL logic levels which are directly compatible with the CompactRio digital input module. A picture of the setup is shown in Figure 20.

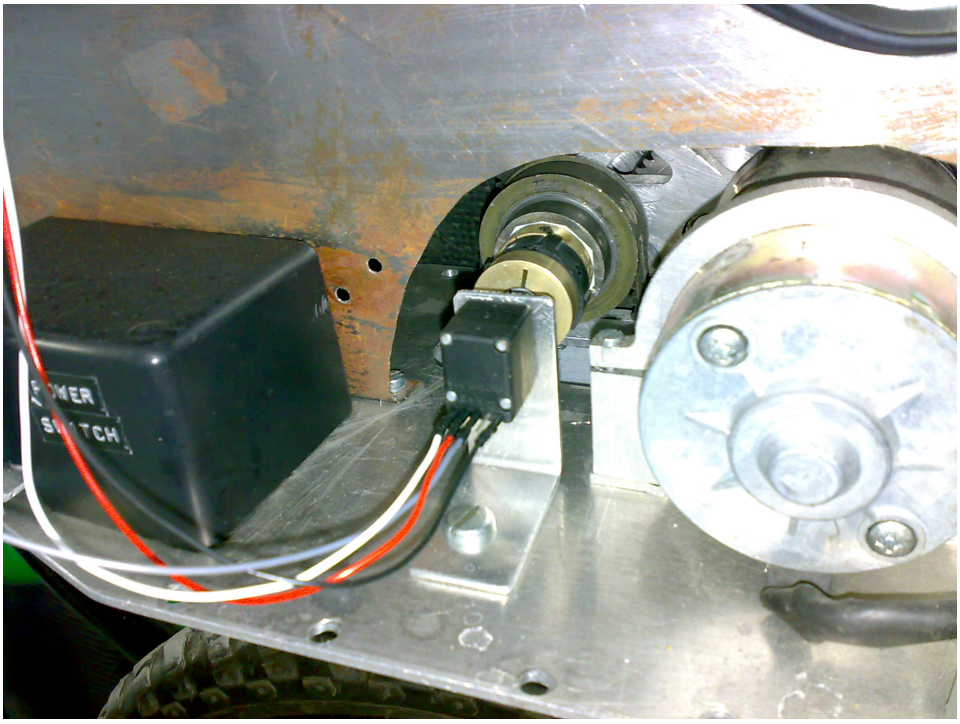

Figure 20: Rotary Encoder on Unicycle

As the operation of the rotary encoder and the voltage levels are exactly the same, no change was required of the software previously designed and this worked well in testing.

7.2.3 Torque Output

The original interface for controlling the motor was designed for 'demanded speed' whereas the Lagrangian derivation of Section 3 used Torque in the Virtual Work equation and thus also as the input variable in the state space equations. The previous software interface used a PWM duty cycle output from the CompactRio in μseconds ranging from $1080\mu s$ to $2010\mu s$, roughly corresponding to -19Nm to +19Nm on the motor.

Hence a conversion was required from 'demanded torque' into the PWM duty cycle. The conversion from PWM to motor input current was not obvious and experimentation was the most immediate way to obtain the required relationship, via spring balances. The idea here is to fix the spring balance tangentially to the rim of the wheel and for a specific PWM

output, measure the force required to stall the wheel. The stall torque relationship would closely approximate the actual relationship as the speed of the motor during balancing will be relatively small compared to its full range. This assumption can be shown to be correct from the motor datasheet [?]. This method has the downside that it is unfeasible to test for higher levels of torque and linear interpolation is used there instead.

The experimental data is shown in Figure 21, where the points in green indicate the experimental data and the points in red are the interpolated points. A crude approximation that assumes the relationship is entirely linear is shown superimposed in black. The two sets of points agree closely.

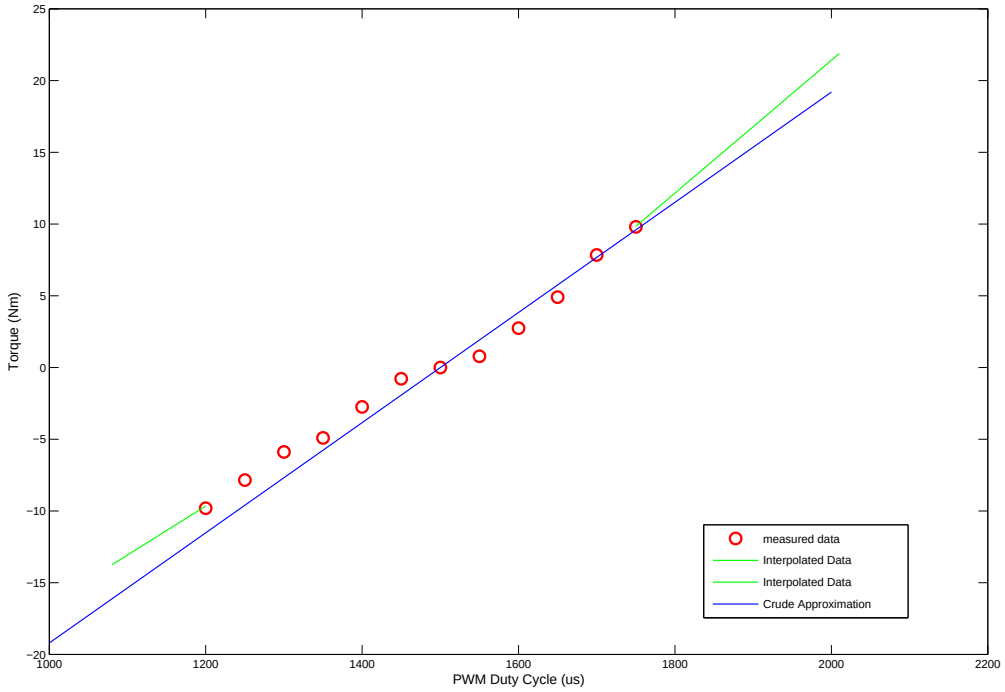

Figure 21: Torque to PWM graph

The dataset thus obtained was included into the calibration software as a *lookup-table*.

8 Testing

8.1 Requirements

The force from the mass of the unicycle combined with the maximum torque of the motor results in a maximum overall torque that can be generated by the unicycle of 40Nm, which necessitates the construction of a sturdy test rig. The requirements of a test rig are:

1. The unicycle is constrained in the roll and yaw axes, and is free to rotate in pitch.
2. Safety of the unicycle and its environment need to be assured, so there should be a means of stopping it from falling to the ground.
3. Ideally, the unicycle should be unconstrained to translate forwards and backwards.

8.2 Testing solution

A trial rig consisted of bungee ropes connected between the midsection of the unicycle on each side, and clamped to tables at the other end. It was quickly apparent that a stronger solution was needed.

The solution presented in Figure 22 was devised to guarantee the first two requirements. A department bicycle rack was used so the unicycle could be set up in a similar manner to the previous solution involving tables. A crane with the capacity to lift 5 tonnes is attached the fore and aft of the unicycle via a nylon rope.

The first problem to be tackled was that of balancing the unicycle with zero reference velocity, so the bungee ropes were fixed to the bicycle racks. Special $1mm$ thick steel plates were constructed in the workshop to be attached to the fore and aft of the unicycle to connect with the crane's shackles. A simple stress calculation was carried out to guarantee their strength in tension: $\frac{2*(25kg*9.8ms^{-2})}{1mm*10mm} = 49MPa$ which is much less 220GPa, the tensile strength of the steel. The nylon ropes are at a steep enough angle that bending moments are not considered. Pictures of the actual test rig are shown in Figure 23.

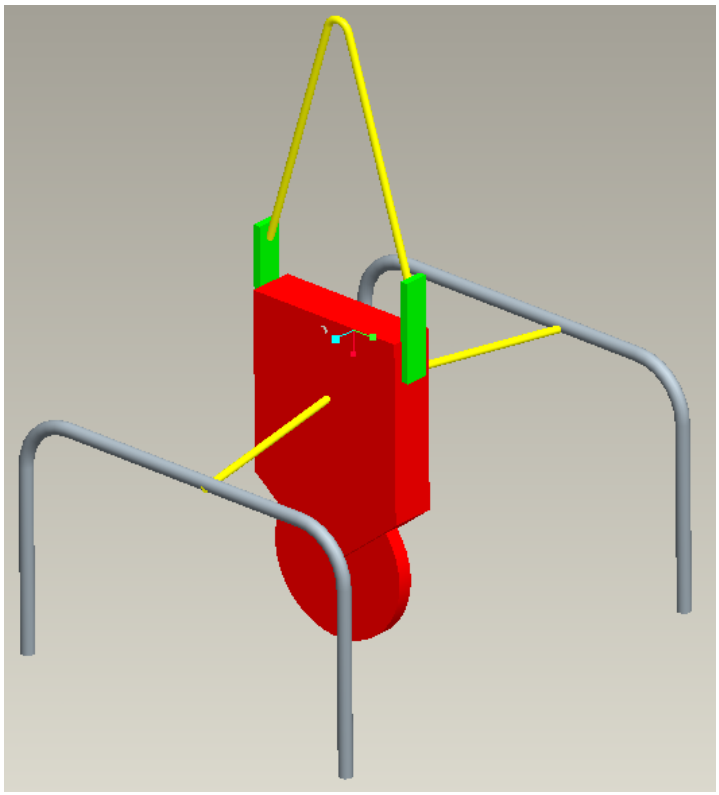

Figure 22: Test rig

Figure 23: Photographs from testing

8.3 Redesign of the test rig

As the bike rack was on temporary loan from the department, a new method was devised to work around its absence. This method has the added bonus of not using the bungee ropes, which were thought to potentially insert their own dynamics into the observed behaviour of the unicycle. The key idea is to have a long angle bracket going through the centre of the unicycle, in parallel with the axis of the main wheel. Thus if the crane is attached to the endpoints of the bracket via nylon rope as in the previous test rig, both lateral stability and the upright position of the unicycle are ensured, due to the angles near equilibrium being small. A concept design is shown in Figure 24.

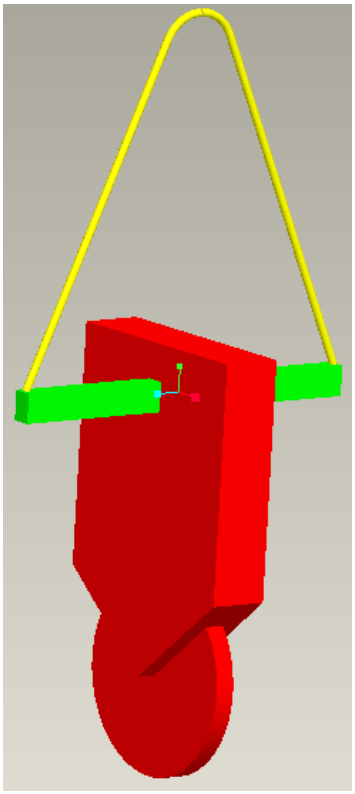

Figure 24: Proposed design for new test rig

The redesigned test rig was implemented using $1.2mm$ thick steel angle beams. A brief stress analysis [**?**] was conducted using $\sigma_{max} = \frac{Moment_{max} \times y_{max}}{I}$ to ensure that the longitudinal stresses involved were below the yield stress of steel. Figure 25 shows pictures of the bracket installed for further testing. Unlike the proposed design, it was not possible to fit the bracket through the centre of the unicycle due to the inertial motor. Instead, 2 brackets were used on each side to give the same effect. The nylon rope will wrap around the apex of each bracket.

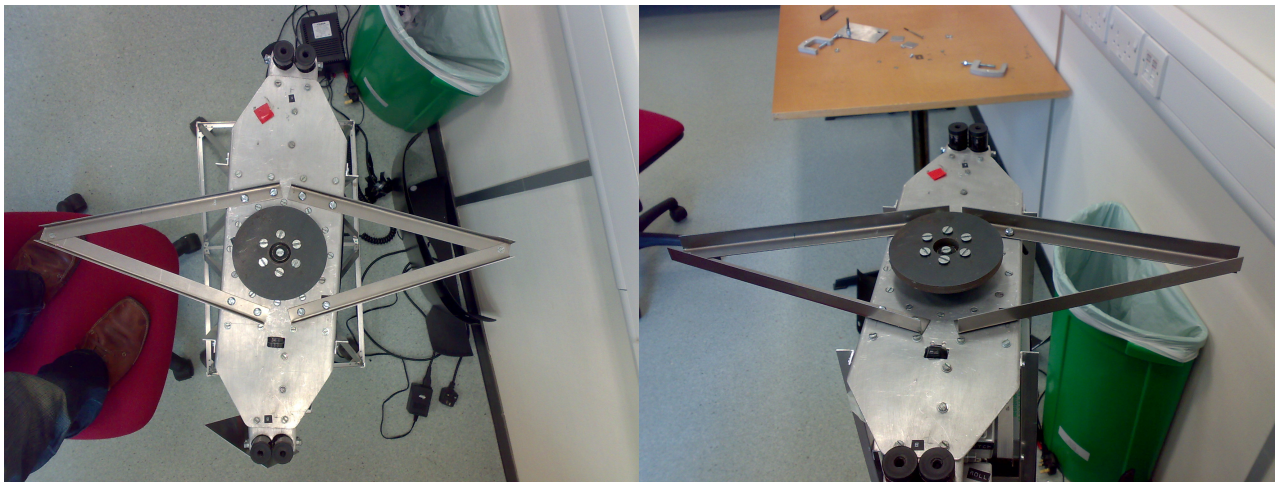

Figure 25: Redesigned testing frame

9 Results & Analysis

9.1 Results

Tests were run with the feedback gain vector and compensator as described in Sections 4 and 5. Data from the sensors was intended to be written to the Flash card on the CompactRio at a desired rate of 50ms per sample, but due to the *write-to-disk* function necessarily residing outside the *Time Critical Loop* and therefore at a lower priority than the controller, data was only sampled at an average of 120ms per sample. Figures 26, 27, 28 and 29 show the individual datasets from each of the 3 sensors, and the resulting torque demanded by the controller. An expected interpolation is shown between the points in each dataset but this is not necessarily representative.

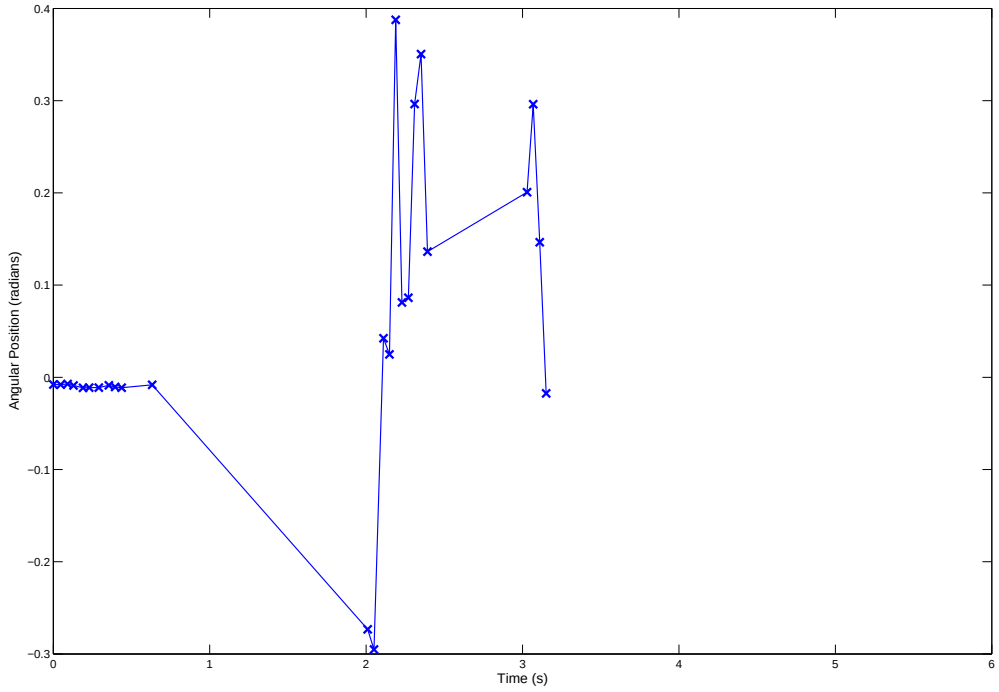

Figure 26: Angular Position sensor data

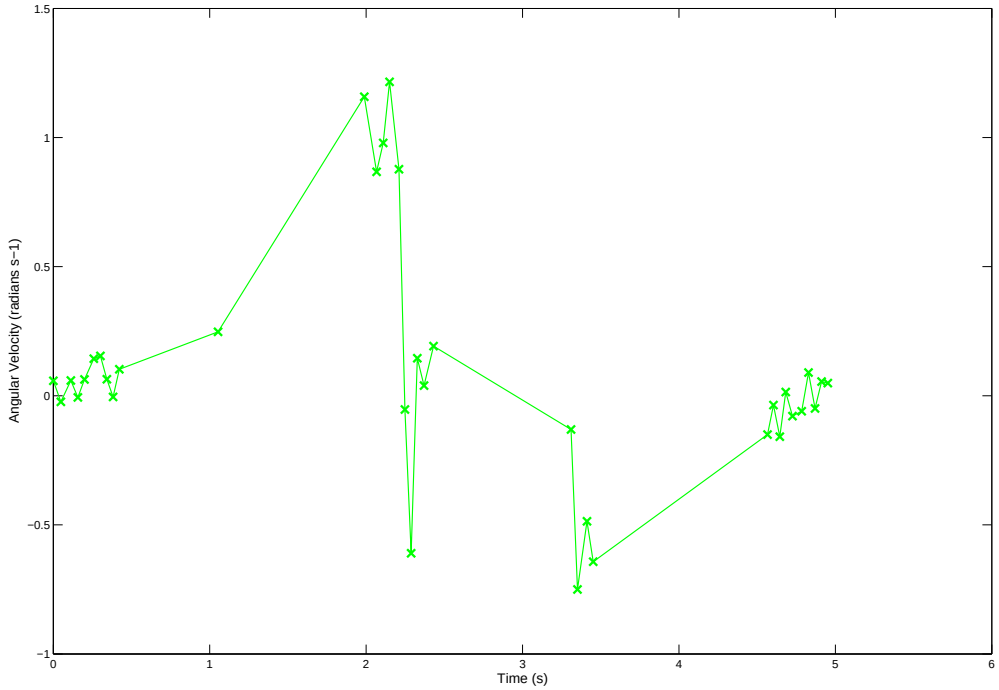

Figure 27: Angular Velocity sensor data

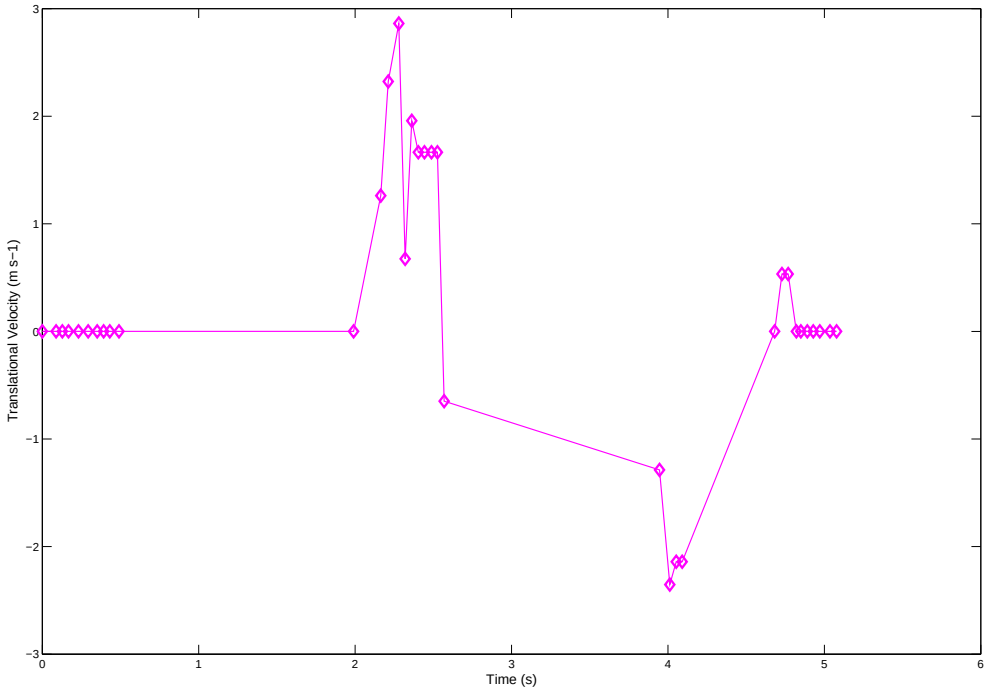

Figure 28: Translational Velocity sensor data

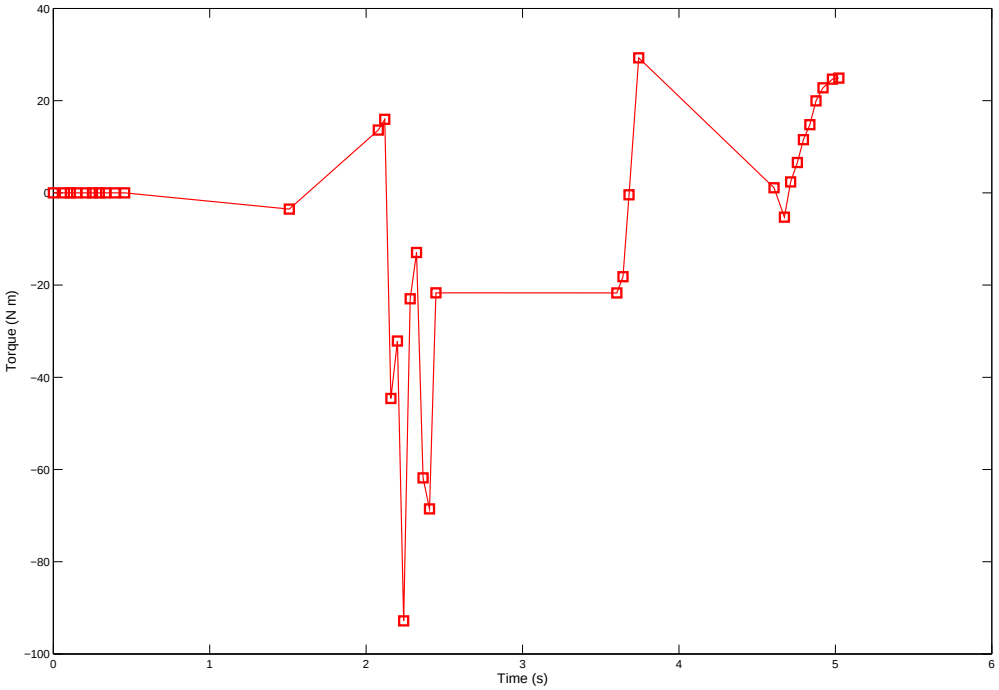

Figure 29: Demanded Torque data

Figure 30 compares all the datasets on the same graph. In order to compare the apparent gradients in each sensor's output with the demanded torque, the torque dataset was scaled down by a factor of 30.

It appears from the sensor data that the unicycle is in a limit cycle. From observation during testing, this behaviour appeared to be the result of the motor relay consistently switching at a point beyond that tolerable by small angle approximations, i.e. the polarity of the motor did not change until the unicycle had swung too far in either direction. This can be seen in Figure 30 where there is a markedly periodic behaviour in all the states, but is less noticeable from the demanded torque data.

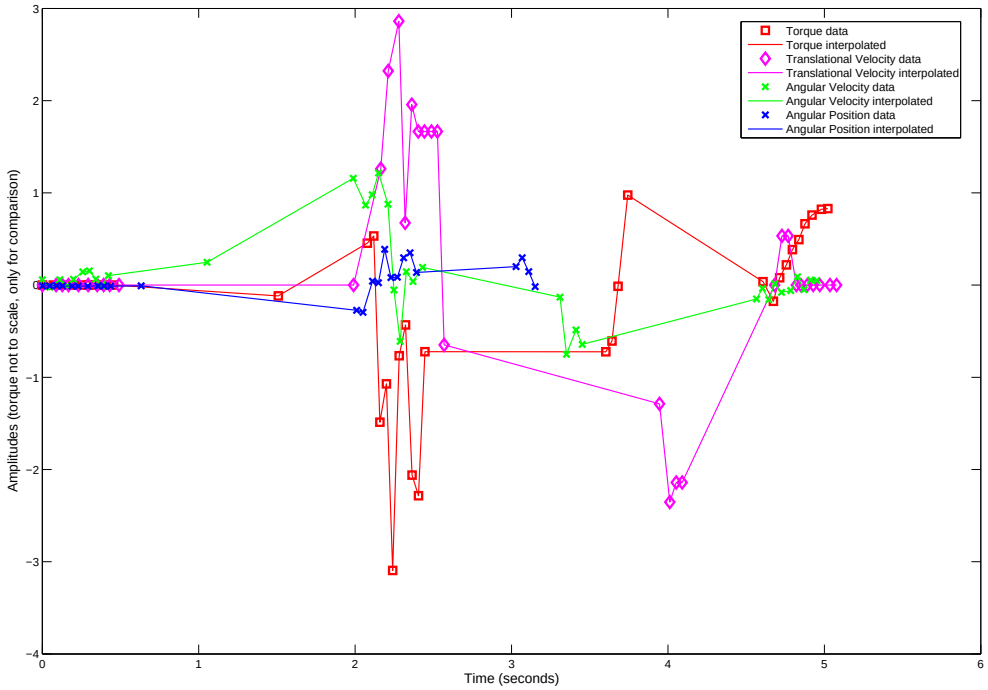

Figure 30: Comparison of all sensor data (Torque not to scale)

9.2 Describing function analysis

An explanation for limit cycle behaviour can be obtained via describing function analysis [?]. A describing function is a harmonic approximation of a non-linearity in the closed loop with the plant. We can use this in a similar way to the -1/k point, except that the describing function is a locus rather than a point. According to the theory, a limit cycle is predicted at the intersection of the describing function and the nyquist loci. Figure 31 shows the actual Nyquist locus with an example describing function, assuming that the nonlinearity is represented by dead-band in the controller. Two potential solutions can thus be proposed.

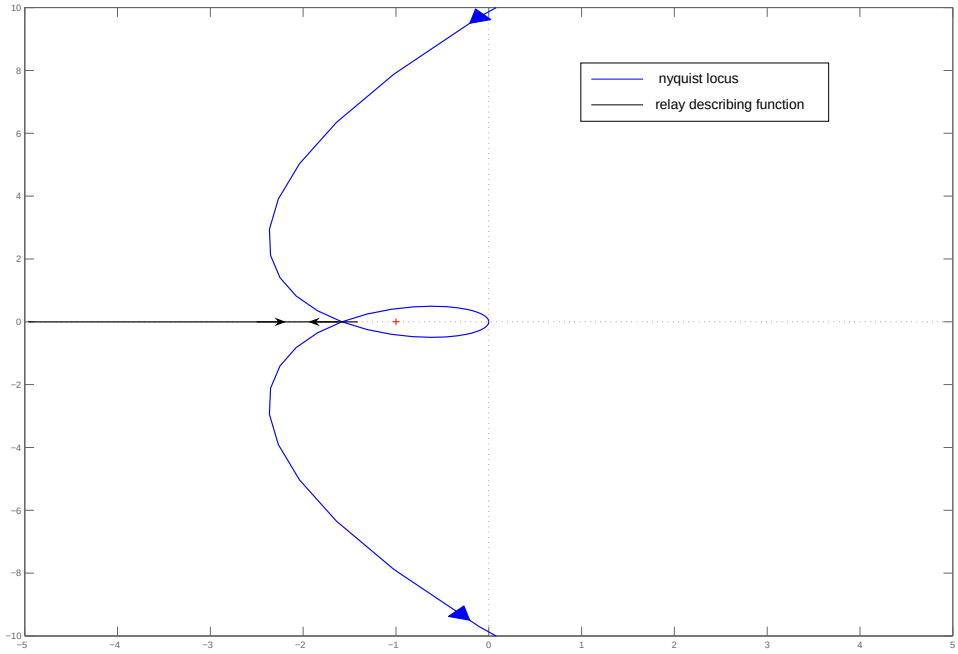

Figure 31: Zoomed-in Nyquist locus with possible relay describing-function

9.2.1 Proportional Gain

By decreasing the proportional gain, we can shift the -180^o point of the Nyquist locus towards the right and hopefully avoid the limit cycle as in Figure 32.

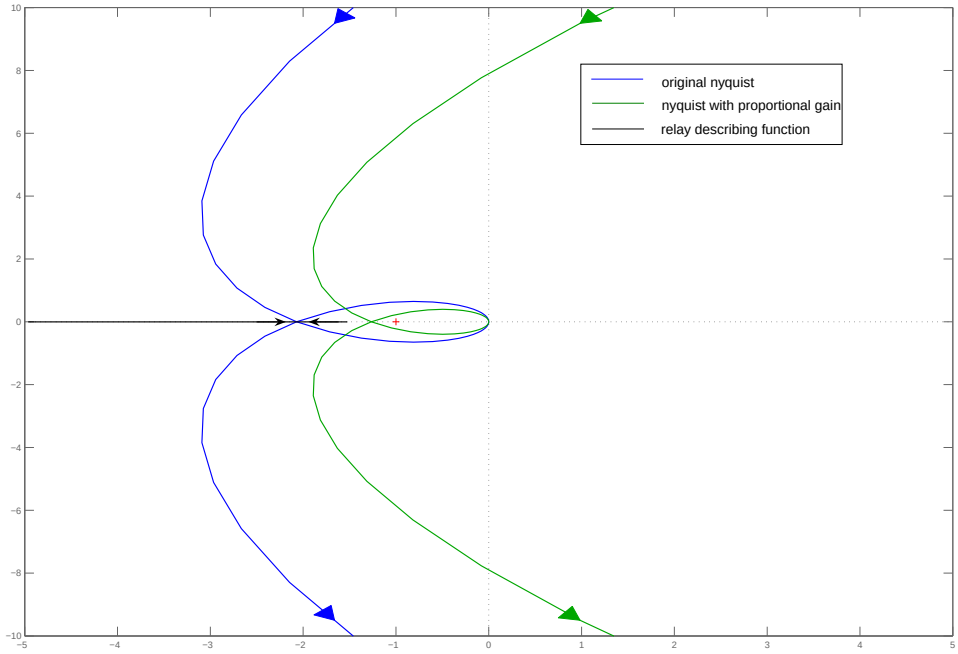

Figure 32: Nyquist with proportional gain

This is however somewhat hit and miss, not making full use of the data obtained via experimentation.

9.2.2 Phase lead

Using the fact that the limit cycle from the sensor data suggests a 0.5Hz limit cycle ($\sim 3.14\,rad\,s^{-1}$), a lead compensator of the form $\left(\frac{s+1}{s+12}\right)$ gives phase advance at both the limit cycle frequency determined by experiment and that determined by the model. Again, this hopefully results in the Nyquist locus avoiding the describing function.

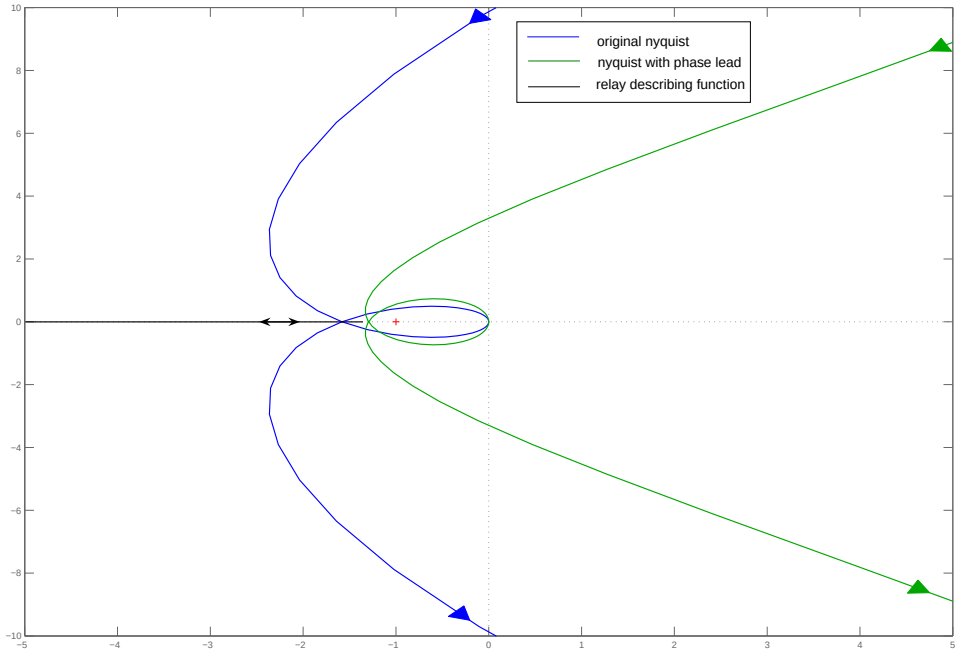

Figure 33: Nyquist with lead compensator

Furthermore, the amount of proportional gain or phase advance that can be supplied to resolve this problem is lower bounded by the fact that the Nyquist locus must encircle the -1 point due to the RHP pole in the plant.

Lastly, an alternative perspective to obtain a solution to this problem would be an attempt to shorten the describing function by physically altering its possible causes. For example, using a faster set of relays in the motor's H-bridge circuit, or decreasing the overall friction in the machinery are two possible solutions to this problem.

10 Conclusions

An understanding of the pitch-dynamics of the unicycle and the control problem involved in stabilising it about equilibrium, was gained.

Simulation results from Simulink showed that the system both with and without the compensator, was sensitive to initial angles up to 15^o in either direction, which is attributed to the linearisation losing validity. The system with the compensator had lower Gain and Phase Margins than without, which can be explained as the trade-off for better *Sensitivity* and *Complementary Sensitivity* characteristics - the response in the presence of noise and disturbances was improved.

Results from testing the unicycle and readings of sensor data showed the presence of limit cycle behaviour. This may have been the result of noisy sensor data, dynamic coupling from the bungee ropes used for testing, or from friction or other nonlinearities. It was decided that the limit cycle behaviour was most likely to be due to dead-band in the controller and describing function analysis was used to propose solutions. Reducing feedback, or the addition of extra phase lead around a frequency of 2Hz, were two controller-related solution proposals. A potential physical solution is to reduce the friction of the gear assembly. Sensor noise particularly in the IR range-finders may also be an issue, given that it is the highest contributor to the controller in terms of feedback gain. This could be replaced by an high precision inclinometer which has already been obtained, and can be relatively easily integrated. Another likely possibility is that the relays driving the motor either possess hysteresis or are too slow, and therefore a reduction in their hysteresis or replacing them with faster relays may resolve the dead-band encountered.

As a new testing strategy has been put forward and the bracket has been constructed, the next step is to re-attempt testing with the new controller propositions. A crane is not necessary for zero reference-velocity testing. A sturdy fixing on a ceiling or any point above the height of the unicycle that can withstand the loads described in Section 8.1, will also work. If the controller changes are unsuccessful, one of the physical solutions may be attempted.

If successful, the next major step would be to plan the lateral balancing scheme. This would require the 10kg disc to be attached to the unicycle and will alter its dynamics, resulting in the need to run through the steps outlined in this report once more for the pitch-balancing case. As there was a question as to whether the inertial motor would be capable of providing enough torque to overcome the inertial and frictional forces involved, it may be wise to conduct an experiment to verify this early in the project as it may be necessary to replace that motor. As with the wheel drive tachometer, problems may be encountered using the IR transceivers as a tachometer. If this is the case, the use of a rotary encoder may be investigated here as well.

A Q-learning

Q-learning [?] is one type of Reinforcement Learning technique with two distinct advantages - it can be used on-line and it does not require any knowledge of the system dynamics. It does however require discretisation of the states and inputs, which is prohibitive for more complex systems with contemporary computational speeds and flash memories. The key formula used in Q-learning is:

$$Q(state, action) \quad = \quad R(state, action) + \gamma.max[Q(next\,state, all\,actions)]$$

where R is a multi-dimensional matrix of rewards indexed by state and action, $\gamma$is a discounting parameter to describe how greedy the algorithm is, and Q is the multi-dimensional look-up table of predicted total rewards for taking a certain action in a certain state.

In the case of the unicycle system R can be set to 0 for all elements within a small range, say $+/-5^o$ and $R = -1$ outside this range, while Q may initially be a matrix of zeros. Q is then built up by exploring the discretised state space with (initially) random actions starting from an initial state in $R = 0$, and updating until a boundary is reached ($R = -1$). The system would then need to be reasserted into the initial *'good'* state space and another trial executed. The immediate issue with this in the case of the robotic unicycle is that such exploration needs to be catered for with specialised rigging and unless the experimentation process was automated, it would be extremely tedious and would get even more complicated if reference velocity tracking was required. This method would however be well suited to the 3rd year inverted pendulum project, where computer power is readily available and automation of the experiment would be relatively simple.

B Linearisation

A standard linearisation result [?] from considering a small perturbation $\delta\dot{x}$ of $\dot{x}$ from equlibrium is

$$\delta\dot{x} \approxeq A\delta x + B\delta u$$

where

$$A = \frac{\partial \underline{f}}{\partial \underline{x}}(x_e, u_e)$$
$$B = \frac{\partial \underline{f}}{\partial \underline{u}}(x_e, u_e)$$

So to linearise the pitch-dynamics equations, we use the above equations on the nonlinear pitch-dynamics equations, rearranged into standard state space form, $\dot{x} = f(x, u)$

From Section 3.1, we get

$$\ddot{x} = \left[\frac{1}{ac - b^2\cos^2\alpha}\right]\left[bc\dot{\alpha}^2\sin\alpha - \frac{b^2 g \sin 2\alpha}{2} + T\left(\frac{c - br_w\cos\alpha}{r_w}\right)\right]$$
$$\ddot{\alpha} = \left[\frac{1}{ac - b^2\cos^2\alpha}\right]\left[bga\sin\alpha - \frac{b^2\dot{\alpha}^2\sin 2\alpha}{2} + T\left(\frac{ar_w - b\cos\alpha}{r_w}\right)\right]$$

And trivially,

$$\dot{\alpha} = \dot{\alpha}$$

Therefore for $(x_e, u_e) = (0, 0)$, i.e. the upright equilibrium:

$$\frac{\partial f_1}{\partial x_1}(x_e, u_e) = \frac{\partial \ddot{x}}{\partial \dot{x}}(x_e, u_e) = 0$$
$$\frac{\partial f_1}{\partial x_2}(x_e, u_e) = \frac{\partial \ddot{x}}{\partial \dot{\alpha}}(x_e, u_e) = 0$$
$$\frac{\partial f_1}{\partial x_3}(x_e, u_e) = \frac{\partial \ddot{x}}{\partial \alpha}(x_e, u_e) = -\left(\frac{b^2 g}{ac - b^2}\right)\alpha$$
$$\frac{\partial f_2}{\partial x_1}(x_e, u_e) = \frac{\partial \ddot{\alpha}}{\partial \dot{x}}(x_e, u_e) = 0$$
$$\frac{\partial f_2}{\partial x_2}(x_e, u_e) = \frac{\partial \ddot{\alpha}}{\partial \dot{\alpha}}(x_e, u_e) = 0$$
$$\frac{\partial f_2}{\partial x_3}(x_e, u_e) = \frac{\partial \ddot{\alpha}}{\partial \alpha}(x_e, u_e) = \left(\frac{abg}{ac - b^2}\right)\alpha$$
$$\frac{\partial f_3}{\partial x_2}(x_e, u_e) = \frac{\partial \dot{\alpha}}{\partial \dot{\alpha}}(x_e, u_e) = 1$$

Hence, we end up with the result in Section 3.2, i.e.

$$\frac{d}{dt}\begin{bmatrix}\dot{x}\\ \dot{\alpha}\\ \alpha\end{bmatrix} = \begin{bmatrix}0 & 0 & \left(\frac{-b^2 g}{ac-b^2}\right)\\ 0 & 0 & \left(\frac{abg}{ac-b^2}\right)\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\dot{x}\\ \dot{\alpha}\\ \alpha\end{bmatrix} + \begin{bmatrix}\left(\frac{c-br_w}{r_w(ac-b^2)}\right)\\ \left(\frac{ar_w-b}{r_w(ac-b^2)}\right)\\ 0\end{bmatrix}\tau$$

Note: a, b and c are physical constants related to the unicycle and are defined in Section 3.1.

C CompactRio Fault

A solution was required to fix problems in connecting with the original CompactRio system. The only was to connect to the controller besides Ethernet, was via a DB9 cable, null-modem cable. Null-modem cables are nearly obsolete therefore one was soldered together using two standard DB9 cables and swapping the TX and RX lines as in Figure 34.

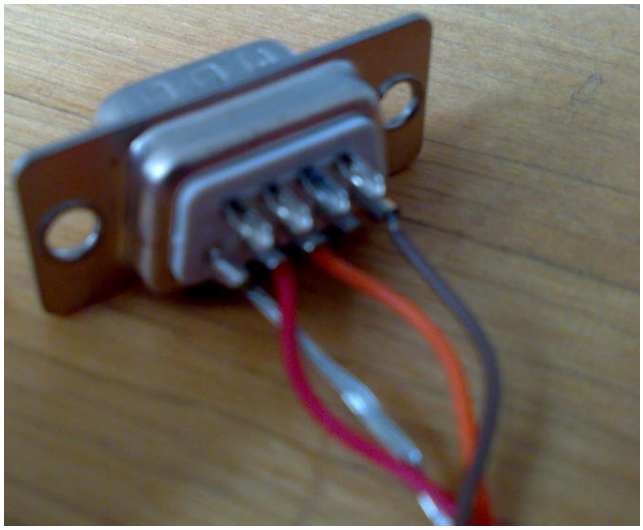

Figure 34: Null-modem DB9 connector

Hyperterminal was then used to connect to the controller (settings: 9600bps, 8-N-1) which was switched into Safe Mode. Figure 35 shows the output at the Hyperterminal window, confirming that the problem lay in the network device, i.e. the ethernet port was unusable.

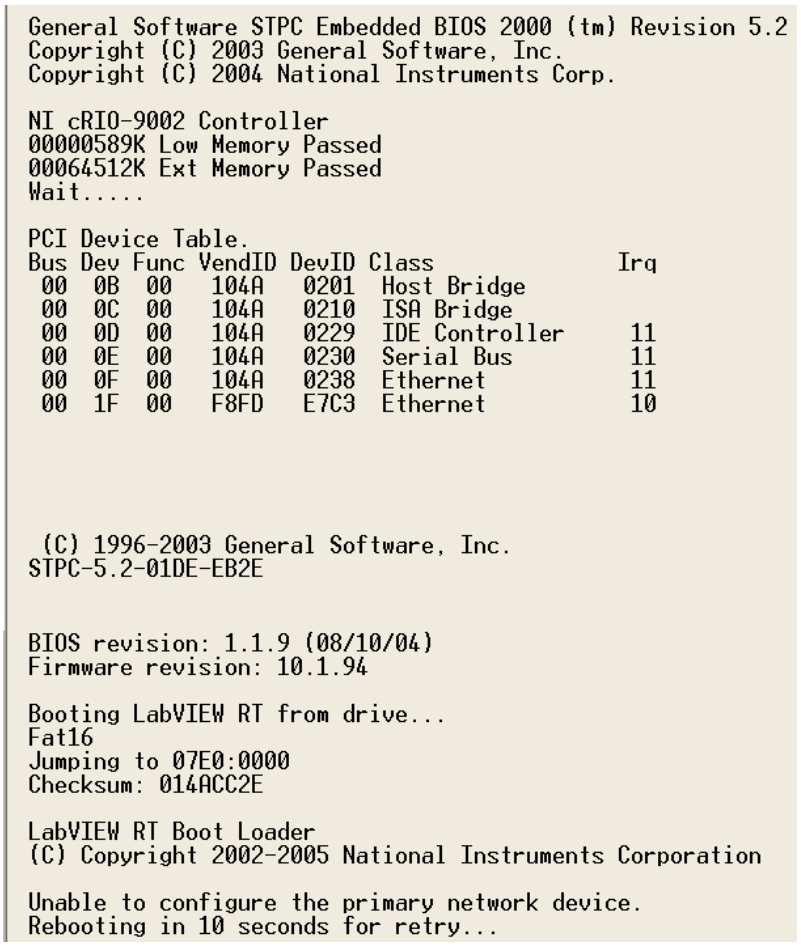

Figure 35: Hyperterminal Connection Window

The result of the diagnosis was that the controller was swapped for another CompactRio of the same make, and the problem with connectivity was thus resolved.

D Health and Safety

Care was taken to ensure that risks were minimised during the entire project. Steel-capped safety boots were worn during testing as per lab regulations. The forethought involved as demonstrated by the detailed test-rig design, resulted in an incident-free run of tests. Much of the remainder of the project was computer-based and thanks to a careful work schedule, the risk of RSI was minimal.