Precise Jumping using a Spider-like Dragline

Prepared by:

Michael Wootton

WTTMIC011

Department of Electrical Engineering
University of Cape Town

Prepared for:

Amir Patel

Department of Electrical Engineering
University of Cape Town

October 2015

Submitted to the Department of Electrical Engineering at the University of Cape Town in partial fulfilment of the academic requirements for a Bachelor of Science degree in Mechatronics Engineering

Key Words: Biomimetics, Mobile Robotics, Control, Arachnology

Declaration

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this final year project report from the work(s) of other people, has been attributed and has been cited and referenced.
3. This final year project report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof

Name:	**Michael Wootton**		
Signature:		**Date:**	14 October 2015

Acknowledgements

I would first like to thank my Supervisor Amir Patel for all his help and advice throughout the duration of this project.

I would also like to thank my mentor Stacey Shield for her suggestions on the project as well as for the loan of her Lego to build the robot

To Callen Fisher, for always having time to answer my questions and offer advice.

To all my friends in engineering, it was an honour to share this rollercoaster ride that is engineering with you and would do it all again if I could.

Lastly to my family, thank you for all your love and support I could not have done this without you.

Abstract

This paper investigates the use of a dragline to achieve precise landing for mobile robotics. The concept was inspired by the jumping spider which uses its silk as a pitch-righting and braking mechanism. Firstly, the dynamic model of the system was derived using Lagrangian mechanics. Then based on the findings by Chen, et al, the torque and speed requirements for the robot's motor were determined. In order to test the theory a platform consisting of a launcher and spider robot were built. Various control methods were investigated and then simulated. By analyzing these results, the best controller was then chosen and implemented on the testing platform. After a discussion and comparison of the results and simulations, a verdict was given as to the effectiveness of the system as a method of precise landing. Finally, recommendations were made for future areas of research and improvements.

Table of Contents

List of Figures

List of Illustrations

List of Tables

1. Introduction

1.1 Background to the study

The purpose of this project is to investigate whether a spider-like dragline can be used to achieve precise landings of a mobile robot. A recent study [1] into the aerial dynamics of jumping spiders suggested that the dragline could become a new method of controlling mobile robotics in the air. By modelling the spider dragline system a controller can be designed to emulated the effect that the dragline has on the spider. This system could be applied on any mobile robotic platform but could be especially useful in search-and-rescue situations where the environment is difficult to navigate.

1.2 Objectives of this study

The objectives of this study are to:

- Develop a dynamic model for the system
- Select an appropriate motor and spool system based on simulations
- Investigate suitable control techniques
- Design and implement the controller
- Build a robotic testing platform
- Analyse the data collected
- Draw conclusions based on the design and implementation of the controller
- Make recommendations based on the conclusions made.

1.2.1 Problems to be investigated

The aim of this project is to design, build and test an aerial robot which uses a spider-like dragline to control its landing distance and stabilize its body position in the air. To do this a dynamic model of the system must be developed. Based on this model, a suitable motor can be chosen. In the design of a controller to achieve the precise landing required, different control techniques will be investigated and their suitability for the task evaluated. Once the controller has been designed, digital simulations of the system will be done to determine the performance of the controller under varying initial conditions. A physical platform must then be built on which to implement the controller and test its real-world capabilities.

1.2.2 Purpose of the study

The purpose of this study is to gain insight into the feasibility of the dragline as form of aerial stabilization and distance control in mobile robotic platforms. Mobile robotics is a fast growing field of research with potential applications in many industries. For mobile robots to be effective they need to be able to have autonomous manoeuvrability in a variety of different terrains. Often nature provides the most efficient solutions to the problems faced in engineering and the spider-like dragline offers a one possible way which manoeuvrability in mobile robots can be improved.

1.3 Scope and Limitations

This project will investigated the effects of a dragline as means of controlling the distance travelled by an aerial robot. While an attempt will be made to stabilize the body angle of the robot, precise control of the body position on landing is beyond the scope of the project.

There are also time constraints that are imposed in that there is a project deadline that must be adhered to. Another limitations include the physical constraints such as input saturation and limited range of initial conditions.

1.4 Plan of development

This report can divided into three main parts. The first part gives an introduction to the report and relevant literature is reviewed and theory is developed that will form the basis of the following sections. The second part describes the design section of the project with an overview of the hardware and software used as well as the design of the controller. In the last section, the results from testing are put forward and followed by discussion of these results. Finally conclusions are made and recommendations for future work are given.

Chapter 2: Provides a review of relevant literature pertaining to the subject under investigation

Chapter 3: Describes the methodology used in carrying out the research performed during this project

Chapter 4: Explains how the modelling of the system was done as well as the assumptions made in simplifying the system.

Chapter 5: Describes the hardware design form building of the launcher to the selection of the individual components used.

Chapter 6: Gives an overview of the software used in the project

Chapter 7: Explains the process used to calibrate the gyroscope and accelerometer used in the design. Details on the method used to calibrate the motor are also given.

Chapter 8: Contains the design process used in designing the controller

Chapter 9: In this chapter the results of the testing are put forward

Chapter 10: The results from the previous chapter are discussed and compared to the simulated results.

Chapter 11: Conclusions are made based on the work done during the project

Chapter 12: Some recommendations for future work are suggested

2. Literature Review

2.1 Biomimetics

There are a number of definitions used to describe biomimetics. However, in the broadest sense it can be defined as learning from nature [2]. An important aspect is applying the knowledge gained, to design synthetic systems and improve technology. This can range from merely taking inspiration, to making the most exact copy possible.

Through evolution, nature has adapted to find lasting solutions to its many challenges. By studying and learning from these processes, it is possible that we will be able to improve current technologies and develop completely new ones in the future. Nature gives us the conviction and motivation to explore avenues that may otherwise have been dismissed as impossible.

Biomimetics has been applied in many fields including science, technology, medicine and robotics. Some of the most famous examples of bio-inspired technology are Velcro, self-cleaning paints and the airplane.

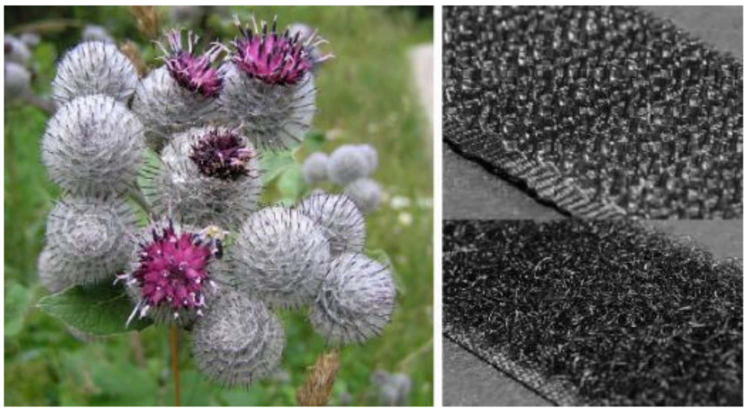


Figure 2.1: Image showing how Velcro has inspired by the Thistle plant [3]

Another example of biomimetics is the work done at FESTO where birds and animals are attempted to be replicated as closely as possible in a robotic form.

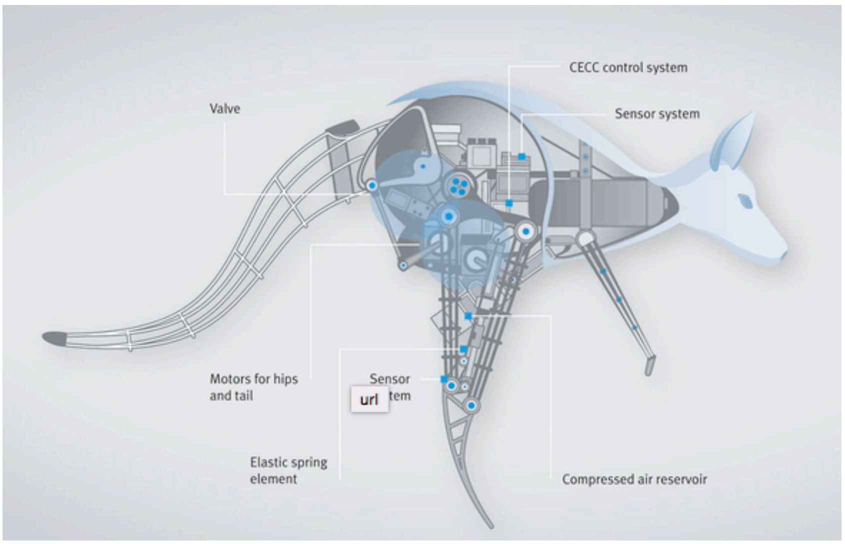


Figure 2.2: Example of the BionicKangaroo that has been developed by FESTO [4]

The potential future applications of bio-inspired design are almost endless. However, the one field which has seen a high growth in bio-inspired design is robotics.

Bioinspired design is a similar concept to biomimetics and are often used interchangeably when discussing designs inspired by nature. However, there is a subtle difference where bioinspired design takes only those parts from nature that will help improve the system. While biomimetics aims to emulate nature as closely as possible.

2.2 Aerial Righting in Nature

This section will investigate the techniques used by flightless animals to right themselves in the air due to either intentional or unintentional flights. Aerial righting is an important attribute for many animals to prevent injuries that could be obtained in accidental falls. The paper by Jusufi et al. gives a good overview of aerial righting techniques used by different animals. There are two main methods that animals use for aerial righting, inertial and aerodynamic.

In inertial righting, the movement of appendages relative to one another is often used to change the creature's angular momentum.

Falling cats have been observed to use the flexing and rotation of the spine and torso to reorient their body position in order land in a safe position. While rodents make use of the twisting of their bodies, in particular, analysis has shown that righting is often produced by rotations between the head and shoulders and shoulders and pelvis.

The use of tails is another way creatures can induce a change in angular momentum. In particular lizards and geckos have been researched into the use of their tails. Lizards have been observed to move to rotate their tails in the opposite direction to the rotation of their bodies.

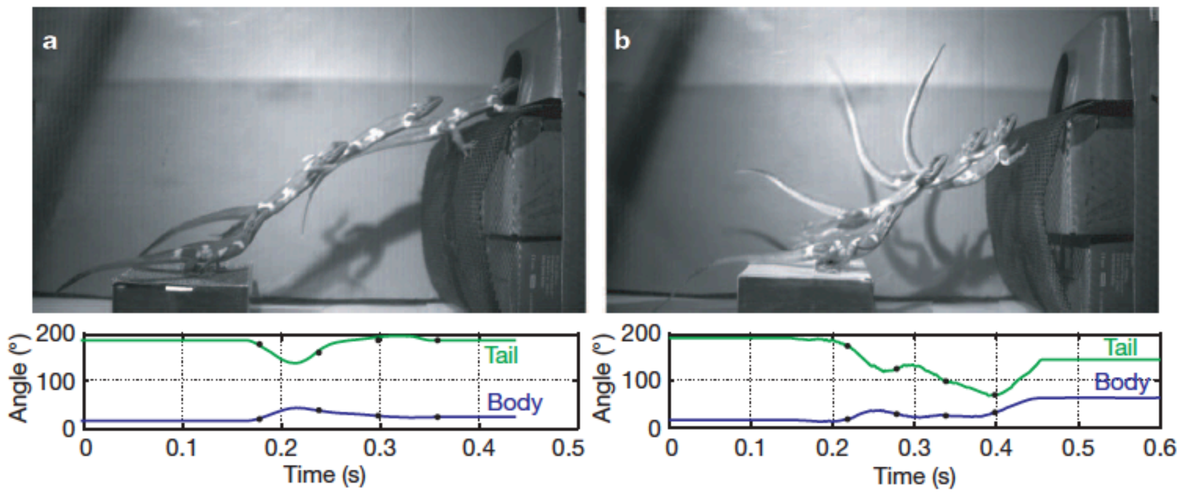


Figure 2.3: Image showing how a lizard uses its tail to control its body position [5]

In the figure above, the Agama lizard was made to jump from a small pedestal to a position of higher elevation. In the first picture the pedestal was covered in sandpaper which provided the lizard with good grip on take-off which resulted in minimal body and tail rotation. In the second picture the pedestal was changed to a smooth surface which caused the lizard experience some angular rotation on take-off. In this case the tail is moved in the opposite direction to the body rotation which results in the stabilization of the body in the air and a safe landing could still be achieved.

It has also been hypothesized the certain dinosaurs, such as the Velociraptor, could have used their tails as a form of stabilization and that their tails could have been even more effective than the lizards of today.

Another creature that makes use of inertial righting is the juvenile mantis. The juvenile mantis does not have any wings so relies on targeted jumps to move around its environment. The mantis makes use of its front and hind legs, trunk and abdomen to transfer the angular momentum to make a successful jump.

At take-off the front legs are extended forward while rotating anticlockwise. At the same time the hind legs are rotating clockwise about the trunk. In the next stage of the jump the front legs stop rotating and the angular momentum is transferred the abdomen which increases its anticlockwise rotation. In the third stage, the rotation of the hind legs is reversed to bring them into a position for landing. At this stage, the abdominal rotation decreases and front legs start to produce a clockwise rotation. In the final stage before landing, the hind legs and abdomen stop rotating which is counter balanced by a rapid anticlockwise rotation of the front legs. The result of this somewhat complex sequence of movements is that the mantis is able to position itself to make the perfect landing [6].

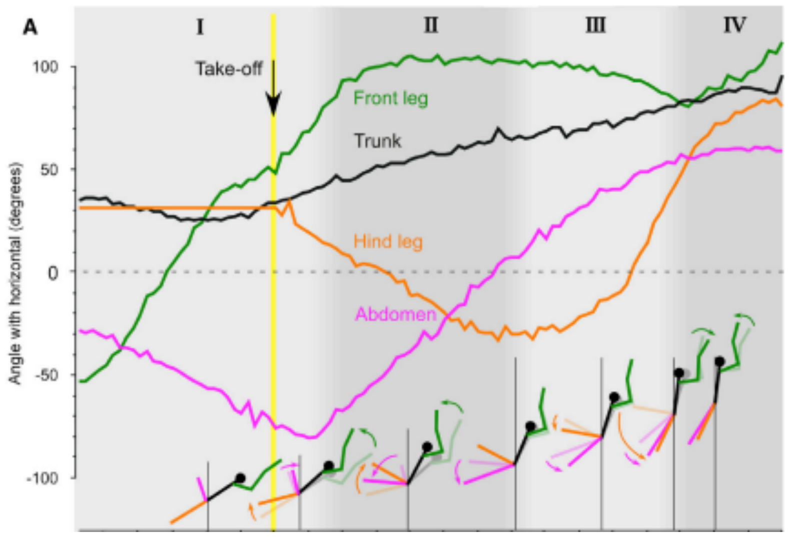


Figure 2.4: Graph showing how the position of the mantis' appendages move during a jump [6]

2.3 Pitch control with dragline

The research which inspired this project was done by Chen et al. This paper investigated the use of silk as means of jump stabilization in salticid spiders. This study was a breakthrough in that previously the dragline silk was only thought of as a safety line for the spider.

In order to examine the dynamics of dragline, high speed video was used to analyze the jumps of both silk and non-silk spiders. The instantaneous dragline force was also estimated along with the motion associated with the body of the spider. The test setup consisted of a lower platform and high platform with the spiders making a jump from the high platform to the one below.

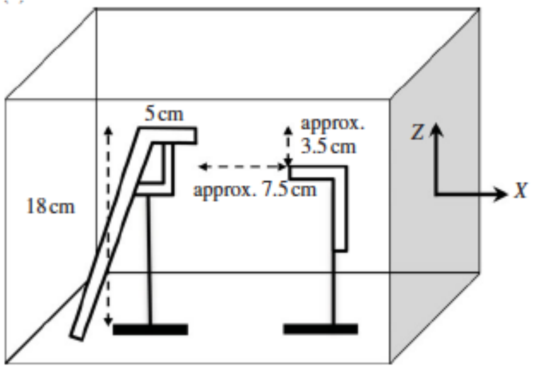


Figure 2.5: Image of test apparatus used in study by Chen et al. [1]

The results from the study showed some promising attributes of dragline usage that could be applied in mobile robotics. Firstly, the silk spiders were found to have decelerated more in the horizontal and spend a longer time in the air than those without silk. Also, the spiders' body position in the air was more stable with less over rotation and made smoother landings. The non-silk spiders were observed to have much greater rearward pitch which contributed to a body position that was upright on landing. This made it difficult for the spider to control its landing and the result was that time taken for the spider to come to a full stop once it hit the ground was longer than that of the spiders with silk. The deceleration that the silk provides, allows the silk spider to make a faster and more controlled landing with very little rotation of the body once contact with the ground is made.

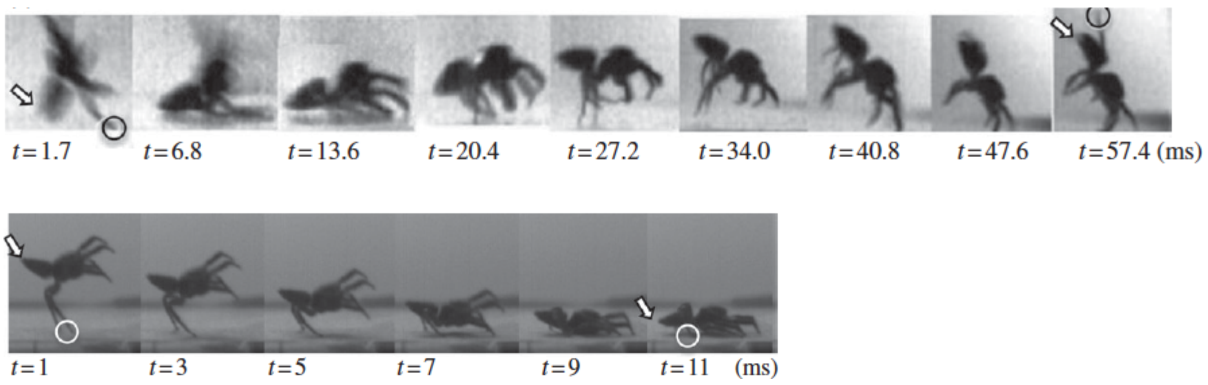


Figure 2.6: Showing the longer, more uncontrolled landings of those spider without silk compared to those with silk.[1]

In terms of body position while in the air, the dragline was also seen to have the advantage of stabilizing the body rotation of the spider.

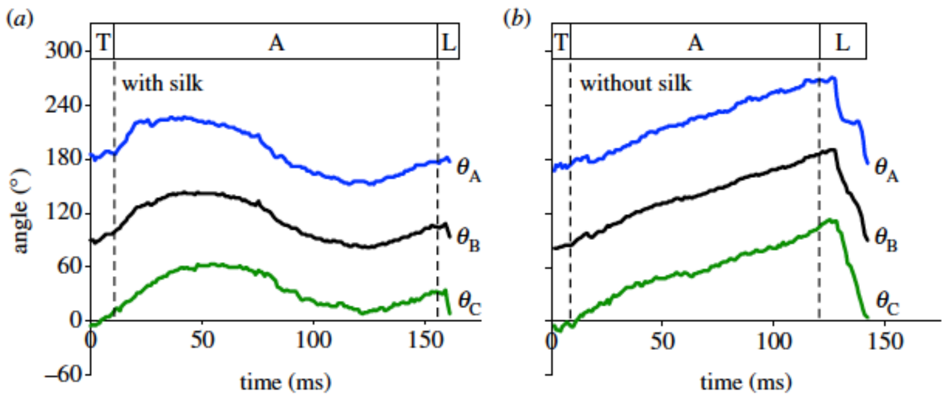


Figure 2.7: Graph of body angle vs. time for silk and non-silk spiders [1]

As can be seen from the figure above, spiders with silk were able reverse the direction of the rotation of their bodies. Whereas, those without continued to rotate at a relatively constant speed until they made contact with the ground. This meant that they were not necessarily in the optimal position to make a smooth landing when the time came. Silk spiders on the other hand, used the silk to ensure when it came to landing that the body position was correct. As it can be seen in figure above, the abdomen angle is approximately 180 deg or parallel to the ground at landing while the non-silk abdomen angle is closer to 250 deg.

The actual silk force required to produce this effect was estimated to be between 0.26 and 0.4 of the body weight of the spider[1].

This feature of the dragline is definitely applicable to aerial robotics where making a controlled landing is vital in reducing the possible damage that could occur from an awkward landing.

In response to this study by Chen et al., the topic was further investigated by Stacey Shield during her Undergraduate Thesis at UCT [7]. The focus of the research was in developing a dynamic model for the dragline system, as well as investigating the effects on the body position of an aerial robot due to a constant dragline force.

By determining a dynamic model for the system using Lagrangian mechanics, simulations were performed to evaluate the effects of the dragline under different initial conditions.

The results of these simulations followed a similar trend to that which was presented in the Chen et al. paper.

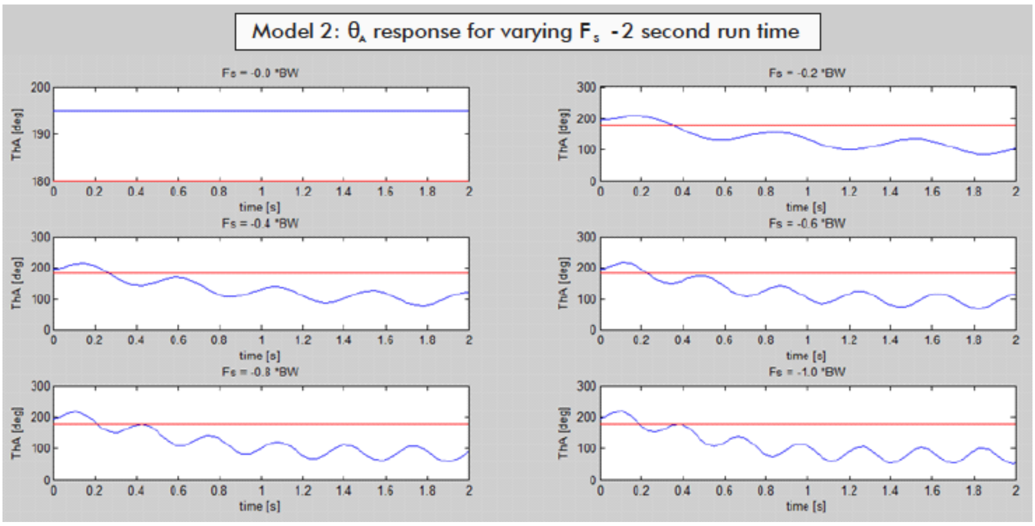


Figure 2.8: Showing the abdomen angle with varying magnitude of silk force [7].

A mobile robot testing platform was also developed to evaluate the simulated results on a physical platform. Tests were performed using a constant dragline force as designing a controller was beyond the scope of the project. From the tests it was concluded that a dragline force does have a stabilizing effect on the body positon of the robot. However, further research still needs to be done into whether it is possible to design an effective controller which can not only stabilize the body position but also assist in creating a tool to perform more precise landings.

2.4 Fully vs. Underactuated Robots

A fully actuated system is one can that produce an acceleration in a random direction. In contrast, an underactuatuated system is one that cannot produce an acceleration in any random direction. The consequence of this is that underactuated systems cannot be commanded to follow any particular trajectory. In mechanical systems, an underactuated system is one that has less actuators than degrees of freedom.

This constraint makes underactuated systems more complex to control as the designer does not have the freedom to pick and choose arbitrary solutions to the problem, but is in general bound in some way by the physical limitations of the system. This is especially true for nonlinear systems. In a fully actuated system, the nonlinear dynamics can be cancelled out using feedback. This turns the nonlinear system into a linear one and control theory for linear systems is quite extensive and well-documented. In the case of underactuated systems, it is not possible to fully cancel out the nonlinear behavior, although it will be shown that it is possible eliminate some of the nonlinear dynamics.

What is interesting about underactuated systems is that they can often be found in nature and so it is not surprising that these kind of problems need to be tackled when it comes bio-inspired robotics. For example, legged robots are underactuated [8] as the number of degrees of freedom will always be greater than the number of actuated joints of the system.

The spider dragline system is also underactuated as there is only one actuator which is the dragline force produced by the motor. The direction of the force is also limited, as it can only be used as a braking force, and cannot be used to input energy into the system, for example to propel the robot farther.

Some famous examples of underactuated systems are the Acrobot, and the cart and pendulum system. The Acrobot is a two joint planar inverted pendulum which has a motor at the elbow but is unactuated at the shoulder.

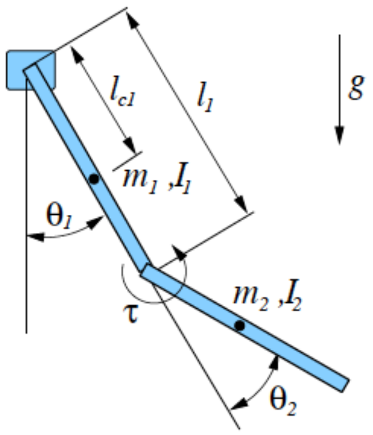


Figure 2.9: Image of the Acrobot coordinate system [9]

The main control problem is to ensure that the Acrobot is stabilized in the vertically upright position. The techniques used to control these systems are also applicable to the spider dragline system as not only is the system also underactuated, but it is also modelled as two links with torque applied at the elbow. One technique that is used to control the system is Partial Feedback Linearization. The details of this method are described in greater depth in the Theory Development chapter, nevertheless, it involves the cancelling out of the nonlinear dynamics of the system so that linear control techniques can be used. Partial Feedback Linearization was pioneered in a paper by Spong [10] on similar system to the Acrobot.

3. Theory Development

3.1 Lagrange Dynamics

Lagrangian mechanics presents a way of defining a system based on the energy of system. This is in contrast to Newtonian mechanics, which relies on knowledge of the forces in system.

$$\mathcal{L} = T - V$$

The equation above is what is known as the Lagrangian equation [11], where T and V are the total and potential energy of the entire system respectively. The Lagrangian approach has the benefit of simplifying the process of determining the equations of motion for complex systems. The kinetic and potential energies of a multibody system are calculated as follows:

$$T = \sum_{i=1}^{N} T_i$$
$$V = \sum_{i=1}^{N} V_i$$

where, N is the number of rigid bodies and T_i and V_i being the kinetic and potential energy of the respective bodies.

For Lagrangian mechanics to be applied, the system must meet three conditions. These are independence, completeness and holonomicity [11].

For a system to be:
1) Independent: all but one of the system coordinates are fixed, the free coordinate can still achieve a continuous range of movement.
2) Complete: all parts may be located at all times.
3) Holonomic: the number of degrees of freedom of the system equals the number of coordinates needed to completely describe the motion of the system.

The generalized coordinates that satisfy the above conditions are given by q_j

$$q_j = generalized\ coordinates$$

As Lagrange is an energy based approach only non-conservative forces need to be accounted for as it is these forces which add or remove energy from the system. The generalized forces of the system are denoted by Q_j [11].

$$Q_j = generalized\ forces$$

where j is the number of degrees of freedom or equations of motion of the system.

The equations of motion of a system can be determined from the following equation

$$\frac{d}{dt}\left(\frac{\delta \mathcal{L}}{\delta \dot{q}_j}\right) - \frac{\delta \mathcal{L}}{\delta q_j} = Q_j$$
$$\frac{d}{dt}\left(\frac{\delta T}{\delta \dot{q}_j}\right) - \frac{d}{dt}\left(\frac{\delta V}{\delta \dot{q}_j}\right) - \frac{\delta T}{\delta q_j} + \frac{\delta V}{\delta q_j} = Q_j$$

For the spider system, the generalized force that will be considered is the force on the dragline.

3.2 Euler Angles

The IMU sensor is fixed to the body of the robot, this means that all readings are in reference to the body frame of the robot. However, to use the readings to do Newtonian calculations on the body the readings need to be in reference to an inertial frame [12]. The Euler angle technique rotates the gyroscopic readings from the body frame to the inertial frame.

In order to perform the rotation three angles are used

$$\phi = roll\ angle$$
$$\theta = pitch\ angle$$
$$\psi = yaw\ angle$$

where roll is typically rotation around the x-axis, pitch is rotation around the y-axis and yaw is rotation around the z-axis.

The most commonly used sequence is Euler 3-2-1 which first consists of a rotation firstly in the z-axis then in the y-axis then in the x-axis. The matrix representation is shown below [12][13][14].

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = \begin{bmatrix} cos\theta cos\psi & cos\theta sin\psi & -sin\theta \\ sin\phi sin\theta cos\psi - cos\phi sin\psi & sin\phi sin\theta sin\psi + cos\phi cos\psi & sin\phi cos\theta \\ cos\phi sin\theta cos\psi + sin\phi sin\psi & cos\phi sin\theta sin\psi - sin\phi cos\psi & cos\phi cos\theta \end{bmatrix} \begin{pmatrix} X_I \\ Y_I \\ Z_I \end{pmatrix}$$

The angles are obtained from previous system states. If the matrix is invertible then a matrix mapping the body reference frame to the inertial frame can be obtained.

The angular rates, which are measured with respect to the body reference frame, can be converted to the inertial reference frame using the following matrix [12]

$$\begin{pmatrix} \dot{\phi}_I \\ \dot{\theta}_I \\ \dot{\psi}_I \end{pmatrix} = \begin{bmatrix} 1 & sin\phi tan\theta & cos\phi tan\theta \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi sec\theta & cos\phi sec\theta \end{bmatrix} \begin{pmatrix} \dot{\phi}_B \\ \dot{\theta}_B \\ \dot{\psi}_B \end{pmatrix}$$

The main disadvantage of Euler 3-2-1 is that for pitch angles of ± 90 degrees a singularity occurs which can cause the implementation to fail [12][14]. As long the system doesn't pass through these angles the mapping will be give accurate results. If the system does pass through these angles then one way to overcome this problem is to use Euler 2-3-1 which produces the same result as Euler 3-2-1 except that singularity doesn't occur at a pitch angle of ± 90 degrees.

3.3 Linearization Techniques

As the system dynamics are nonlinear most linear control techniques can not be applied. There are methods which can be used to linearize the system, the first of these that was used was Taylor series linearization.

3.3.1 Taylor Series Linearization

The Taylor series is powerful tool was states that any linear or nonlinear function that is continuous and has n derivatives that are also continuous can linearized about the point $x, u = a, v$ [15]. The expansion is as follows.

$$f(x,u) \approx f(a,v) + f_x'(x-a) + f_u'(u-v) + \frac{1}{2!}f_x''(x-a) + \frac{1}{2!}f_u''(u-v) + \frac{1}{3!}f_x'''(x-a) + \frac{1}{3!}f_u'''(u-v) + \cdots$$

A linear approximation of a system at a point can be made if the higher exponential terms are equated to zero. Therefore a system given by x = f(x,u), y = g(x,u) where x is a system of differential equations and y is the output can be linearly approximated as follows [15].

$$f_x' = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} = \mathrm{A}$$

$$f_u' = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} & \cdots & \frac{\partial f_1}{\partial u_n} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} & \cdots & \frac{\partial f_2}{\partial u_n} \\ \frac{\partial f_n}{\partial u_1} & \frac{\partial f_n}{\partial u_2} & \cdots & \frac{\partial f_n}{\partial u_n} \end{bmatrix} = \mathrm{B}$$

$$g_x' = \begin{bmatrix} \frac{\partial g_1}{\partial x_1} & \frac{\partial g_1}{\partial x_2} & \cdots & \frac{\partial g_1}{\partial x_n} \\ \frac{\partial g_2}{\partial x_1} & \frac{\partial g_2}{\partial x_2} & \cdots & \frac{\partial g_2}{\partial x_n} \\ \frac{\partial g_n}{\partial x_1} & \frac{\partial g_n}{\partial x_2} & \cdots & \frac{\partial g_n}{\partial x_n} \end{bmatrix} = \mathrm{C}$$

$$g_u' = \begin{bmatrix} \frac{\partial g_1}{\partial u_1} & \frac{\partial g_1}{\partial u_2} & \cdots & \frac{\partial g_1}{\partial u_n} \\ \frac{\partial g_2}{\partial u_1} & \frac{\partial g_2}{\partial u_2} & \cdots & \frac{\partial g_2}{\partial u_n} \\ \frac{\partial g_n}{\partial u_1} & \frac{\partial g_n}{\partial u_2} & \cdots & \frac{\partial g_n}{\partial u_n} \end{bmatrix} = \mathrm{D}$$

where A,B,C and D are the state space matrices for the approximated system.

$$\frac{d}{dt}x(t) = Ax(t) + Bu(t)$$
$$y = Cx(t) + Du(t)$$

The main disadvantage with Taylor series approximation is that the system dynamics are only linearized around a single point. While this can often be used to approximate other points close by, if dynamics of the system significantly change then the approximation often becomes inaccurate [16]. Therefore this technique is best suited to systems which have a fairly restricted area of operation. In the spider dragline system both the initial conditions and the the amount dragline force applied can drastically change the dynamics. Attempts to linearize the system about one or even numerous points proved unsuccessful as once the any of the aforementioned variables were changed the approximation became extremely inaccurate.

One method that could be used to overcome this problem is a model predictive approach, where next states could be predicted using the nonlinear model and then using these predictions the linear model could be continually updated for future values. This also has complications brought about by the linear model constantly being adjusted. In order to deal with this issue a gain scheduling approach would have to employed. This was felt to be an overly complicated solution to the problem so an alternative approach was then investigated.

3.4 Partial Feedback Linearization

Fully actuated nonlinear systems can be linearized using Input-State or Input-Output Feedback linearization. The concept behind feedback linearization is to design a controller which effectively cancels out the nonlinear dynamics of the system. This creates what is referred to as a pseudo-linearized system on which linear control techniques such as state space pole placement can be used to produce the desired response. It should be noted the requirements of feedback linearization are that all states must be measurable, either directly or by a state observer. Also accurate knowledge of nonlinearities in the system is assumed, without this closed loop performance can suffer [15].

Underactuated systems cannot be fully feedback linearized. An underactuacted system has one or more states whose dynamics are not dependent on the input. These are called passive states. Those states that can be controlled by the input are actuated or active states. Feedback relies on the input cancelling the nonlinear behaviour of the states. However, if there are states which do not depend on or can be directly controlled by the input, then linearization is not possible. However, it is still possible to linearize some of the states and this is achieved by using partial feedback linearization (PFL).

The general form of PFL is called task space PFL. In task space PFL a function of both passive and active joints can be feedback linearized. The general case can be split into two further cases, the first is a collocated partial feedback which linearizes the dynamics of the actuated joints. The second is a non-collocated partial feedback, this linearizes the dynamics of the unactuated joints. The general process for applying PFL is detailed next [17][9].

For a given underactuated system defined in the form of the manipulator equation as shown below,

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = Bu$$

where,

$$H(q) = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

the above equation can be split up in terms of active and passive joints,

$$H_{11}\ddot{q}_1 + H_{12}\ddot{q}_2 + \emptyset_1 = 0$$

$$H_{21}\ddot{q}_1 + H_{22}\ddot{q}_2 + \emptyset_2 = \tau$$

where $\emptyset_1$ and $\emptyset_2$ consists of the all the Coriolis and gravitational terms, $\ddot{q}_1$ represents the accelerations of the passive joints, $\ddot{q}_2$ represents the accelerations of the active joints.
For task space PFL given the following arbitrary function,

$$y = f(q)$$

where y is a function of both passive and active joints.

We define

$$J_1 = \frac{\delta f}{\delta q_1}, \qquad J_2 = \frac{\delta f}{\delta q_2}$$

$$J = [J_1\, J_2]$$

If actuated joints are commanded such that

$$\ddot{q}_2 = \bar{J}^{+}[v - \dot{J}\dot{q} + J_1 H_{11}^{-1}\emptyset_1]$$

where

$$\bar{J} = J_2 - J_1 H_{11}^{-1} H_{12}$$

and

$$\bar{J}^{+}$$

is the pseudo inverse

Then it can be shown that

$$\ddot{y} = v$$

A suitable controller could then take the form of

$$v = \ddot{y}^d + K_d(\dot{y}^d - \dot{y}) + K_p(y^d - y)$$

The gains K_d and K_p can be designed using state space pole placement.

If $Z_1 = y$ and $Z_2 = \dot{y}$

Then the linearized state space system becomes

$$\frac{d}{dt}\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} Z_1 \\ Z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v$$

From this pole placement can be performed using

$$|SI - A + bk^T| = \phi_c$$

where ϕ_c is the desired characteristic equation.

4. Methodology

In this section the framework will be laid for how the research and design was completed during this project. It will also give an overview of the processes and techniques used in meeting the stated objectives of the study.

4.1 Modelling and simulation

The aim of this project is to design and implement a controller to allow a mobile robotic platform perform a precise landing. To do this the dynamics of the system need to be fully understood. This was done by developing the dynamic model of the system using the Lagrangian method detailed in the previous chapter. A model of the system had already been developed in however these results were recalculated for the sake of verification and understanding. All simulations were done using MATLAB and Simulink.

Using this model, simulations were done in order choose the most appropriate motor based on torque requirements. The torque parameters for the motor were based on the findings by Chen et al .

4.2 Hardware integration and Calibration

The next stage of the project was to design and integrate the hardware platform that will be used for testing. It was decided that the hardware build was to be done early on in the project timeline to ensure that it would be ready for testing once the controller design had been completed.

Firstly, the chassis to house the motor and electronics will be designed and built using Perspex and Lego. These materials are particularly suited to the design due to their robustness and ease of assembly. The spool for the motor was designed on SolidWorks and 3D printed.

Then the components such as the iNEMO, encoder and XBee Transceivers were tested and communication interfaces were setup. During this phase the calibration of the on-board IMU sensors, accelerometer and gyroscope was performed.

Motor tests and calibration will be performed in order to determine relationship between input voltage and corresponding output torque.

4.3 Controller Design and simulation

Once the hardware and software is complete, the task of the designing a suitable controller will be tackled. This will be divided into three stages, linearization of the system, research into appropriate control techniques and the design, simulation and evaluation of controllers designed. The final choice of controller will be based on performance criteria such settling time and overshoot while attempting to minimize the control action. The robustness of the controller to variation in parameters such as inertia and body mass will be also be evaluated.

4.4 Testing and Verification

The best controller of those simulated will then be implemented. For the testing of the system a launcher will be designed in order to test the system under varying initial conditions. The testing process will consist of performing numerous launchers for a range of setpoints and initial velocities. The data is sent to a data logging application which will then be used to compare to the simulated results. The jumps will also be filmed to give a visual insight into the flight trajectory and body position of the spider robot.

Once testing is complete the data will analysed and compared. Conclusions will then be drawn on the results and recommendations made on future improvements.

5. Modelling and Simulation

In section the modelling of the dynamics of the system will be modelled using the Lagrangian method described in Theory Development section of this report. [18] Accurately modelling the system is important for the implementation of an effective controller. The model developed in this section will be used extensively the following chapters.

5.1 Defining the System

The system under consideration consists of three main parts, the substrate, the dragline, and the robot body. The substrate is the launcher, which is fixed in place, and also provides the point to which the dragline is secured. The dragline is connected from the substrate to spool on the robot body. As the dragline is in tension during the aerial phase it was modelled as a massless, rigid body. The robot body houses all the electronic components and spool. In order to simplify the modelling of the system, the body was reduced to a rectangular box with uniformly distributed weight whose moment of inertia is given by,

$$I = \frac{12}{m}(w^2 + l^2)$$

where, m is the mass of the robot body, w is the width of the body and l is the length.

5.2 Assumptions and Simplifications

In modelling the system various assumptions were made which were necessary in simplifying the analysis of the system dynamics. Firstly, the effects of the spiders' legs during the flight are ignored. While the spider does splay its limbs during its jump, there is not excessive motion which could cause a change in its angular momentum. Therefore, it can be assumed that the legs are only used to effect the aerodynamic drag force during the jump [1][7].

5.3 Equations of Motion

The first step was to determine a generalised coordinate system which satisfies the requirements of Lagrangian mechanics. The coordinates chosen were,

θ_0 - the angle between the dragline and the horizontal
θ_s- the angle between dragline and the center line of the body
L - the length of the dragline

These coordinates satisfy the conditions for completeness, independence and holonomicity. They were also chosen as they provide the relevant information about the state of the system that can be used in controller design.

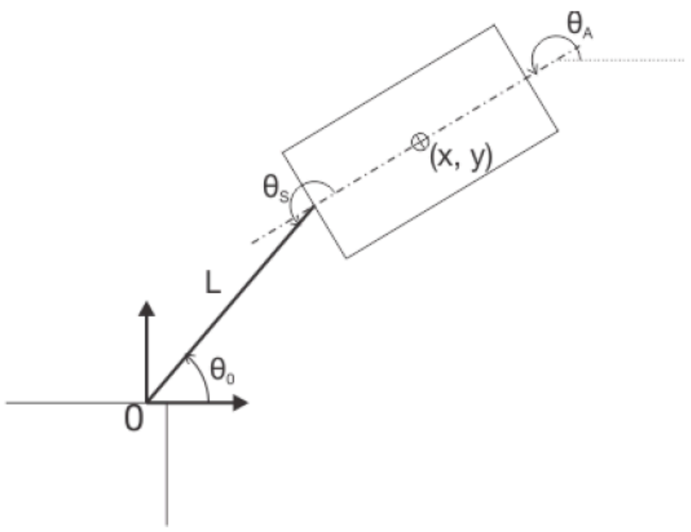


Figure 5.1: Coordinates for system [7]

Due to the high chance of computational error in performing the Lagrangian calculations by hand, MATLAB's Symbolic Toolbox was used to derive the equations of motion of the system.

The three equations of motion for the system were computed as follows.

In the θ_s direction:

$$I(\ddot{\theta}_0 + \ddot{\theta}_s) - \frac{1}{4}ml^2(\ddot{\theta}_0 - \ddot{\theta}_s) + \frac{1}{2}mlLsin(\theta_s) + \frac{1}{2}mglcos(\theta_0 - \theta_s) - \frac{1}{2}mlL\dot{\theta}_0^2\sin(\theta_s) + \frac{1}{2}lmL\ddot{\theta}_0\cos(\theta_s) + ml\dot{L}\dot{\theta}_0cos(\theta_s) = 0$$

In the θ_0 direction

$$I(\ddot{\theta}_0 + \ddot{\theta}_s) + \frac{1}{4}ml^2(\ddot{\theta}_0 - \ddot{\theta}_s) + mL^2\ddot{\theta}_0 + \frac{1}{2}mglcos(\theta_0) - \frac{1}{2}ml\ddot{L}sin(\theta_s) - \frac{1}{2}mglcos(\theta_0 - \theta_s) + 2mL\dot{L}\dot{\theta}_0 - \frac{1}{2}mlL\dot{\theta}_s^2\sin(\theta_s) - mlL\ddot{\theta}_0\cos(\theta_s) + \frac{1}{2}mlL\theta_s\,\ddot{\cos}(\theta_s) - ml\dot{L}\dot{\theta}_0\cos(\theta_s) + mlL\dot{\theta}_s\dot{\theta}_0\sin(\theta_s) = 0$$

In the L direction

$$mg\sin(\theta_s) + \ddot{m}L - mL\dot{\theta}_0^2 + \tfrac{1}{2}l\cos(\theta_s)\left(\dot{\theta}_0^2 + \dot{\theta}_s^2\right) - \tfrac{1}{2}lsin(\theta_s)(\ddot{\theta}_s - \ddot{\theta}_0) - l\dot{\theta}_0\dot{\theta}_s\cos(\theta_s) = -F_s$$

5.4 Simulations of the Model

To simulate the system, the equations of motion were rearranged into the manipulator form which is given by,

$$H(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = B$$

where, H(q) is mass matrix and contains the coefficients of the acceleration terms, $C(q,\dot{q})$ is made up of the Coriolis terms, G(q) contains the gravitational terms and B represents the generalized forces. Rearranging the equation allows the system to be represented in terms of the accelerations of the states,

$$\ddot{q} = \left(B - C(q,\dot{q})\dot{q} - G(q)\right)/H(q)$$

6. Hardware

In this chapter an overview of the components used in the project will be given as well as the thought process behind their selection.

6.1 Choosing the Motor

6.1.1 Issues encountered in previous testing

The previous year's testing found that use of geared motors was not satisfactory. This was mainly due to friction of the gears preventing the spool from spinning freely. This made precise control of the motor difficult.

6.1.2 Types of DC Motors (brushed vs brushless)

There were two main types of gearless DC motors that were considered, brushed and brushless. Firstly, brushless motors were considered. The main advantage that brushless motors have over brushed, is their high torque capabilities relative to their weight and size. Initial research into the viability of brushless motors was promising, with several micro-brushless motors being singled out due to their light weight and high torque. Brushless motors operate by using an AC current to activate poles. The magnetic field causes the rotor to align itself with the poles that are activated. By switching the poles which are activated, the rotor will rotate according. This means that brushless motors require an electronic speed controller (ESC) which converts the DC supply voltage to an AC current to drive the motor. However, the controller uses a very crude method of control which doesn't allow for the precise control of motor speed that is needed in this project. In order to use the brushless motor, the entire motor controller system would have to be designed using hall sensors and switching circuits to control the sequence in which the stator poles are activated. This was deemed to be outside the scope of this project, as the time required to develop a suitable control system would compromise the original objectives of this project.

This meant a brushed DC motor would be used. The advantage of a brushed motor is that they require a much simpler control circuit; for unidirectional speed control only one switching element is needed. The speed of a brushed motor is proportional to the voltage across its terminals. Increasing the terminal voltage increases the magnetic field strength, which in turn causes the rotor spin at a higher speed. The one disadvantage with gearless brushed motors in particular, is that the torque produced is usually much lower than a brushless of a similar weight.

6.1.3 Torque Requirements and Design Variables

In order to choose a suitable motor the load torque requirements needed to be determined. The motor needs to be able to provide enough torque to ensure that the required dragline force can be produced. Torque is determined by a force acting at a certain perpendicular distance from the point of interest.

$$T = F \cdot r$$

From studies done by Chen et al., it was found that the force on the dragline for jumping spiders varied between 0.2 and 0.4 of the spider's body weight [1]. Therefore, to calculate the force, and thus torque required by the motor, the bodyweight of the robot needs to be determined.

Table 6.1: Determining the mass of the robot

Component	Weight (g)
Battery	21
Motor	30
Chassis	60
iNEMO	15
Circuitry	10
Encoder	1
XBees	5
Total	143

From the table 6.1 above it is clear that the total mass of the robot is approximately 143g. Of course the mass of each individual motor may vary slightly but to simplify calculations a set mass of 30 grams was used.
The weight of the robot is then calculated as:

$$W = mg = 0{,}143 \times 9{,}81 = 1{,}4\ N$$

where, m is mass and g is gravitational acceleration.
From the weight it is now possible to calculate the maximum dragline force, this will be taken as 0,4 of the bodyweight but incorporate a safety factor it was decided to use 0,5 of the bodyweight. This gives a maximum force of:

$$F_{max} = 0{,}5 \times 1{,}4 = 0{,}7\ N.$$

Through research of the commercially available motors, a shortlist of three potentially suitable motors was made. The specifications of these motors is shown below.

Table 6.2: Comparison of Motor Specifications for the three motor being considered

Motor Name	Stall Torque [Nm]	No Load Speed [rpm]	No Load Speed [rad/s]
SRE-260W-2295	0,01494072	18500	1936,33
SRE-280W-2856-42	0,012258313	9000	942,00
SRE-260W-19100-38	0,010787315	12600	1318,80

The torque-speed relationship of the three motors is shown below.

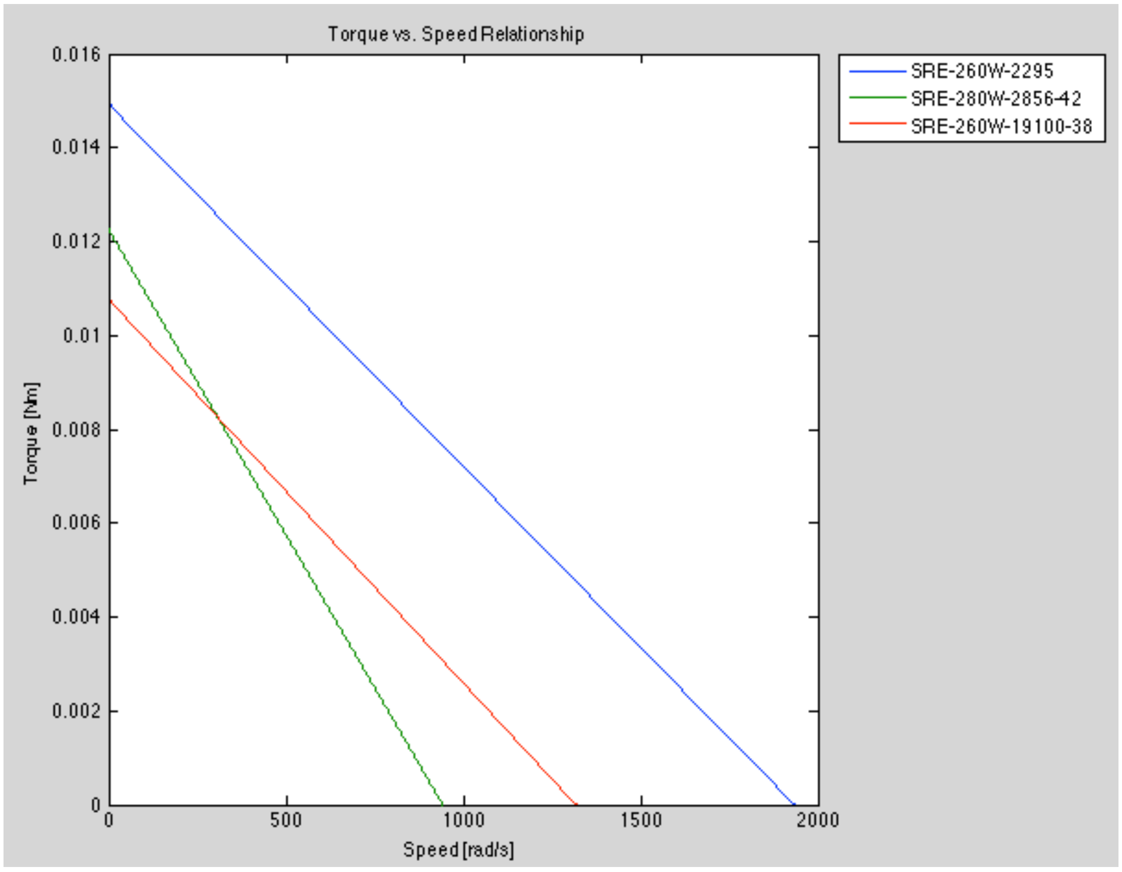


Figure 6.1: Torque vs Speed graphs

From the torque vs speed graphs, it can be seen that as speed increases, the amount of torque that can be produced decreases, and vice versa. Therefore, an important criteria is that the motor will be able to produce required torque at the maximum speed. The maximum initial velocity that will be tested at is 8 m/s. From the torque vs. speed graphs of these motors it was clear that they would not be able to produce the required dragline force at this speed. To overcome this problem it was decided to incorporate the radius of the spool as a design variable.

From the torque vs. speed graph the equation relating torque and speed is given by:

$$T = \left(\frac{\omega_{NL}}{T_{stall}}\right)\omega + T_{stall}$$

$$T = \left(\frac{\omega_{NL}}{T_{stall}}\right)\left(\frac{v}{r}\right) + T_{stall}$$

And as F = T/r

$$F = \left(\frac{\omega_{NL}}{T_{stall}}\right)\left(\frac{v}{r^2}\right) + \frac{T_{stall}}{r}$$

By keeping the velocity constant, a graph of force vs. radius can be plotted. The graph of force vs. spool radius at the maximum test velocity for the three motors under consideration is shown below.

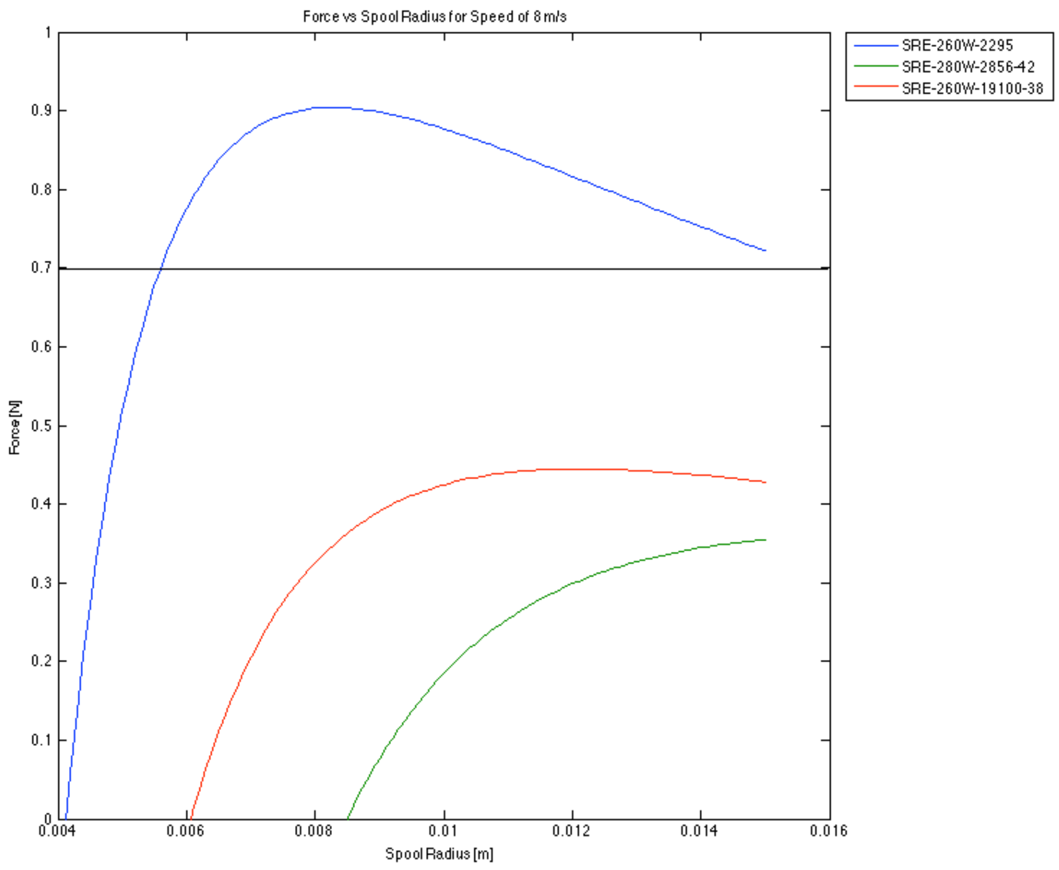


Figure 6.2: Force vs Radius for different motors at a velocity of 8 m/s

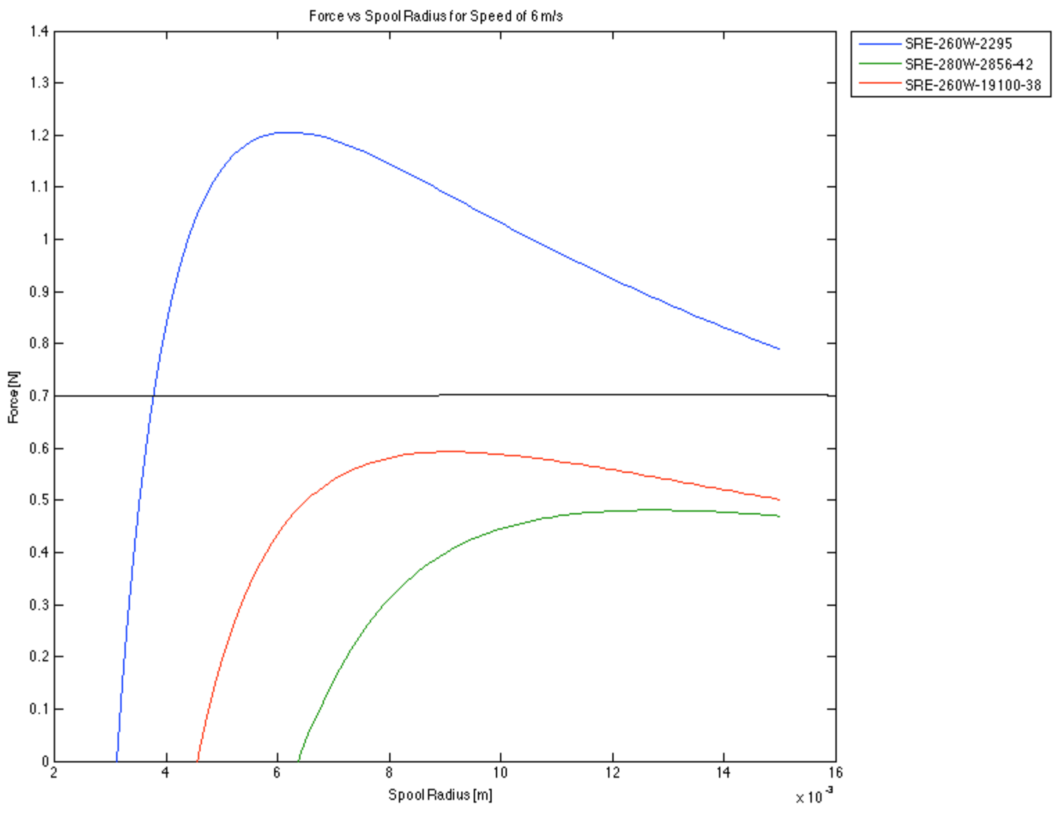


Figure 6.3: Graph showing force vs. spool radius relationship at a velocity of 6 m/s

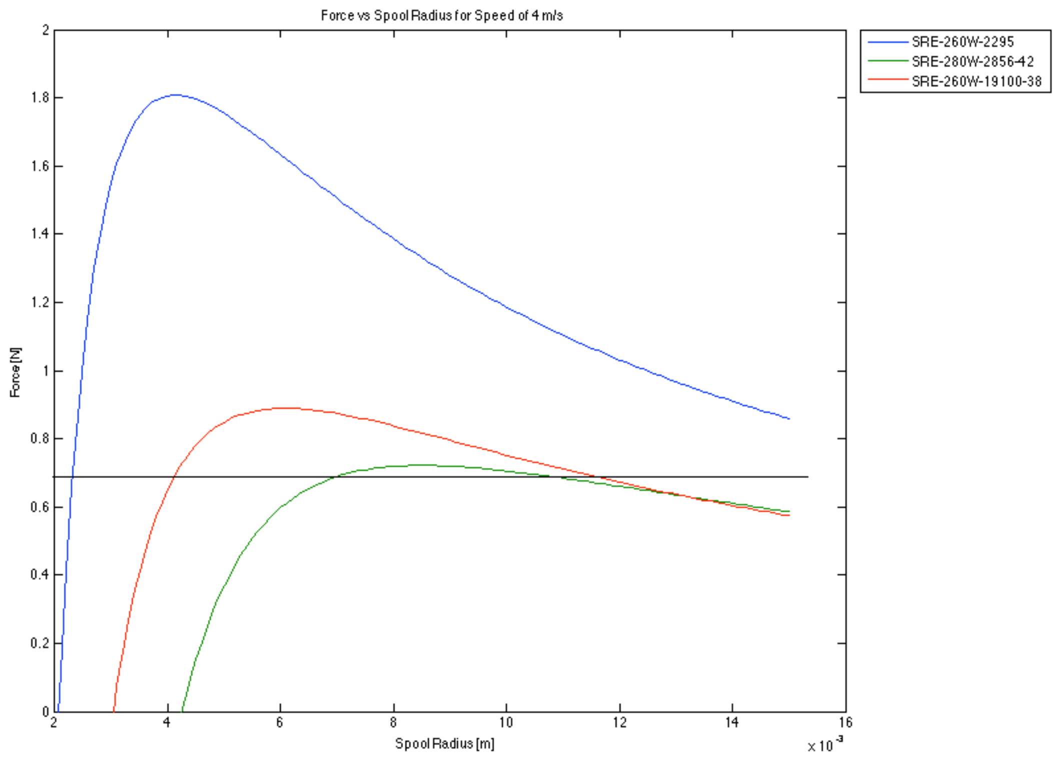


Figure 6.4: Graph showing force vs. spool radius relationship at a velocity of 4 m/s

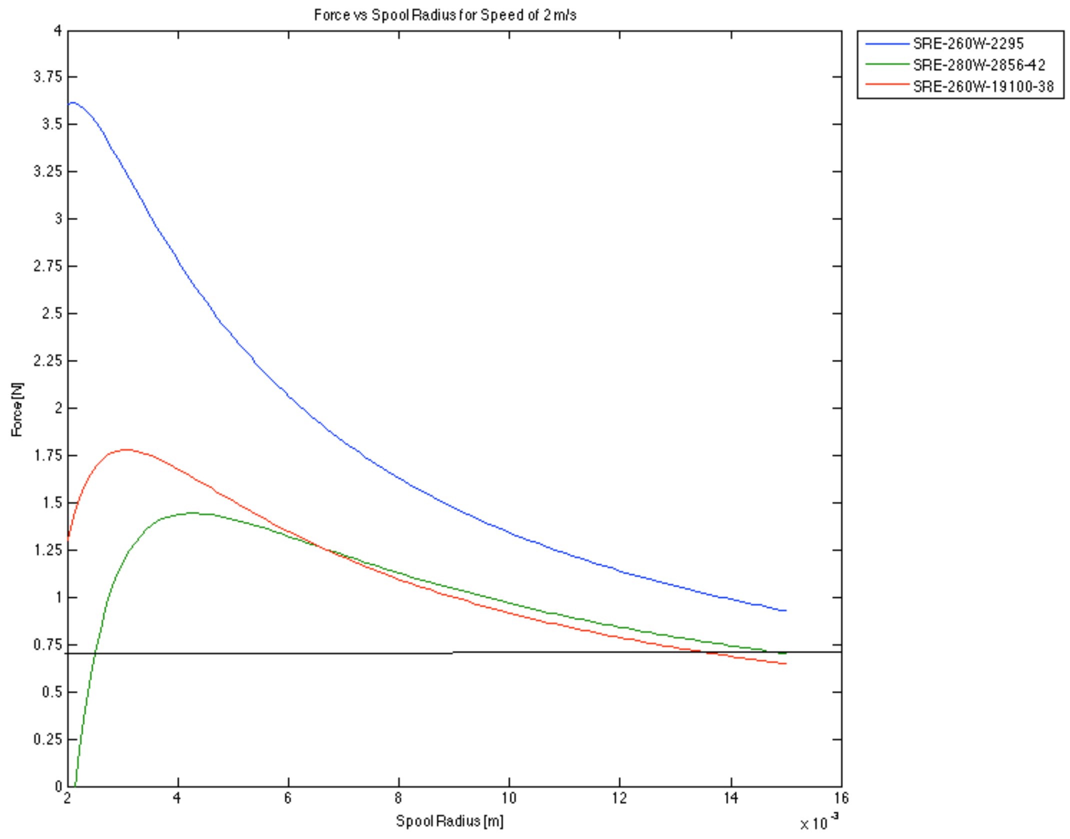


Figure 6.5: Graph showing force vs. spool radius relationship at a velocity of 2 m/s

Figures 6.8 to 6.11 show how the force produced by the motor changes with spool radius for different velocities. From these graphs it is clear the SRE-260W-2295 produces the most force over the range of velocities and therefore the most appropriate choice out of the three motors. To ensure that the most force possible is produced over the range of velocities, a spool radius of 5mm was chosen.

6.2 Design of Chassis and Spool

In this section the design of robot chassis and spool will be described. The requirements for the chassis were that it should be robust enough to survive multiple landing without major damage being caused to the internal electronic components and the same time provide a solid platform for the motor. The SolidWorks assembly for the Perspex base is shown below.

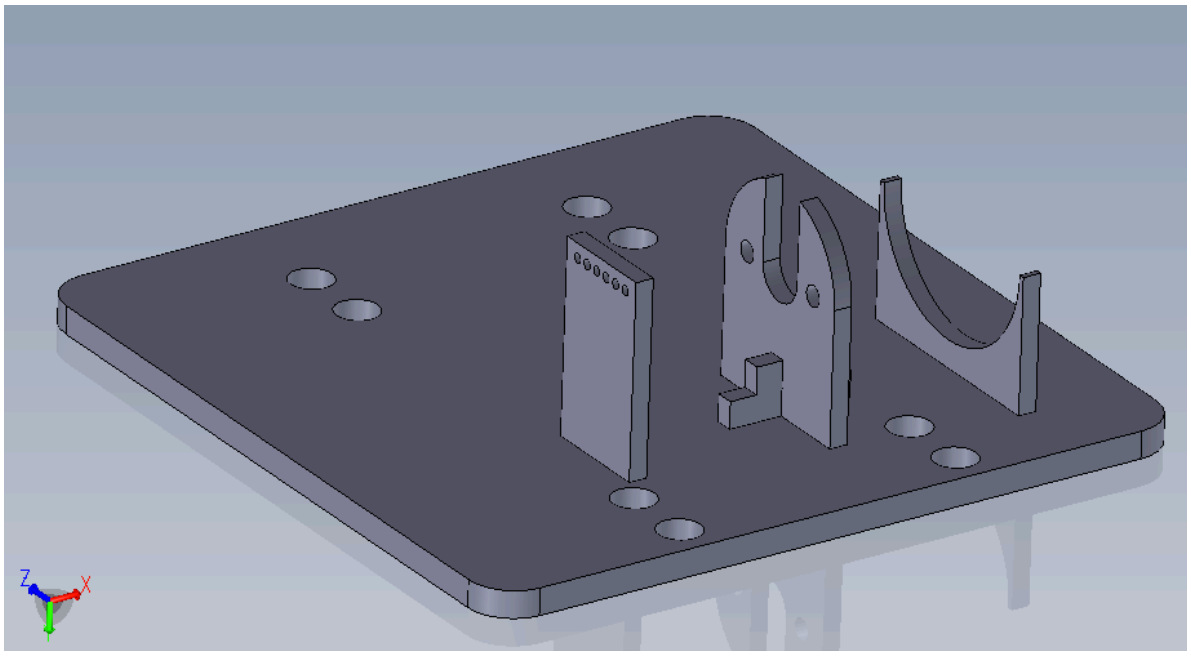


Figure 6.6: SolidWorks assembly of robot chassis

The base plate was laser cut out of 3mm Perspex while the motor support and encoder stand were designed from 2mm Perspex. The holes in the base plate were designed to fit the connectors of Lego Technic so that a roll cage could be constructed to protect the internal components.

The spool was also designed on SolidWorks and built to the specifications calculated in the previous section. The final product was 3D printed using a MakerBot 3D printer.

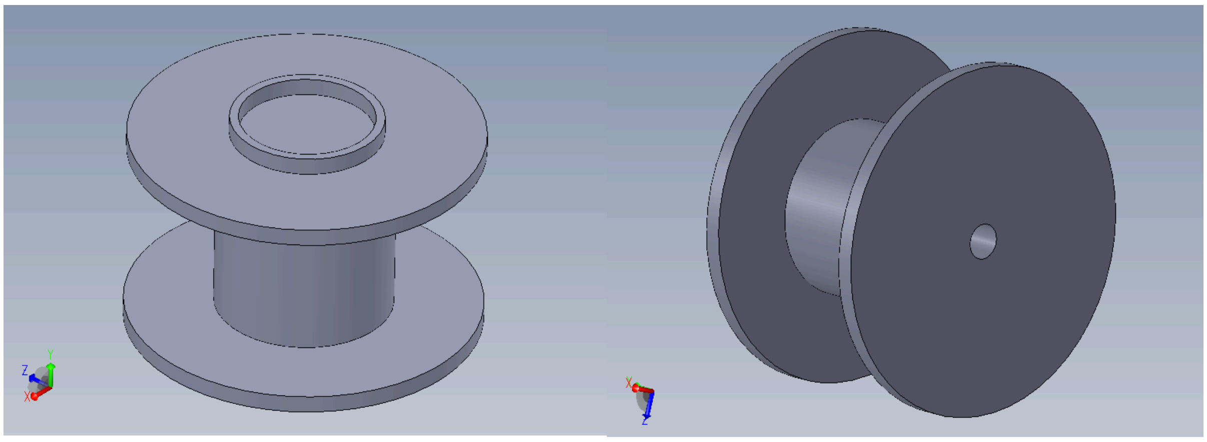


Figure 6.7: SolidWorks image of spool

Shown below is the completed spider robot with Lego roll cage and internal components installed.

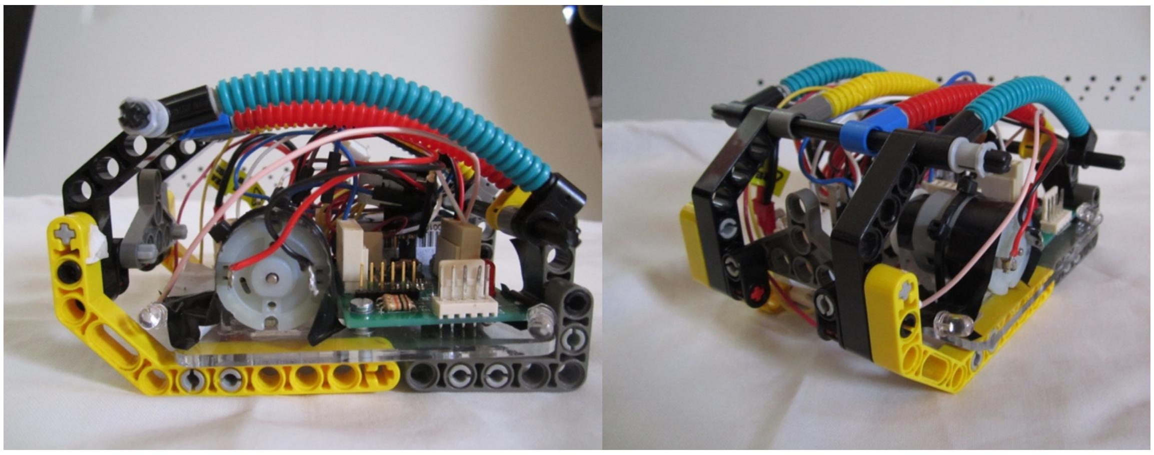


Figure 6.8: Image of the completed spider robo

6.3 Overview of components

6.3.1 Microcontroller

The microcontroller used was the iNEMO development board with STM32F103RE chip. This dev board was developed by Callen Fisher specifically for use in small mobile robotics. It is light weight and contains an accelerometer, gyroscope, and magnetometer.

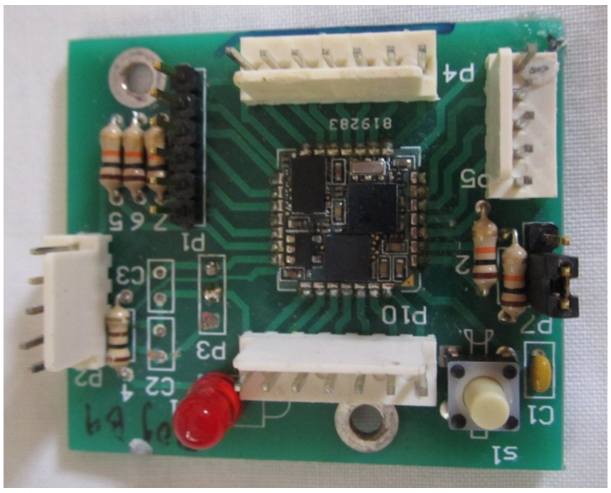


Figure 6.9: Image of iNEMO development board

Some relevant data for the STM32F1 is shown below [19][20].

Table 6.3: Table of showing some relevant features of the STM32F1

Voltage Supply	3.3v
MCU	32 bit, 72Mhz
Accelerometer	LSM303DLHC
Gyroscope	L3GD20
Supported Interfaces	CAN, USART,SPI and I2C

ADC	8 ADC channels
Timers	2 advanced, 4 general purpose, 2 basic

As shown in table 6.3, the STM32F1 has a system core clock of 72Mhz which is fast enough to implement control algorithms. It also has an on-board accelerometer and gyroscope which will be important for giving measurements of speed and body position of the robot.

This microcontroller was also chosen because it is has been used by other members of the project group in previous years which meant there was code available that was already written. Having code for initializing peripherals already written will allow for more time to be allocated to other parts of the project.
As the iNEMO does not have serial connector the microcontroller was debugged via a SWD cable connected to the STM32F4 discovery board.

6.3.2 *Driving Circuit*

In order to drive the motor, a driving circuit was built using the IRL520n MOSFET. The output of iNEMO is 3.3v. This meant that the MOSFET needed to be logic level. The IRL520n was ideal as it has a maximum gate threshold voltage of 2v [21].

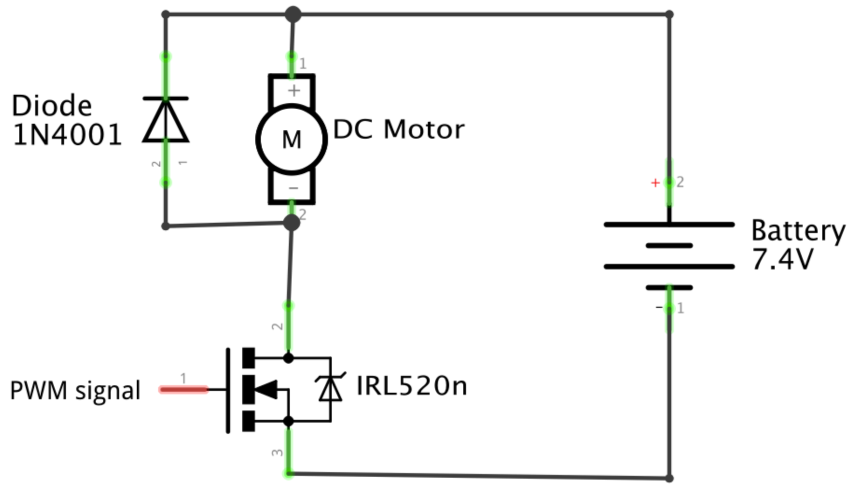


Figure 6.10: MOSFET Driving circuit

Due to the inductance of the motor, charge can build up on the negative terminal. Inductance resists a change in current so when the MOSFET is suddenly turned on, the current attempts to continue to flow. This current spike can have damaging effects on the MOSFET switch. The fly-back diode gives the current a path to dissipate through the motor. The diode chosen was the 1N4001 which is a standard diode with a breakdown voltage of 100V which is more than sufficient for this application.

6.3.3 Encoder

For an effective controller to be implemented it necessary for the speed of the motor to be measured. To do this a shaft encoder was needed. The encoder that was chosen was the Pololu Magnetic Encoder which is specifically designed for small DC motors. The encoder consists of a magnetic disc and a hall sensor. The resolution that was used was 6 counts per revolution[22]. This is accurate enough for the purposes of this project as it amounts to 60 degrees and corresponds to 5mm in terms of dragline length.

The encoder is made to fit on the Pololu motor's rear shaft however the motor chosen for this project was not made by Pololu and does not have a rear shaft which to attach the encoder. The solution to this was to attach the magnetic disc to the end of the spool and build a mount on which to attach the sensor.

6.3.4 XBee

The XBees are wireless RF modules. They are used to send data from the spider to the base station. The XBee model used was the XBee Pro series 1. The advantage of using these modules is that they are fast and robust and they also have a range of 90m indoors[23]. The XBee connected to the STM32F1 via a USART connection and is powered off the same voltage as the microcontroller which makes interfacing between them simple.

A program called X-CTU is used to pair the transmitter and receiver XBees. This is to make sure they are transmitting at the same baudrate and have the same ID number.

6.3.5 Launch detection circuit

For effective implementation of the control algorithm the point at which the robot left the launcher needed to be known. To sense when this took place a simple distance sensor circuit was designed using the QRD1114 package. The QRD1114 consists of a Infrared LED and a phototransistor [24]. The amount of reflected infrared light from the LED determines the resistance of the phototransistor. The closer the device is to an object the more light gets reflected back which reduces the resistance of the phototransistor. By connecting the phototransistor in series with a resistor a potential divider circuit is created. In this way the distance from a surface can be determined by measuring the output voltage from the divider circuit, as for close objects the phototransistor resistor will be low outputting a voltage high and as the distance decreases the resistance increases and the voltage drops.

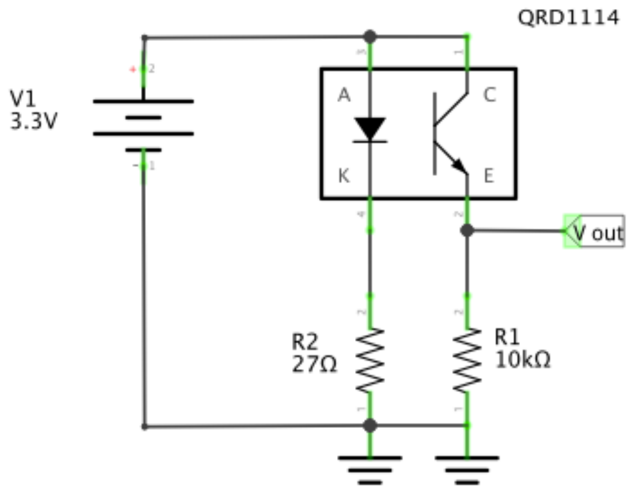


Figure 6.11: Circuit for detecting the launch

The infrared LED requires a current of 50mA and with a supply voltage of 3.3v and a forward drop of 1.7v meant a 27ohm resistors was needed to be placed in series [24].

The circuit was install underneath the robot and the output voltage signal connected to an input pin on the microcontroller. Using input capture, a high meant the robot was on the launcher while a low meant it had left the launcher.

6.3.6 *Battery*

The battery used in this project was a Zippy Li-Po 2 cell. It was chosen for its light weight and with a voltage 7.4v it was ideal for powering the both the iNEMO and the motor.

Figure 6.12: Image of Zippy 350 2S battery [25]

6.3.7 *Block Diagram of Hardware layout*

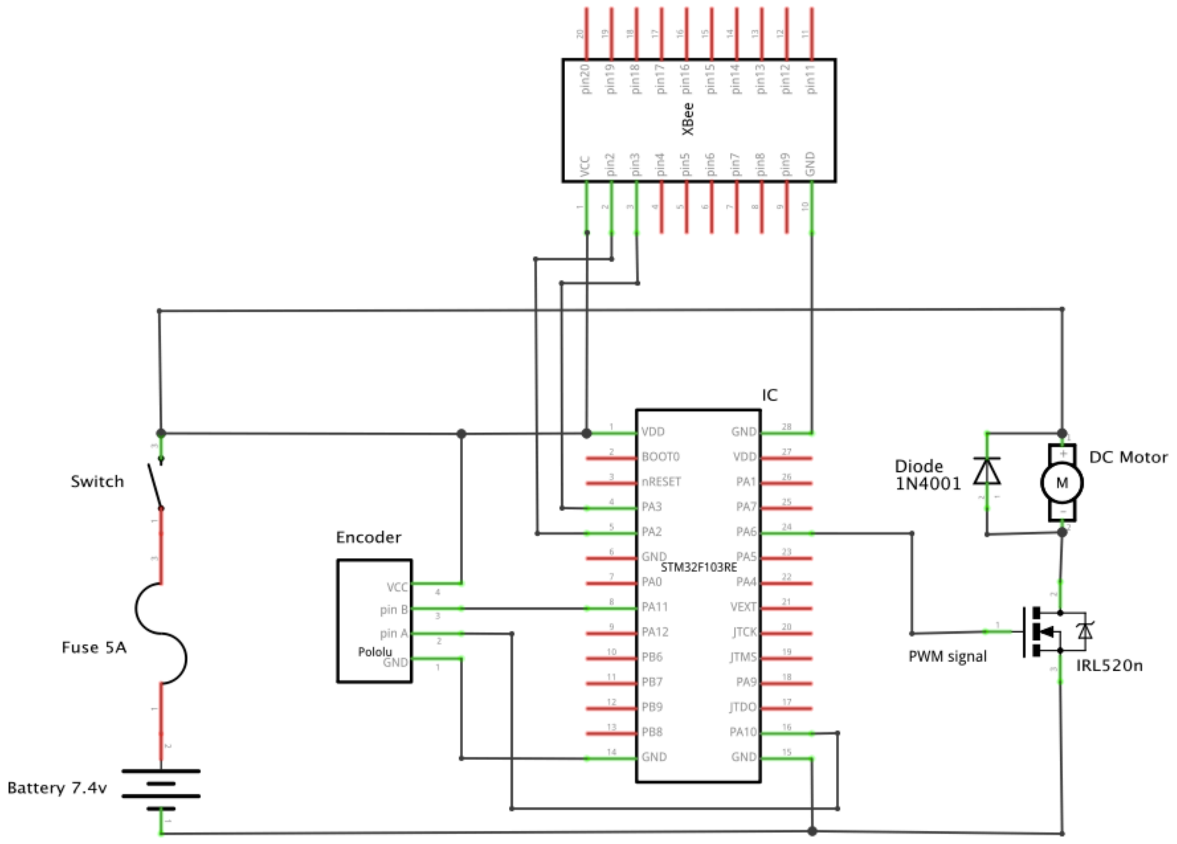


Figure 6.13: Block Diagram of Hardware System

6.4 Design of Launcher

To test the robotic platform a launcher was required to assess the design under different initial conditions such as launcher angle and velocity. A launcher had been built for a similar purpose in previous years testing but improvements and modifications had to be made before it could be fully operational.

The first task was to design a tension spring that would be able to launch the robot at the required velocities. The choice to use a spring over other materials such as elastic, was based on the superior durability as well as reduced susceptibility to temperature changes and deformation.

The spring was custom designed in order ensure the correct spring constant was selected to produce the kinetic energy needed. An online spring calculator was used to determine the parameters such as wire and spring diameter, and spring length which are required for manufacture. The maximum test velocity was decided as 8m/s. This would give adequate range of velocities over which to test. The calculations for the spring are as follows.

Kinetic energy at 8 m/s: $E_k = \frac{1}{2}mv^2 = 0.5 \times 0.2 \times 8^2 = 6.4$ J
Spring Energy: $E_{spr} = \frac{1}{2}kx^2$
Where x was chosen to be 0.6m to make calibration of position to speed easier
Therefore required spring constant: $k \approx 35\ N/m$

The manufacturing dimensions which were found to fit these specifications are shown below.

Table 6.4: Manufacturing specifications for tension spring

Material	Carbon Black
Hooks	Machine
Wire Diameter	1mm
Body Length	245mm
Outer Diameter	11.7mm

On the launch platform itself the guide rails had to be replaced and modifications had to be made incorporate the new spring design.

7. Software

7.1 Data Logging Application

The application used to log the test data was developed by Callen Fisher, it was programmed in C# and was run using Microsoft Visual C# 2010. The application interfaces with the Xbee transceiver through the serial communications port. The data that was logged included readings from the accelerometer, gyroscope as well as motor speed. The purpose of logging this data was for use in the calibration of the accelerometer and gyroscope. The data was also logged during the testing phase of the project to allow for further analysis on the behaviour of the system.

7.2 System Software

As mentioned in the hardware chapter the microcontroller used for this project was the iNEMO development board. The language used to programme the on-board STM32F1 is C and the IDE used was Atollic TrueStudio. Most of the code used to initialize the peripherals and communication with the XBees was provided for by Callen Fisher. This was in order to allow for more time focus on the main goals of this project. However, the code still needed to be integrated for use in tasks specific to the project and this process will be detailed in the following sections.

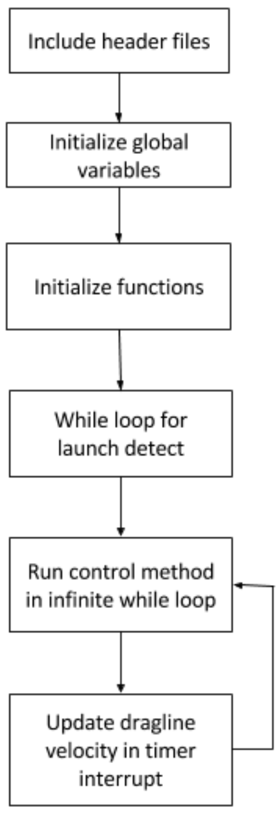

Include header files
Initialize global variables
Initialize functions
While loop for launch detect
Run control method in infinite while loop
Update dragline velocity in timer interrupt

Figure 7.1: Flow Chart of system software implementation

7.2.1 PWM

The motor was run at a frequency of 1.2kHz which is a standard frequency for dc motor. In order to generate PWM at this frequency the correct prescalar and auto-reload value needed to be calculated.

The formula used to calculate timer frequency is given below,

$$Timer\ frequency = \frac{System\ core\ clock\ frequency}{(Prescalar + 1)(Auto\ reload + 1)}$$

The system core clock for the STM32F1 is 72 MHz and choosing a prescalar value of 0 gives a auto reload value of 60000 in order to to generate the required frequency of 1.2kHz. The duty cycle is updated by adjusting the value loaded into the timer capture register where 60000 gives a duty cycle of 0% and 0 results in a duty cycle of 100%.

The formula used to update the duty cycle based on the required voltage is shown below,

$$Duty\ cycle = 60000 - \frac{volts}{7.4} \times 60000$$

where, 7.4 is the battery voltage and volts is the voltage that is calculated from the control action. To make sure the voltage did not go above the rated voltage of the motor the value of volts was hardcode limited to 6v.

7.2.2 Encoder and Input capture

The speed of the motor was measured using an magnetic encoder. The output of the encoder is a square wave which needed to be inputted to the microcontroller in order for the motor frequency to be calculated. It was decided to use input capture to record the time for each period. To do this GPIOA11 was first setup as an input. Then timer 1 was initialized for input capture.

One factor that had to be considered when setting up the timer was the possibility of an interrupt event overflow which could lead to false readings. At low frequency it possible that time between two rising edges could be longer than the timer period. If this occurs the timer cannot account for the extra period and frequency will be calculated at a much higher value than the true value. The obvious way to overcome this is to make the prescalar value high which will increase the timer period. This unfortunately results in the timer resolution decreasing, which could also lead to inaccuracies in the frequency calculations. So a balance had to be found between preventing timer overflow and minimising resolution inaccuracy. This was done by determining the lowest motor speed from simulations and then basing the prescalar calculations on this value.

The encoder has a resolution of 6 counts per revolution which means that there will be a rising edge every 60 degrees of rotation by the motor shaft. If the frequency is calculated as 1/period, where the period is the time between rising edges, the calculated value will be three times that of the actual frequency of the motor. So the corrected formula for calculating motor frequency becomes,

$$frequency = \frac{1}{3 \times time\ between\ rising\ edges}$$

From simulations of the system it was determined that the lowest motor speed would be 2 m/s which gives an angular speed of 400 rad/s and a frequency of 64Hz. This means that the lowest encoder frequency will be approximately 180 Hz.

From this the timer could be calculated using the formula given in section 7.2.1. The auto reload value was chosen to be 655536 to ensure the longest possible period.

$$prescalar = \frac{72\times 10^6}{(180)(2^{16}+1)} - 1 \approx 5$$

8. Calibration

8.1 Accelerometer and Gyro Calibration

In order to obtain accurate readings the gyroscope and accelerometer had to calibrated. The gyroscope is particularly sensitive to changes in temperature which needs to be accounted for. Firstly the accelerometer was calibration was done and this process is described next.

To calibrate the accelerometer a specially design calibration gimbal was used. The sensor was placed on the gimbal and data was logged for one minute for each position, X up and down, Y up and down, Z up and down. The logged data was then imported to MATLAB. Calibration code was then used to analyse the data. The code outputs a 4x3 matrix which is then multiplied by a matrix containing the measured accelerometer reading appended by a one to give the calibrated readings. This calculation is shown below,

$$\begin{bmatrix} X_{cal} \\ Y_{cal} \\ Z_{cal} \end{bmatrix} = [X_{raw} \quad Y_{raw} \quad Z_{raw} \quad 1] * A$$

where subscript 'cal' refers to the calibrated values, subscript 'raw' refers to the measured values and A is the calibration matrix. The A matrix for the particular INEMO used was calculated as,

$$A = \begin{bmatrix} 0.1540 & 0.0029 & -0.0011 \\ -0.0051 & 0.1596 & -0.0034 \\ -0.0016 & -0.0018 & 0.1550 \\ 0.0113 & 0.0066 & 0.0343 \end{bmatrix}$$

Next, the temperature calibration for the gyroscopes was performed. The calibration code outputs three graphs for the pitch, roll and yaw directions which relates the gyroscope reading in deg/s to temperature. The equation of best fit is shown on each graph. This equation is what is used to negate the effects of temperature on the gyroscope readings. The calibration graphs are shown below.

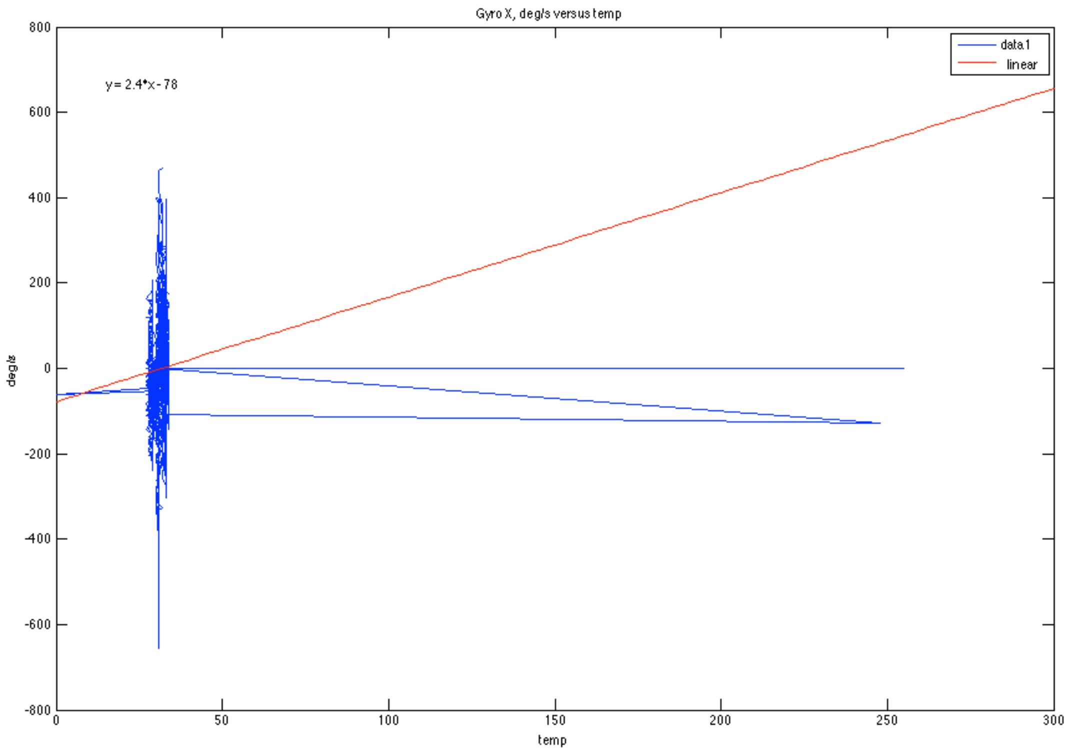


Figure 8.1: Calibration graph for gyroscope X vs. temperature

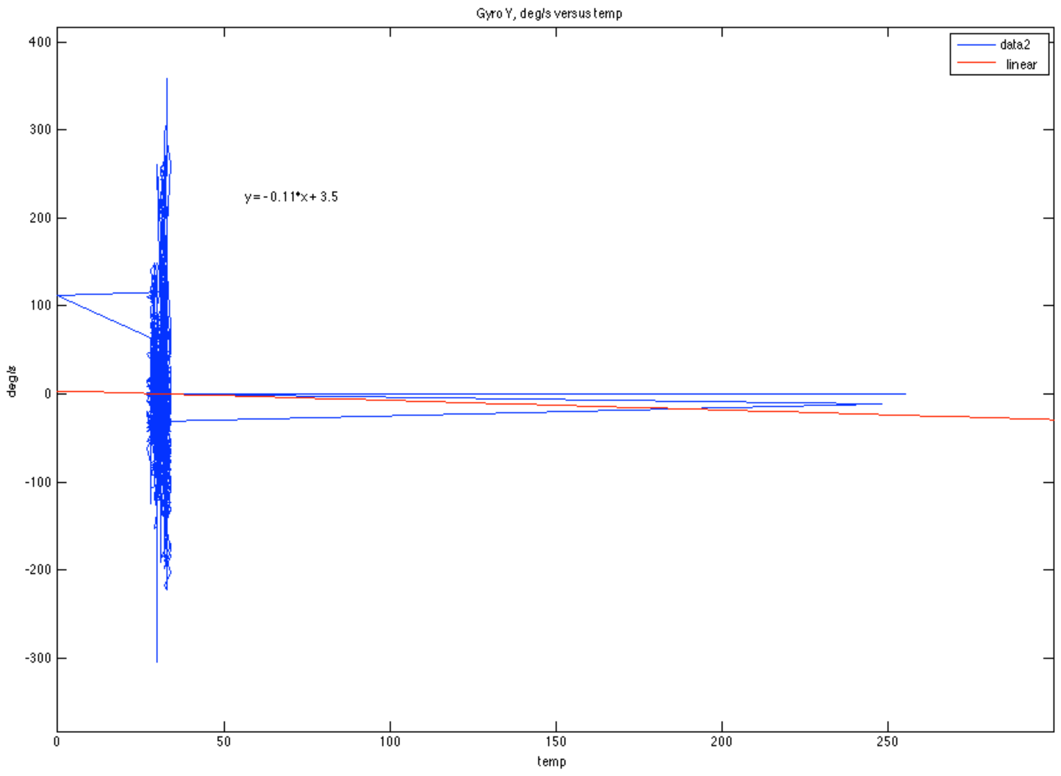


Figure 8.2: Calibration graph for gyroscope Y vs. temperature

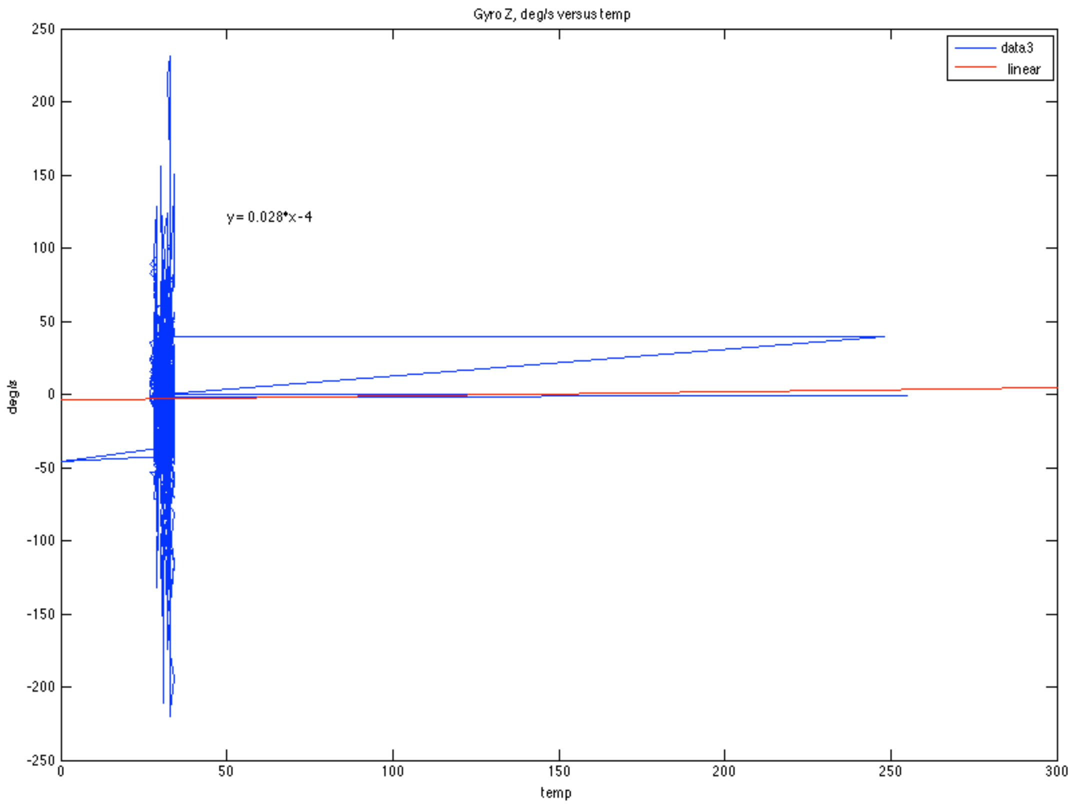


Figure 8.3: Calibration graph for gyroscope Z vs. temperature

The gyroscopes also have a scaling bias which must be accounted for. To determine the bias, the sensor was turned 360 degrees in the pitch, yaw and roll direction while logging data. The gyroscope reads angular velocity and so by integrating the data it was possible to check whether the sensor reading actually equalled the required 360 degrees. In the cases where this was not the case a scaling factor was calculated to ensure that the gyroscope gave a correct reading.

8.2 Motor Torque Calibration and Testing

Motor calibration was done to determine the relationship between terminal voltage and the motor stall force. This relationship was required in order to control the motor in to producing the necessary dragline force. The process to collect the necessary data was as follows.

First the motor was placed in a vice to keep the it secured during testing. Then a thread was wrapped around the spool and to the end was attached a mass. The terminal voltage of the motor was then adjusted until the motor torque perfectly balanced the force due to weights on the end of thread. At this particular voltage the force applied from the weights is equal to the stall force of the motor.

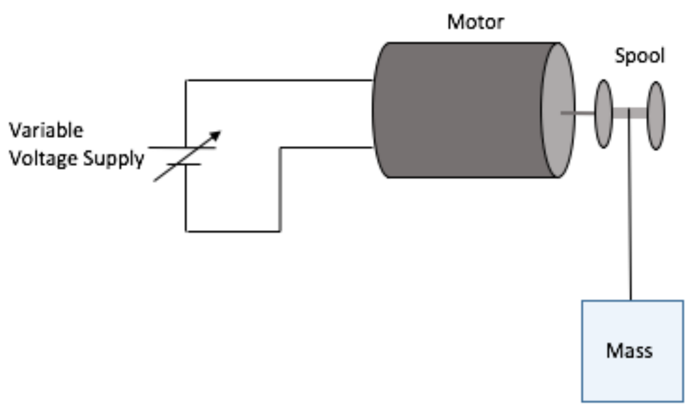


Figure 8.4: Diagram of testing setup

From repeating this process and incrementing the mass by 10 to 15 grams each time as well as taking multiple readings for each mass. The voltage readings were recorded and the average value for each test weight were plotted on a graph. From this graph the relationship between terminal voltage and the corresponding stall force was established.

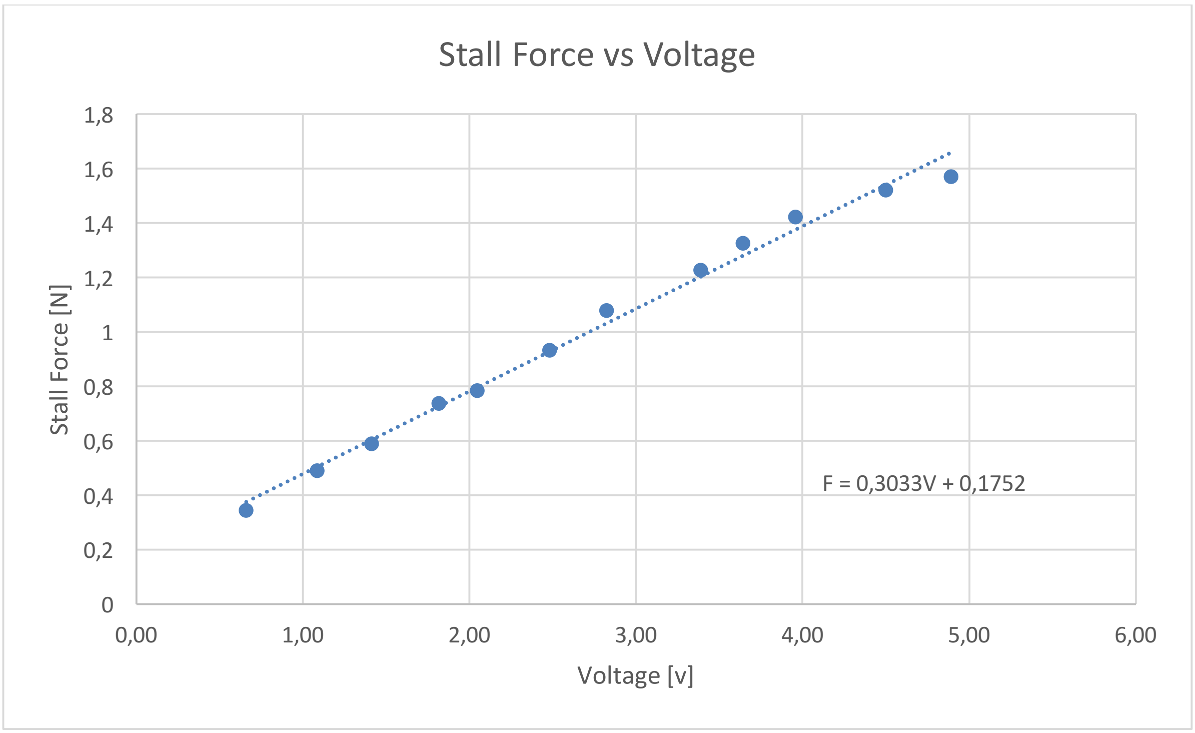


Figure 8.5: Graph of Stall Force vs Terminal Voltage

As can be seen from the graph above, the equation relating stall force and voltage is given by $f_{stall} = 0{,}3033V + 0.1752$

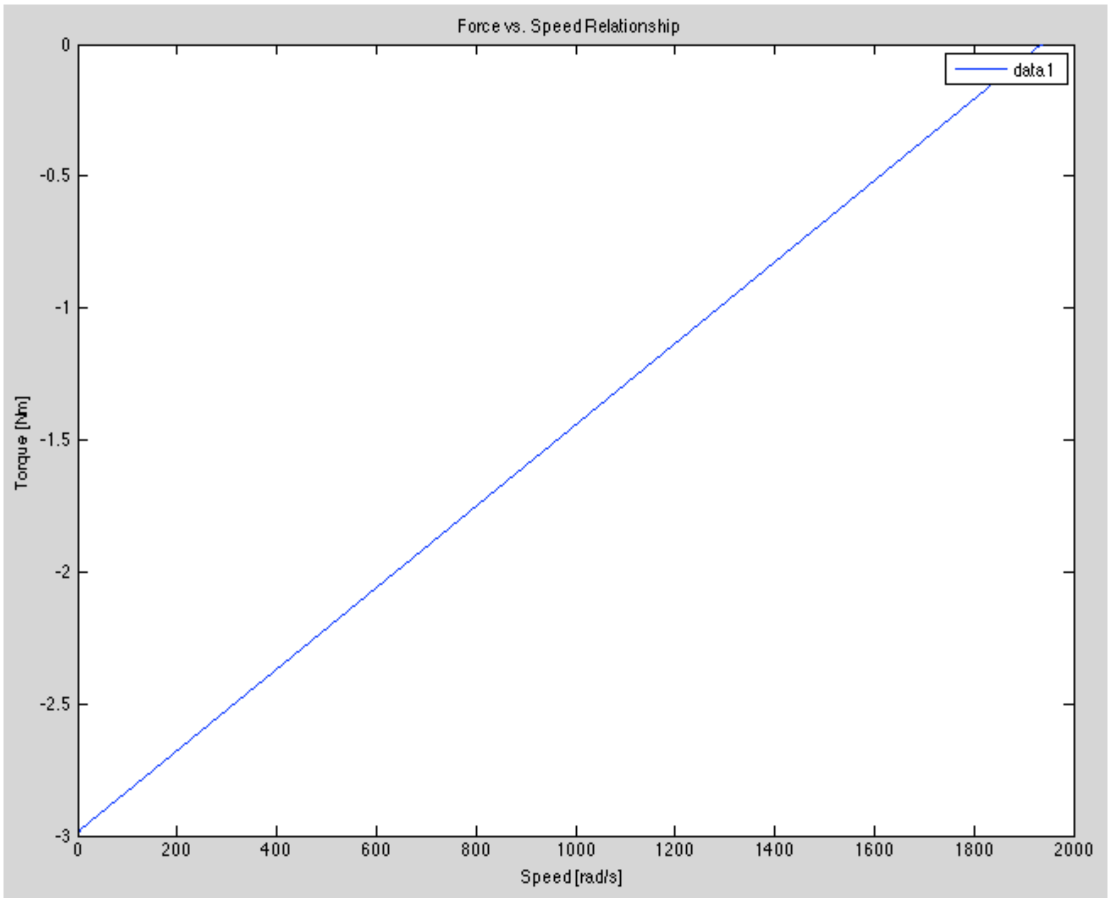


Figure 8.6 Force vs speed graph at the nominal voltage of 6v

From force vs. speed graph the equation relating force and speed can be determined.

$$f = m\omega + f_{stall}$$

where, m is the gradient of the line and f_{stall} is stall force at a particular voltage. In order to control the motor the force command generated from the control action must be mapped to a voltage. This can be achieved by substituting f_{stall} into the above equation giving,

$$f = m\omega + 0.3033V + 0.1752$$

For the motor in question the gradient is 0.0015 and assumed to be constant for all voltages. This assumption is valid due to the fact that the force-speed characteristics of the motor are based on the intrinsic properties of the motor and should follow a linear relationship for all voltages up to and including the rated voltage.

By rearranging the equation the voltage required can be calculated.

$$V = (f - m\omega - 0.1752)/0.3033$$

9. Controller Design

In this section will deal with the process in designing the controller for achieving the precise landing of the robot.

9.1 System Analysis

Before the controller can be designed the open loop system needs to be analysed to gain an understanding of what is possible in terms of controller design.

Firstly, the x position of the center of mass the system was observed for varying launch angles.

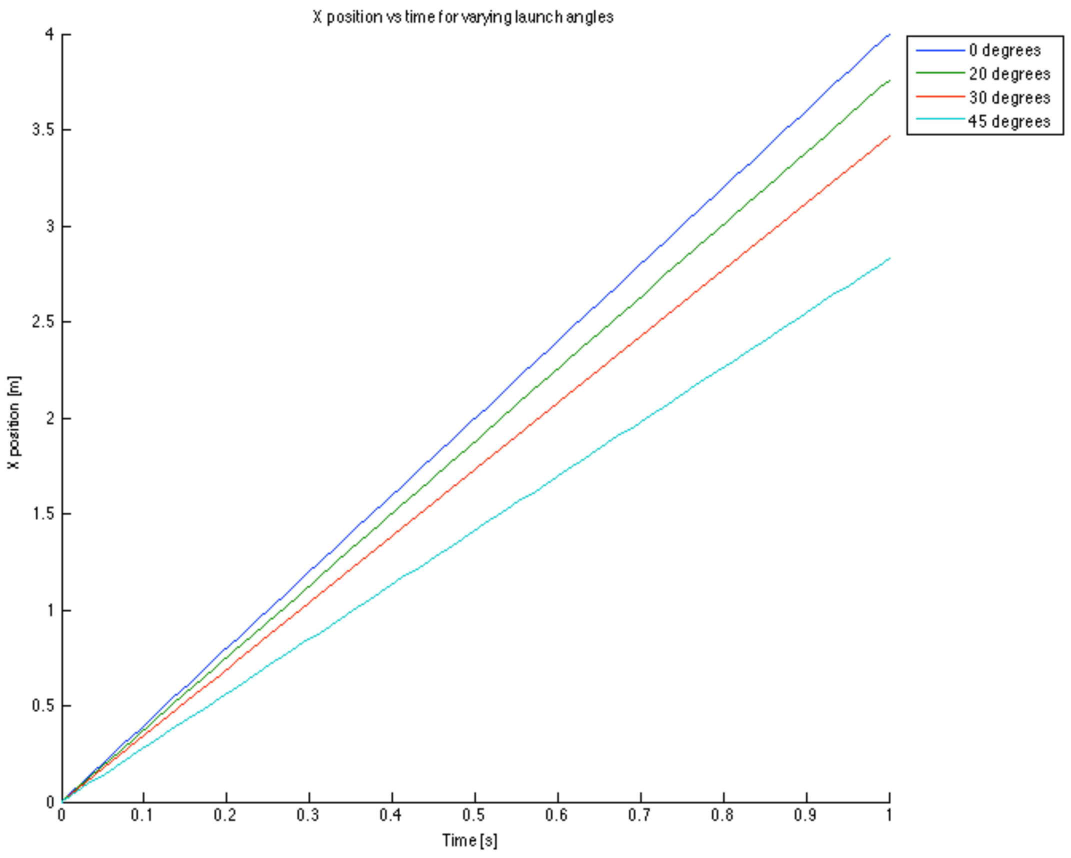


Figure 9.1: X position as a function of time for different launch angles

The result was that by increasing the launch angle less distance in the x direction was achieved. As the launch angle increases the component of velocity in the x direction decreases while the y increases and because x position is a function of velocity this means less distance is achieved.

Next, the effects of changing the initial velocity was investigated.

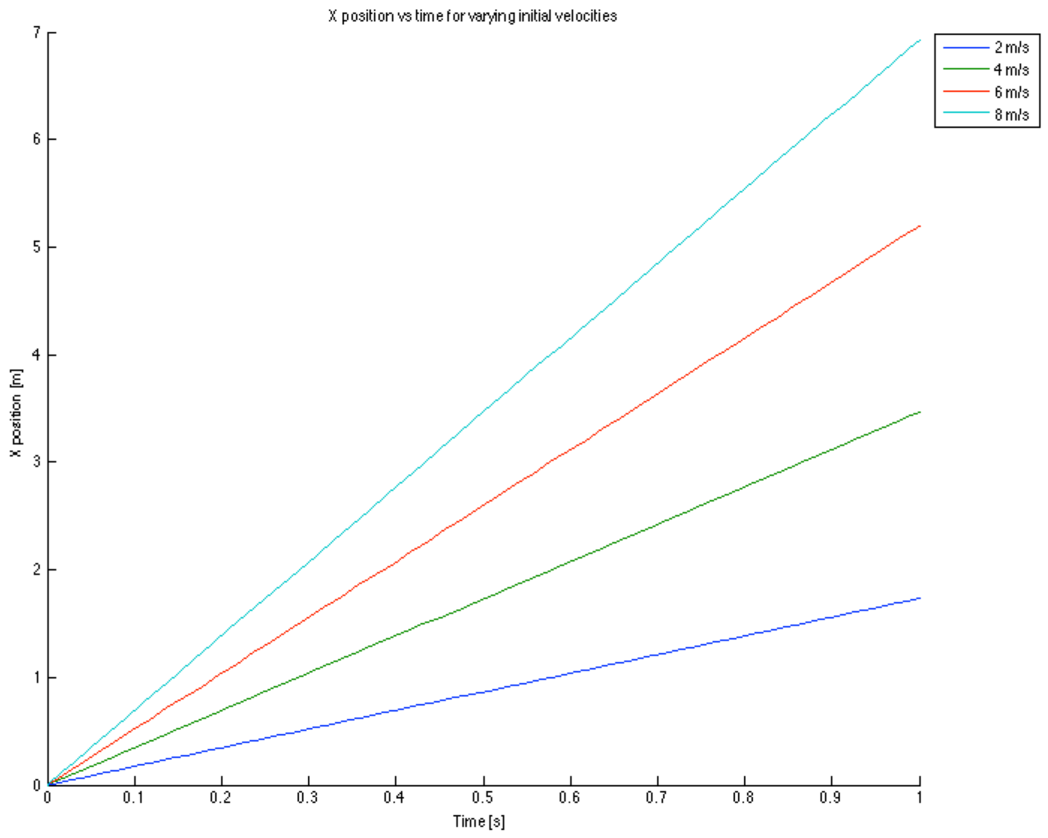


Figure 9.2: X position as a function of time for varying initial velocities and a launch angle of 30 degrees

In the figure above the results of increasing the initial velocity for a constant launch angle of 30 degrees are shown. Increasing the initial velocity lead to an increase in the x distance achieved per unit time. This is to be expected as x position is proportional to velocity so increases the velocity should result in a greater x distance.

The effect of a constant dragline force on the x position was also analysed. For increasing dragline force the less distance in the x direction. It is important to note the stabilizing effect the dragline force has on the system. For a force of -0.8 N the x position graph even begins to take on the shape of a first order response. This is encouraging in terms of controller implementation as it shows that by controlling the dragline force the x position can also be controlled.

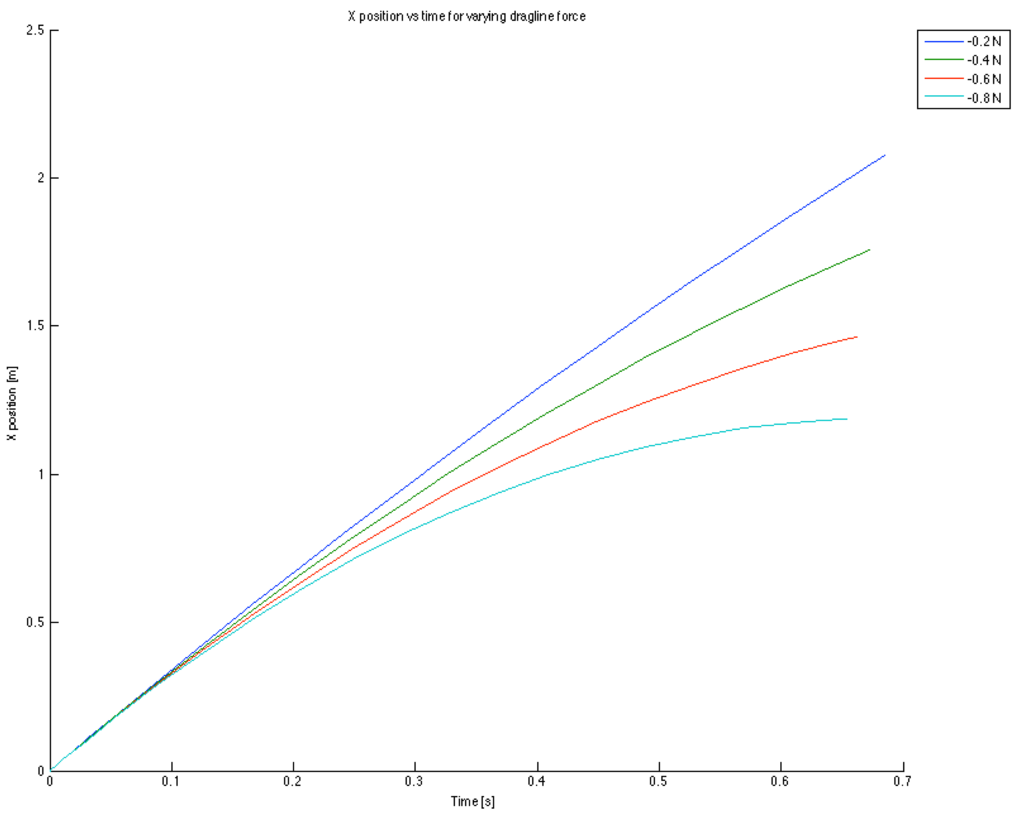


Figure 9.3: Graph of the effect of different magnitudes of dragline force on the X position

The body position of the robot is also an important aspect that needed to be investigated. In the figure below the abdomen position is shown for various magnitudes of dragline force.

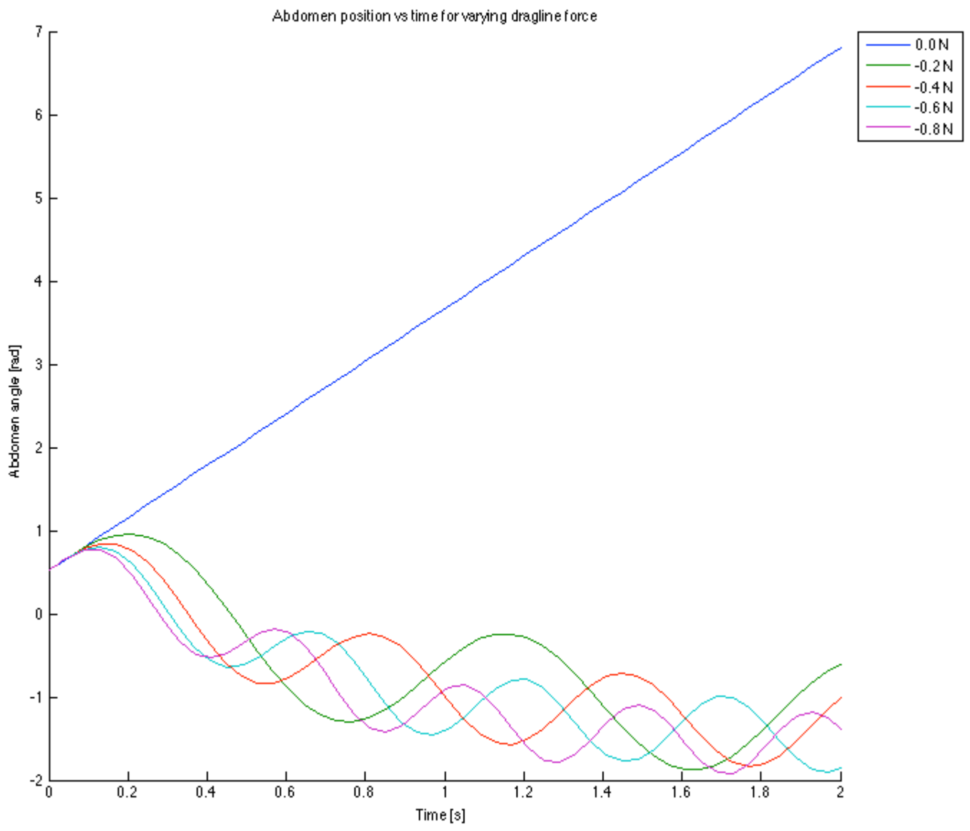


Figure 9.4: Abdomen angle with respect to time for varying magnitudes of dragline force

Given that the initial rotation of the body was 3.14 rad/s it is clear from the figure that the without the influence of a dragline force this rotation will continue unimpeded. However, the introduction of a dragline force causes the a pitch reversal of the body which eventually leads to the stabilization of the rotation. Even low values of force result in the body angle following a sinusoidal path rather than spinning uncontrollably. In terms of control this means that even if the body angle is not controlled directly any dragline force that is used to control the x position of the body will have the added effect of stabilizing any body rotation.

9.2 System Constraints

As in most physical systems there are constraints that will effect what is possible in terms of controller design. These limiting factors determine the ultimate performance of the controller and set the bounds as to what can be achieved. Identifying these limits is important as a controller that performs well in a simulated environment otherwise may not have the expected results when implemented on a physical system.

The main limit in spider dragline system is the amount of force that can be produced by the motor. As seen in the hardware chapter, the amount of force that can be produced is dependent on the torque-speed characteristics of the motor. So, at higher speeds, the amount of force available is decreased.

Linked to this constraint is the limit on the amount of voltage that can be supplied to the motor. It is by adjusting this supply voltage that the controller will be implemented as a higher voltage relates to a higher force output. Therefore, when designing the controller this needs to be taken into consideration as supplying the motor with voltages greater than the rated voltage could cause it to burn out.

Another constraint is the sampling rate available by the processor on the microcontroller. While the STM32F1 does have a CPU speed of 72Mhz which should be more than sufficient for implementing the control method, it should still be noted that this speed is not infinite and that the possibility of errors relating to sampling could occur.

It is also not physically possible for the motor to add energy to the system. This means that the maximum x distance that can achieved is determined by the initial conditions. This is due to the fact that although the dragline is modelled as a rigid body, this is only an accurate representation when it is in tension. If the motor attempts to add energy by increasing angular velocity in the same direction that the body is travelling, it will cause the dragline to go slack preventing the reaction force needed to increase the velocity of the system. As a result, the possible setpoints that can be achieved will always be less than or equal to the distance that is achievable based on the initial conditions. The fact that the motor can only be used as a braking force has consequences for setpoints but also for performance characteristics such as settling time.

9.3 Linearization

The dynamics of the system determined using the Lagrangian method were found to be nonlinear. This meant most classical approaches of control are not applicable. For nonlinear systems, a common technique is to apply some form of linearization in order to simplify the implementation of a controller. Once the system has been linearized it makes the control problem significantly easier as it is then possible to apply the classical techniques of linear systems. There are different approaches that can be taken to linearize a system and the most effective method is often dependent on the dynamics of the system and the type of control that is required. In the following section two approaches for the linearizing the system are investigated and the their success at approximating the system discussed.

9.3.1 Taylor Linearization

The first method of linearization that was used was Taylor series linearization. The main disadvantage with Taylor series approximation is that the system dynamics are only linearized around a single point. While this can often be used to approximate other points close by, if dynamics of the system significantly change then the approximation often becomes inaccurate. Therefore this technique is best suited to systems which have a fairly restricted area of operation. In the spider dragline system both the initial conditions and the the amount dragline force applied can drastically change the dynamics. Attempts to linearize the system about one or even numerous points proved unsuccessful as once the any of the aforementioned variables were changed the approximation became extremely inaccurate.

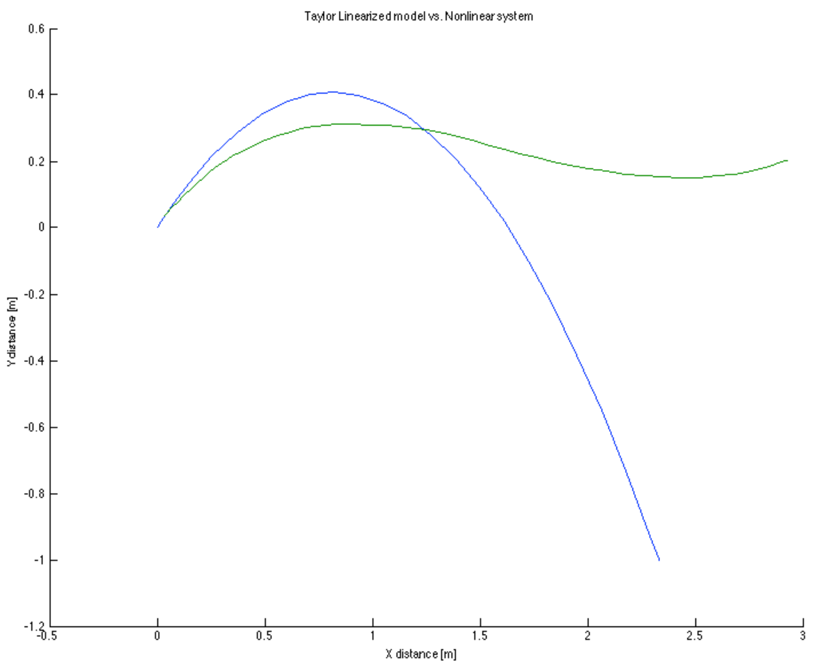
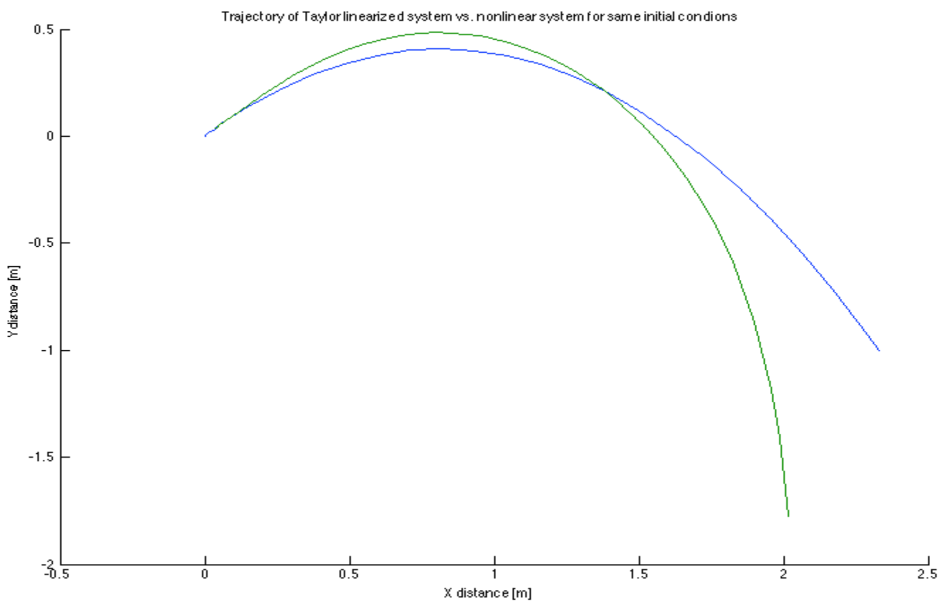
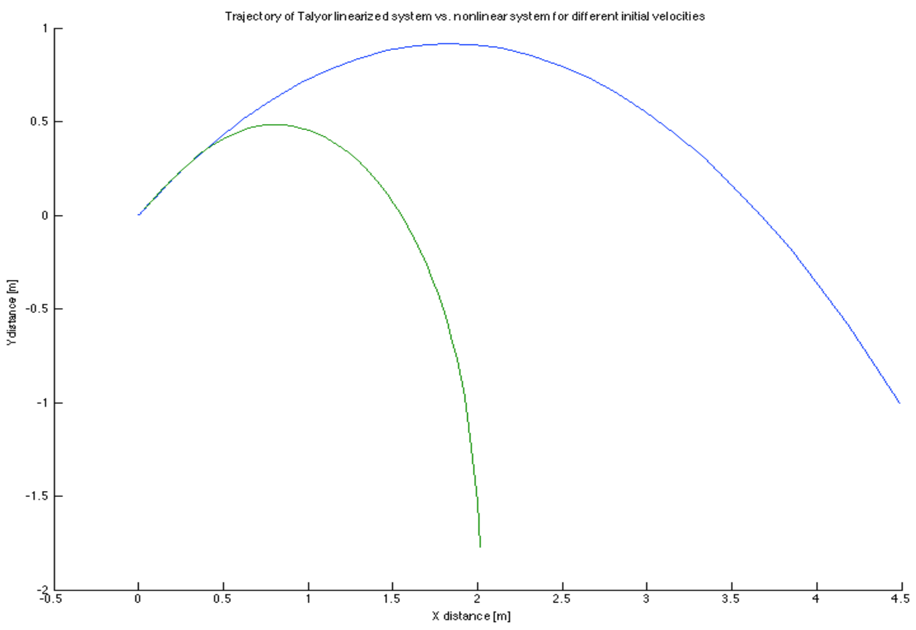


Figure 9.5: Comparison of Taylor linearized model (green) and the original nonlinear model (blue)

In the figure 9.5 above, the system is linearized for a varying points depending on the length of the dragline. The purpose of changing the point around which the system is linearized is to give a better approximation as the closer the system is to the point around which it is linearized the more accurate the linearization will be. However, it is clear from figure 9.5 that even using this model scheduling approach the approximation is quite poor, with the trajectory of the linearized system deviating significantly from the actual system.



9.6 (a)



9.6 (b)

Figure 9.6: (a) Trajectory of linearized system around single point (green) and nonlinear system (blue) for the same initial conditions. (b) Trajectory of linearized system (green) and nonlinear system (blue) for different initial velocities.

Figure 9.6a shows how the linearized system compares to the nonlinear system for the same initial conditions. The system is linearized around the peak point of the trajectory and while the approximation is not perfect, it follows a similar shape to the nonlinear system. Figure 9.6b shows the effects of changing the initial conditions of the nonlinear system. The only difference between (a) and (b) is that in (b) the nonlinear system was given an initial velocity of 6 m/s as opposed to 4 m/s in the linearized case. As a result the approximation in (b) is considerably worse off as the linearized model has no way of accounting for the change in the initial conditions of the actual system. This makes this method of linearization unsatisfactory for this system as it is inevitable that there will be changes in the system dynamics and the standard approach to Taylor linearization cannot account for these variations.

One method that could be used to overcome this problem is a model predictive approach, where next states could be predicted using the nonlinear model, and then using these predictions the linear model could be continually updated for future values. This also has complications brought about by the linear model constantly being adjusted. In order to deal with this issue, a gain scheduling approach would have to employed where different PID controllers would have to be used depending on the state of the system. This was felt to be an overly complicated solution to the problem, so an alternative approach was then investigated.

9.3.2 Partial Feedback Linearization

The next method of linearization that was used was partial feedback linearization (PFL). This technique is a more robust than Taylor linearization. The advantage being it does not rely on linearizing around a single point but rather uses feedback to cancel out the undesirable nonlinear dynamics of the system.

The general form of PFL is task space PFL, using this approach it is possible to linearize a function of the actuated and unactuated states. The x distance travelled by the spider, which is the state that needs to be controlled, can be written as a function of the generalized states as shown below.

$$x = Lcos(\theta_0) + \frac{1}{2}lcos(\theta_0 - \theta_s)$$

By applying the task space algorithm discussed in the Theory Development chapter, the J matrices become,

$$J_1 = \frac{\partial x}{\partial q_1} J_2 = \frac{\partial x}{\partial q_2}$$

where, q1 and q2 are the unactuated and actuated states respectively.

The control action v will then be given by,

$$v = \ddot{x}^d + K_d(\dot{x}^d - \dot{x}) + K_p(x^d - x)$$

As the function of x has now been reduced to a simple double integrator, state space pole placement can be used to determine the desired damping and settling time. As shown in the theory development chapter, the accelerations of the actuated states becomes,

$$\ddot{q}_2 = \bar{J}^{+}[v - \dot{J}\dot{q} + J_1 H_{11}^{-1} \emptyset_1]$$

while the unactuated state accelerations are given by,

$$\ddot{q_1} = -H_{11}^{-1}(H_{12}\ddot{q_2} + \emptyset_1)$$

However, in order to control the system a torque command is needed as it is the dragline force that will be used as the input to the system. This means a mapping from the control action to a force value is necessary. This was done by substituting above two equations for the accelerations into the following equation for the silk force [17][9],

$$F_s = H_{21}\ddot{q}_1 + H_{22}\ddot{q}_2 + \emptyset_2$$

This gives the force command as,

$$F_s = \frac{mv(L - lcos(\theta_s))}{Lcos(\theta_0) - lcos(\theta_0 - \theta_s)}$$

where, m is the mass of the robot, v is the control action, L is the length of the dragline, l is the length of the body, θ_0 is the angle of the silk and θ_s the angle of the body relative to the silk.

The requirement for the settling time was that the setpoint should be reached before the spider landed but at the same time the constraints on the input need to be observed. The landing time will vary due factors such as initial velocity and launch angle as well as y distance from the ground. Using the nonlinear system with a silk force of zero, simulations were done for initial conditions similar to that which would be used during the testing phase. From these simulations it was determined that a reasonable settling time would be 0,5s.

With a damping factor of 0.9 the proportional and derivative gains were calculated as,

$$K_p = 78$$
$$K_d = 16$$

The control action v is then

$$v = 78\left(x^d - x\right) - 16\dot{x}$$

In the next section the controller will be tested for robustness.

9.4 Robustness

Once the controller has been designed, it is important to analyse its performance with respect to changes in model parameters. As there will always be slight measurement inaccuracies when calculating variables such as the mass of the body and its mass moment of inertia. This is particularly true in calculating the moment of inertia where simplifications were made in modelling the system. If the weight of the robot is not evenly distributed then the actual mass moment of inertia could be different to that which is used in the model. However, a good controller should still be able to deal with changes in parameters.

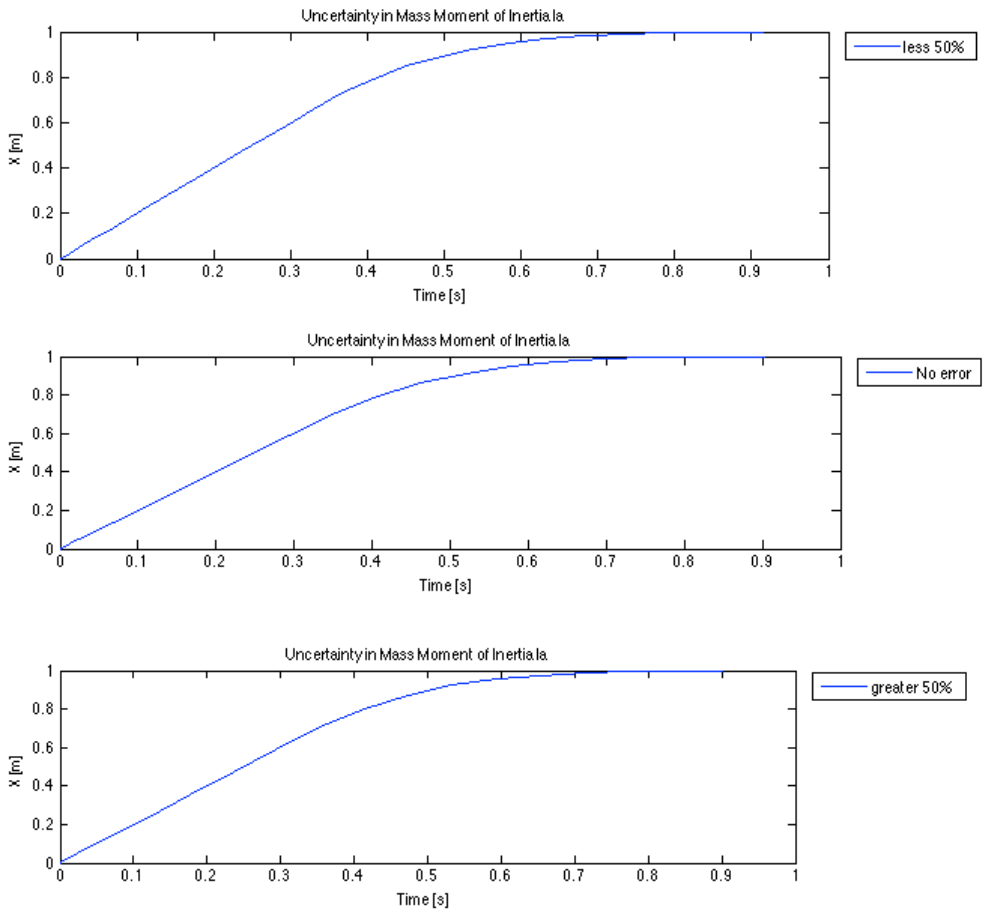


Figure 9.7: The effect of changes in mass moment of inertia on controller performance

The figure above shows the performance of the controller for variation in mass moment of inertia. The method used to simulate these results was that the value used to calculate the feedback controller was kept at what was originally calculated, while the value used in the system dynamics block was increased or decreased by 50%. The resulting response was almost identical for changes in mass moment of inertia. This justified the simplification of the model and negated the need to perform in-depth experiments, such as the pendulum method, to derive a more accurate evaluation of the mass moment of inertia.

The test of robustness was done on changes in body mass and while there is less chance of inaccuracies in calculating this value there are still measurement errors involved. Also it is useful to able be to make modifications to the hardware design without significantly changing the expected performance of the controller.

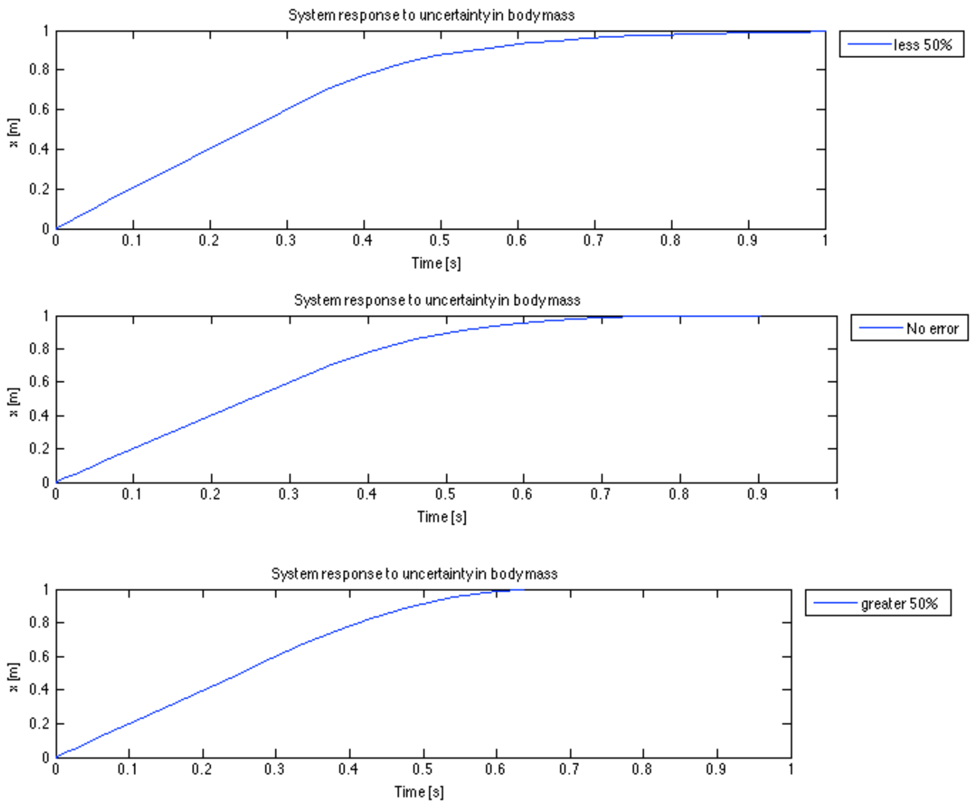


Figure 9.8: The effect of variations in body mass on controller performance

The result of changing the body mass still gave an acceptable a response. In the case where the weight is reduced settling time was slightly slower by about 0.3s compared to the case where the weight is increased. Beside this the response still reached the setpoint in both cases and gave a zero steady error.

There next variable that was investigated was the dragline force. The brushed dc motor will experience some friction due to the brushes as well as from the magnetic field from the permanent magnets interfering with the rotor. This means that even when there is no voltage supplied to the terminals of the motor there will still be a small amount of dragline force caused by this friction. The friction was modelled as constant force on the dragline.

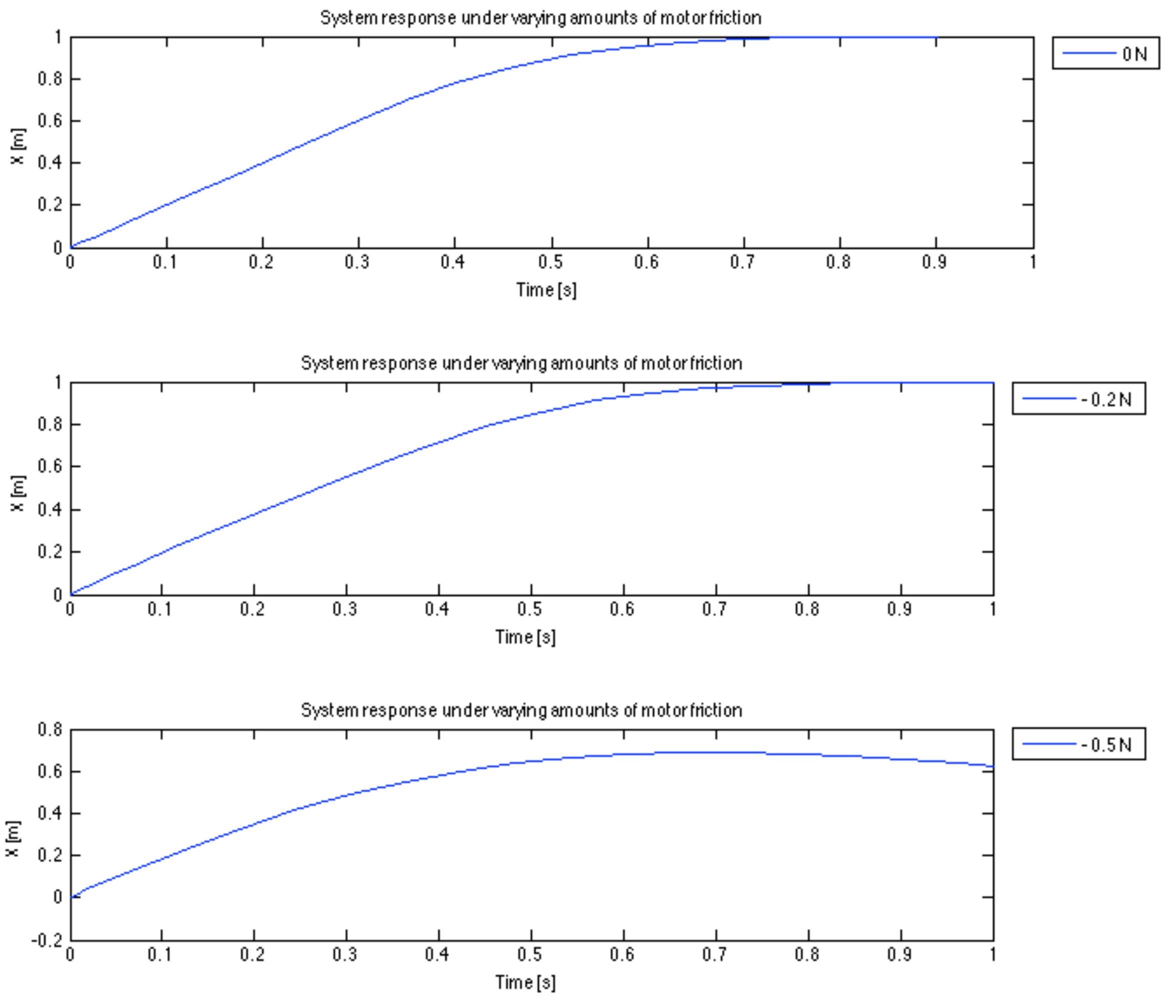


Figure 9.9: The effect of motor friction on the system response

As can be seen from the figure above for values up to – 0.2N the system still responds as expected with the only difference being a marginally longer settling time. However, as the force increases above this value there is too much energy taken out of system which results in the desired setpoint not being obtained. This should not be a problem as the friction of the motor is not expected to be greater than -0.1N and the controller is robust enough to deal with values in this range.

9.5 Simulations

In this section the response of the system will be tested for different initial conditions and the resulting simulations analysed.

9.5.1 System block diagram

The figure below shows the block diagram of the closed loop system.

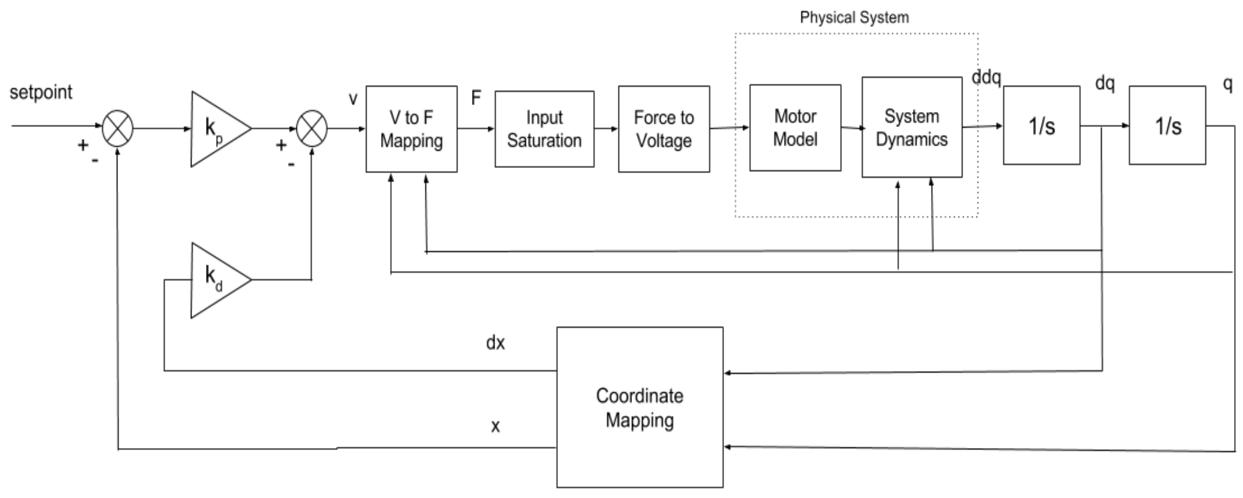


Figure 9.10: Block Diagram of system simulated using SIMULINK where ddq is the acceleration of the states, dq is the velocity of the states, q is the position of the states, dx is the velocity in the x direction, x is the position of the CoM in the x direction, Kp and Kd are the proportional and derivative gains respectively, v is the control action and F is the force command.

The V to F mapping block converts the control action to a force using the equation derived in section 9.3.2. This force is then passed through a saturation block which ensures that the force stays between the physical limits of the system. The force command is then inputted into the system dynamics block which outputs the state accelerations. The coordinate mapping block converts the general coordinates to the x,y thA coordinate system as it is the x distance that is quantity that is required to be controlled.

9.5.2 Simulations of controller

In order to test the effectiveness of the controller, the output and the required force were plotted for various initial velocities. What was discovered was that although the controller was designed for a settling time of 0.5s, the actual time to reach the setpoint was often much longer. This was the result of the force only being able to be used as a brake. So the actual time to reach the setpoint from launch was partly determined by the distance needed to be travelled and the magnitude of the initial velocity and launch angle. However, it can still be said that the controller achieved the design requirements as it was found that the setpoint was consistently reached 0.5s from the moment the control action was applied to the system.

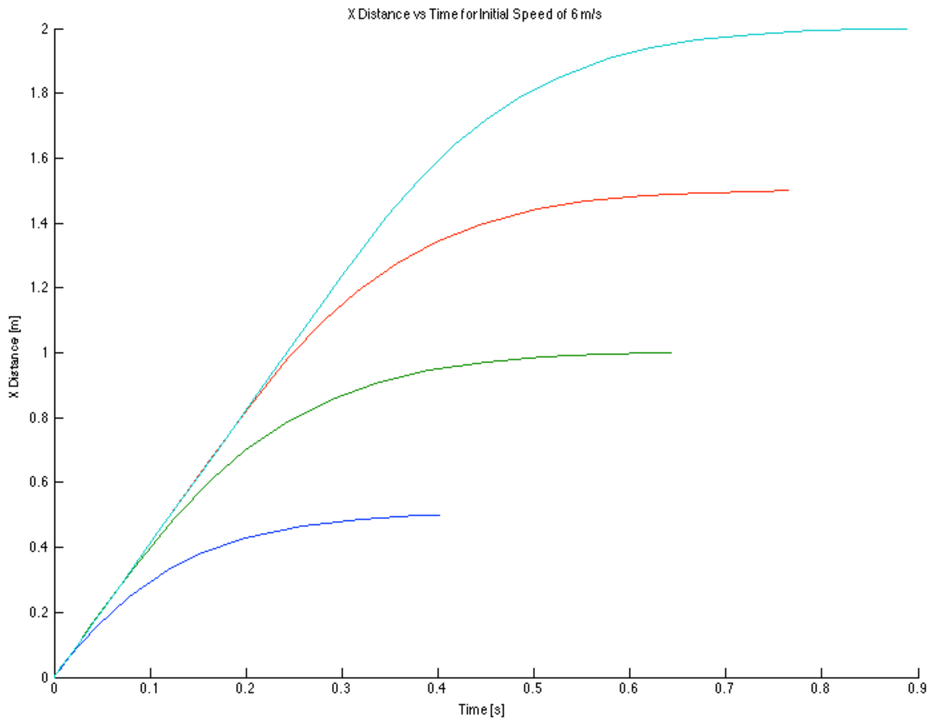


Figure 9.11: X distance vs. time for setpoints 0.5, 1, 1.5 and 2m for initial launch velocity of 6 m/s

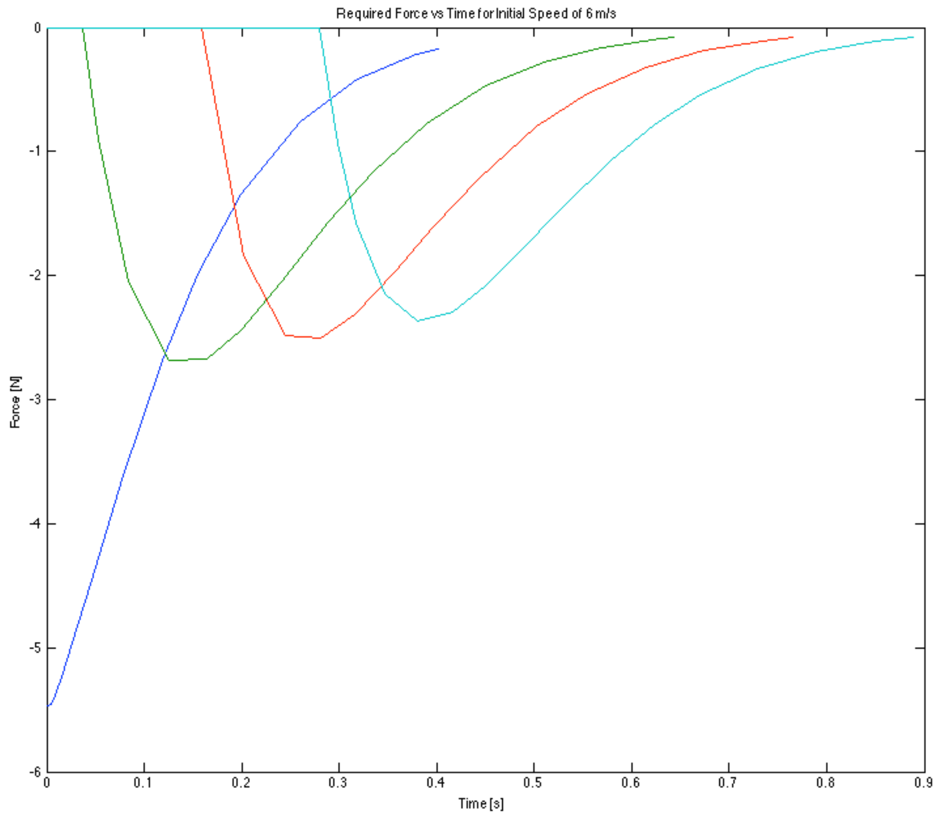


Figure 9.12: Force command vs. time for setpoints 0.5, 1, 1.5 and 2m for initial launch velocity of 6 m/s

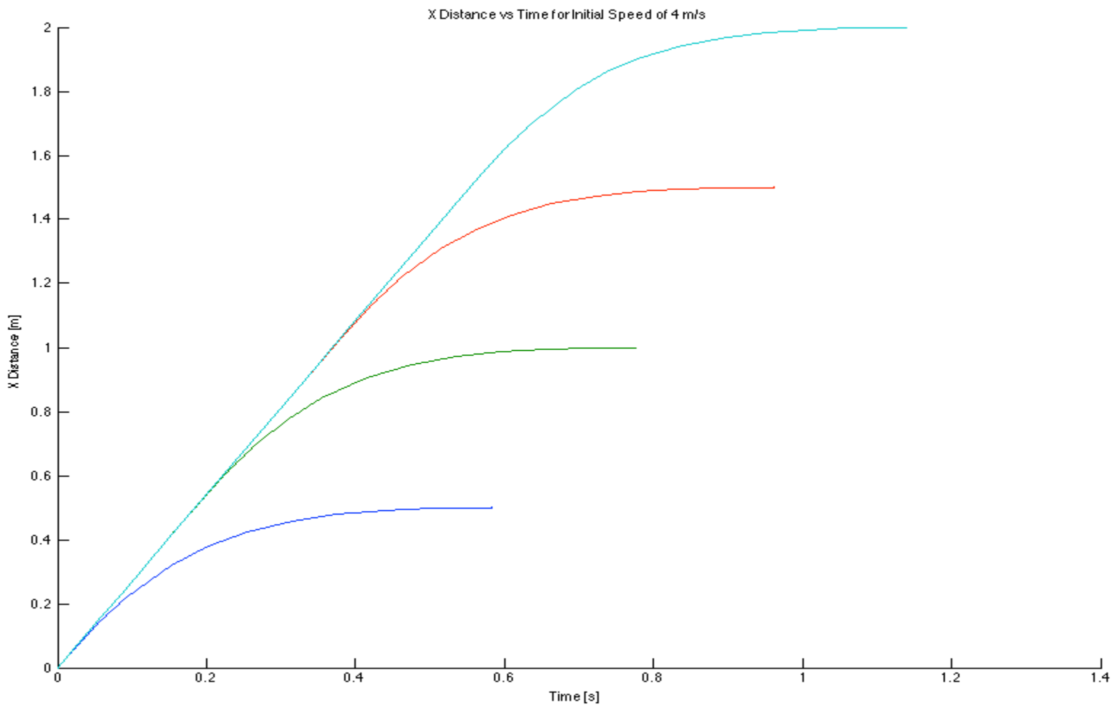


Figure 9.13: X distance vs. time for setpoints of 0.5, 1, 1.5 and 2m and an initial launch velocity of 4 m/s

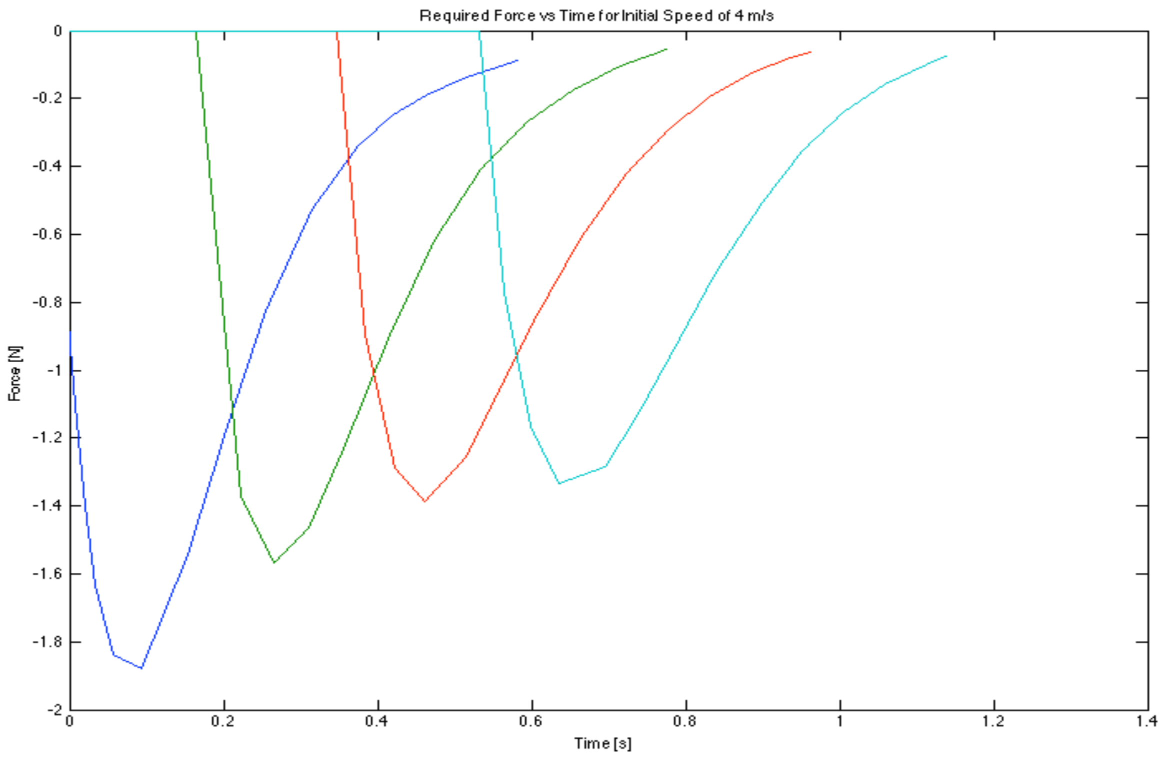


Figure 9.14: Force command vs. time for setpoints 0.5, 1, 1.5 and 2m for initial launch velocity of 4 m/s

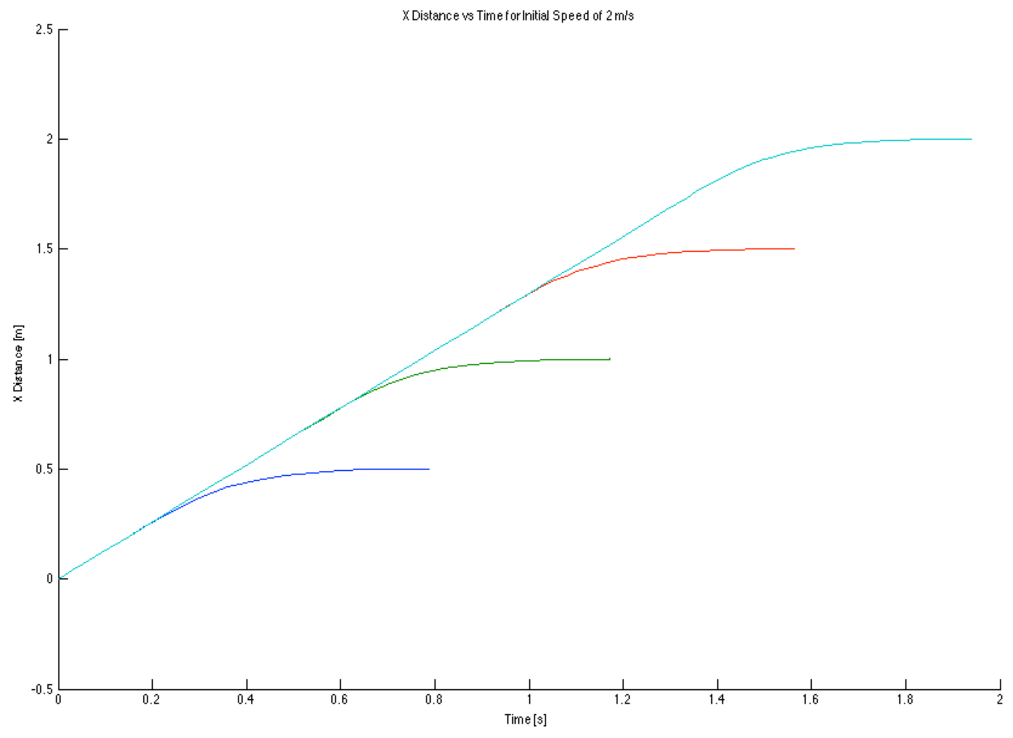


Figure 9.15: X distance vs. time for setpoints 0.5, 1, 1.5 and 2m for initial launch velocity of 2 m/s

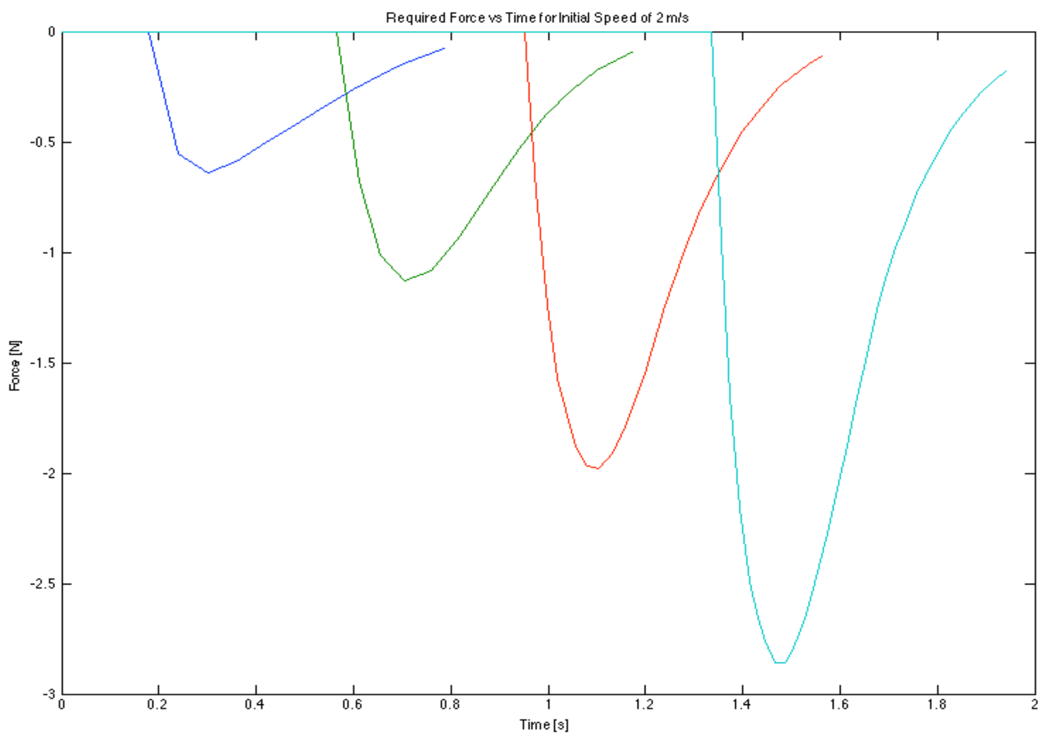


Figure 9.16: Force command vs. time for setpoints 0.5, 1, 1.5 and 2m for initial launch velocity of 2 m/s

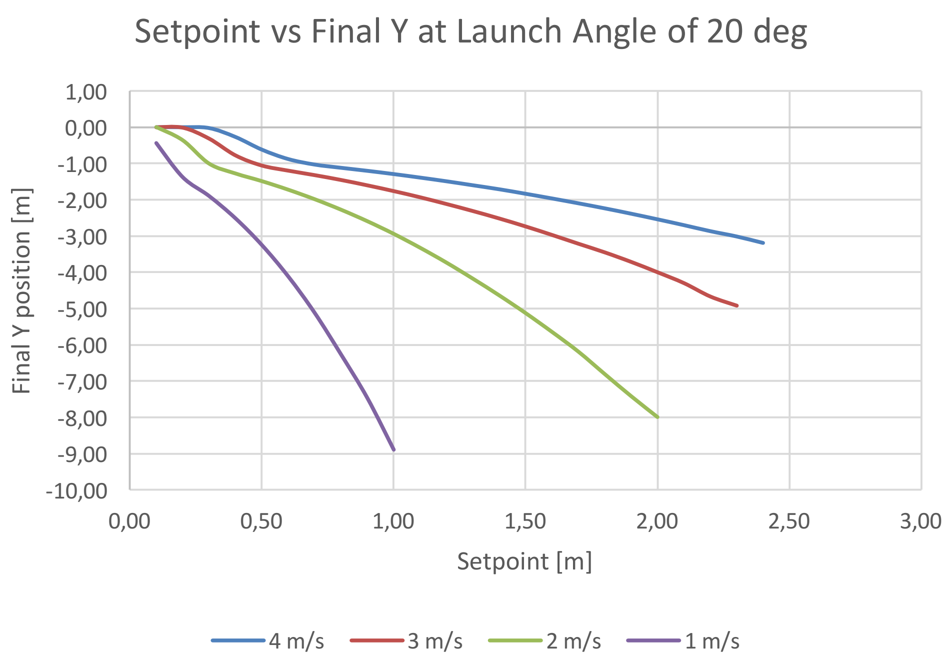


Figure 9.17: Relationship between the final y position and setpoint for a launch angle of 20 degrees and a range of initial velocities

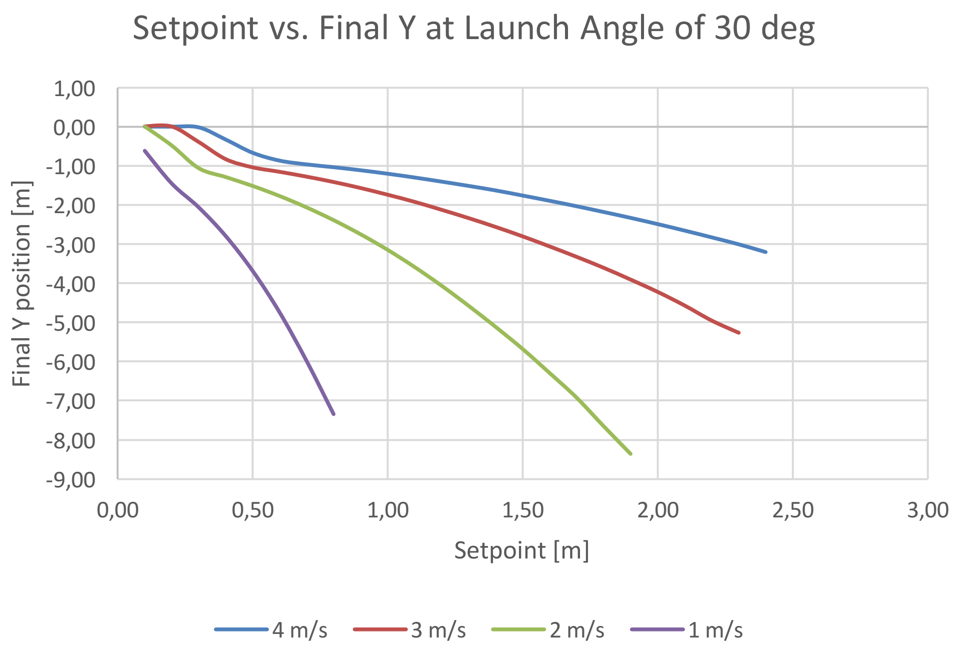


Figure 9.18: Relationship between the final y position and setpoint for a launch angle of 30 degrees and a range of initial velocities

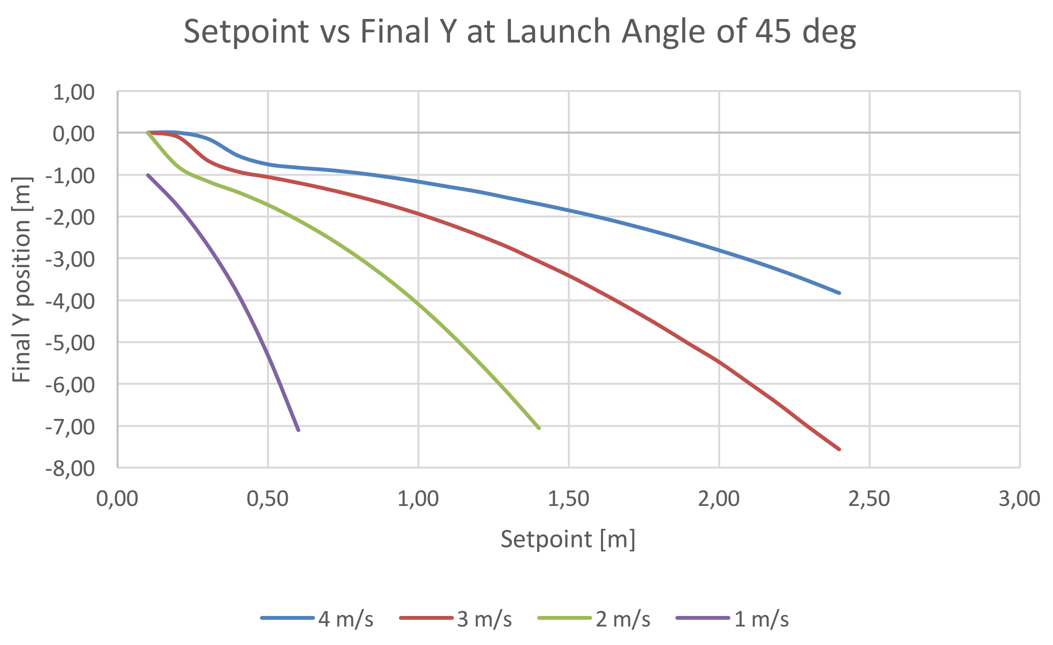


Figure 9.19: Relationship between the final y position and setpoint for a launch angle of 45 degrees and a range of initial velocities

From figures above it is that the controller performed well for a range of different setpoints at different initial velocities. There was zero steady error, no overshoot, and a settling time of around 0.5s from the point at which the control action was initialized.

10. Testing and Implementation

This section will deal with the controller implementation and the testing of the dragline spider platform. First, the digital implementation of the controller will be discussed.

10.1 Controller Implementation

In order to implement the controller, a control method was written where all the necessary calculations for implementation were performed. The flowchart for the control method is shown below.

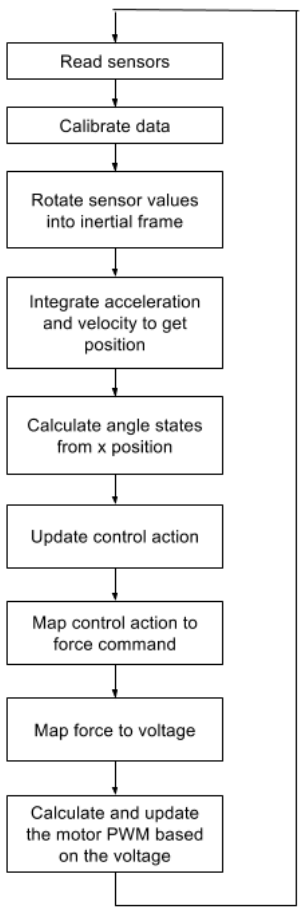

Read sensors
Calibrate data
Rotate sensor values into inertial frame
Integrate acceleration and velocity to get position
Calculate angle states from x position
Update control action
Map control action to force command
Map force to voltage
Calculate and update the motor PWM based on the voltage

Figure 10.1: Flowchart for control method

Firstly, the data from the accelerometer and gyroscope is assigned to global variables. This data is then calibrated using the equations calculated in the chapter on calibration. Since all the data that is read from the sensors is with respect to the body frame, they must be rotated into the inertial frame using the Euler angles method described in the theory development chapter. This is necessary as the calculations for determining the states of the system are all in terms of the inertial frame. The next step was to determine the velocity and position of the robot in the x direction. This was done by integrating the acceleration with respect to inertial frame of the body in the x direction.
For the implementation of PFL all the states must be known. In the physical system, the states that could be measured were the dragline L , the x position, and the body angle relative to the horizontal, θ_A. The states that were not measurable directly were θ_0 and θ_s and their derivatives. Fortunately, these could be derived from the measurable states.

$$\theta_0 = \operatorname{acos}\left(\frac{2\left(x - Lcos(\theta_A - \pi)\right)}{l}\right)$$

$$\theta_s = \theta_0 - \theta_A + \pi$$

$$\dot{\theta_0} = \frac{\dot{x} - \dot{L}\cos(\theta_0) - lsin(\theta_0 - \theta_s)}{-Lsin(\theta_0) - lsin(\theta_0 - \theta_s)}$$

$$\dot{\theta}_s = \dot{\theta}_0 - \dot{\theta}_A$$

Once these states had been calculated the control action was updated. The control action was then mapped to a force command. From the force command, a voltage was determined using the equation calculated in the motor calibration section. The voltage was then converted to a PWM duty cycle which was used to drive the motor. This method was run in an infinite while loop to ensure that the motor PWM was continually updated. For the integration, a sample time of 0.01s was used as this is the speed at which the data was read from the sensors.

10.2 Problems experienced during testing

Due to various hardware and software issues the amount of testing of the physical platform that was possible was not as extensive as what was hoped to be achieved at the start of the project. Some of these problems will be discussed next.

The first issue that caused setbacks in the project timeline was the malfunctioning of the INEMO boards. Several of the INEMO boards used during the project developed an issue where they were unable to connect to the PC via the ST-LINK to flash new code onto the microprocessor. This was despite the fact they were still able to send sensor data via the XBEEs to the logging application. As the new INEMO boards were readily available, time was spent attempting to debug the problem. This included continuity tests on all the pins, as well as voltage tests. However, nothing out of the ordinary was detected. Although the reason behind this issue was never fully understood, one possible theory is that either the USB ports on the laptop or the STM32F4 discovery board that was being used to interface between the laptop and the INEMO, malfunctioned causing current spike to damage some of the internal components on STM32F1 processor.

The other issue that caused delays was the reliability of the communication network that was used to log the sensor data for further analysis. The problem was that the application used to log data would receive the first packet but then after that, multiple CRC errors would show up and the data stopped logging. As the application was written by a third party, the lack of intimate knowledge of the code made debugging challenging. As a result this issue was not able to be resolved. Instead, it was decided to analyse the tests purely using video footage.

However, another problem that was experienced during testing was that when attempting to test the controller, the motor would receive full power as soon as it left the launcher. This caused the robot to come to a sudden halt well short of the desired setpoint. One possible cause of this could have been errors involved in double integrating the x acceleration to obtain position. Any measurement error associated with the acceleration quickly escalates when integrated, which could have meant that the robot assumed it had reached the setpoint causing the motor to power on, when in reality it was still some distance from the setpoint.

Due to the aforementioned problems that were experienced, and the time constraints on this project, this meant that results obtained from testing the physical system were limited. However, from the simulations that were done the implementation of a PFL controller should be a feasible option for controlling the X position of the robot. It is these simulations that will be discussed further in the next section.

11. Discussion

11.1 Discussion of simulations

Based on the simulations done on the controller in Chapter 9, the use of a PFL controller to achieve precise landings does seem like a feasible option. The simulations did produce some differences to the results put forward in the study by Chen, et al. that are worth highlighting.

In terms of the required force, the magnitudes that were necessary to achieve the desired settling time were larger than what was put forward by Chen, et al. The magnitudes of force on the spiders' dragline was quoted as being in the region of 0.26 to 0.4 of the body weight of the spider [1]. While from the simulations, in extreme cases the force could be as high as four times the body weight and on average between one to two times the body weight.

One reason for this is that, in order for the robot to reach the setpoint, the dragline force is required to change the momentum of the body or in other words apply an impulse. The larger the mass and higher the velocity, the larger the force is needed to bring about the change in momentum. In the case of the spider, the force is applied over the entire duration of the jump which reduces the size of the impulse and as a result the magnitude of the force on the dragline is also reduced. Whereas, the controller that was designed in this project was optimized for speed, meaning that the dragline force was applied over a shorter relative time frame increasing the impulse that is produced.

Another aspect that is worth noting is that although the controller performed well in reaching the setpoint, the corresponding Y position varied considerably depending on the initial conditions. This limits the possible achievable setpoints for a given test platform setup. For example, if the launcher is situated 1m above the ground and it takes 1s the reach the desired setpoint, by this time the robot may have already hit the ground making this setpoint impossible to achieve.

12. Conclusions

Although, due to time constraints on the project, the testing of the platform was limited, the results obtained through simulation show that the controller designed using PFL represents a feasible option to achieve precise landing with the use of a dragline.

This report has shown that the dragline could very well be a viable option for stabilizing the landings of mobile robotics in the future.

In the modelling chapter, a dynamic model of the system was determined and verified against the model developed by Stacey Shield [7]. Although some simplifications had to be made in the modelling of the system, the robustness tests performed on the controller showed that the desired response was still achievable for variation in the system parameters.

The results from the hardware chapter showed that the choice of motor suitable for the task was not a simple one. Balancing the torque requirements while trying to keep the overall weight of the system as low as possible was not a straight forward task, as the weight of DC motors generally increases with torque output. However, by including spool radius as part of the design parameters a satisfactory motor was able to be selected.

In the controller design chapter, different linearization techniques were investigated. Taylor series linearization was found to be unsuitable for this system, giving poor approximations of the dynamics. The next option that was explored was feedback linearization. Due to the underactuated nature of the system, full input-state or input-output feedback could not be used, so partial feedback linearization had to applied. This proved to be successful as it gave a good response and was fairly straight forward to implement. The only downside being that knowledge of all the states of the system is necessary for implementation.

13. Recommendations

In this chapter some recommendations for future work will be discussed.

13.1 Optimal Controller

The controller that was designed in this project was able to meet the objectives set out at the start of the project. However, the design can always be improved especially in terms of minimizing the force command. By using a cost function in a LQR or MPC controller the force input could be weighted appropriately to reduce its magnitude. This means it is possible that a smaller, lighter motor could be used.

13.2 Kalman Filter

To reduce the unbounded error associated with integrating the accelerometer readings to obtain position an Extended Kalman Filter could be implemented [12]. The design and implementation of the Kalman Filter would bound the integration errors, leading more accurate calculations of the x position of the robot.

13.3 Gain Scheduling

In this project there was little emphasis on ensuring the that the robot reached the setpoint before it hit the ground. However, given a particular testing setup with the launcher installed at a known height off the ground a gain scheduling approach could be used. From the initial conditions the time to taken to hit the ground could be calculated. It is then known that the settling time for the controller must be less than this and appropriate gains could be calculated automatically so that this target is met.

13.4 Alternative methods of force control

In order to reduce the weight of the robot alternatives to the brushed DC motor could be investigated. Options for research could include using a clamping method driven by a servo motor or if an accurate motor controller could be design, a brushless motor would also be effective.

14. List of References

[1] Y.-K. Chen, C.-P. Liao, F.-Y. Tsai, and K.-J. Chi, “More than a safety line: jump-stabilizing silk of salticids,” *J. R. Soc. Interface*, vol. 10, no. 87, pp. 20130572–20130572, Aug. 2013.

[2] A. von Gleich, C. Pade, U. Petschow, and E. Pissarskoi, *Potentials and Trends in Biomimetics*. Springer Science & Business Media, 2010.

[3] “Biomimetics: Amazing Animal Technology,” *HubPages*. [Online]. Available: http://buffoon.hubpages.com/hub/biomimetics-biomimicry. [Accessed: 14-Oct-2015].

[4] “BionicKangaroo|Festo.” [Online]. Available: https://www.festo.com/group/en/cms/10219.htm. [Accessed: 14-Oct-2015].

[5] T. Libby, T. Y. Moore, E. Chang-Siu, D. Li, D. J. Cohen, A. Jusufi, and R. J. Full, “Tail-assisted pitch control in lizards, robots and dinosaurs,” *Nature*, vol. 481, no. 7380, pp. 181–184, Jan. 2012.

[6] M. Burrows, D. A. Cullen, M. Dorosenko, and G. P. Sutton, “Mantises exchange angular momentum between three rotating body parts to jump precisely to targets,” *Curr. Biol.*, vol. 25, no. 6, pp. 786–789, 2015.

[7] Stacey Shield, “Dragline-Assisted Pitch Angle Righting,” Undergraduate Thesis, University of Cape Town, 2014.

[8] “MIT CSAIL Research Abstracts.” [Online]. Available: http://publications.csail.mit.edu/abstracts/abstracts06/littledog/littledog.html. [Accessed: 14-Oct-2015].

[9] Russell Tedrake, “Readings | Underactuated Robotics | Electrical Engineering and Computer Science | MIT OpenCourseWare.” [Online]. Available: http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-832-underactuated-robotics-spring-2009/readings/. [Accessed: 10-Oct-2015].

[10] M. W. Spong, “Partial feedback linearization of underactuated mechanical systems,” in *Proceedings of the IEEE/RSJ/GI International Conference on Intelligent Robots and Systems ’94. “Advanced Robotic Systems and the Real World”, IROS ’94*, 1994, vol. 1, pp. 314–321 vol.1.

[11] “Lagrange Equations | Engineering Dynamics | Mechanical Engineering | MIT OpenCourseWare.” [Online]. Available: http://ocw.mit.edu/courses/mechanical-engineering/2-003sc-engineering-dynamics-fall-2011/lagrange-equations/#?w=535. [Accessed: 14-Oct-2015].

[12] Justin Coetser, “Low-cost Inertial Navigation System,” Undergraduate Thesis, University of Cape Town, 2013.

[13] H. Alemi Ardakani and T. J. Bridges, “Review of the 3-2-1 Euler Angles: a yaw-pitch-roll sequence.” Unversity of Surrey, 2010.

[14] “Understanding Euler Angles | CH Robotics.” .

[15] M. S. Tsoeu, “Lectures Notes: EEE4093F - Model Predictive Control (MPC).” University of Cape Town.

[16] R. E. Deakin, “Linearization Using Taylor’s Theorem and the Derivation of Some Common Survrying Observation Equations.” RMIT University, 2005.

[17] A. Shkolnik and R. Tedrake, “High-dimensional underactuated motion planning via task space control,” in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 2008, pp. 3762–3768.

[18] J. K. Vandiver, “2.003SC Engineering Dynamics.” Massechusetts Institute of Technology: MIT OpenCourseWare, Fall-2009.

[19] STMicroelectronics, “STM32F103xx Reference Manual.” Jun-2014.

[20] STMicroelectronics, “iNEMO system-on-board Datasheet.” Oct-2013.

[21] International Rectifier, “IRL520N Datasheet.” May-1998.

[22] “Pololu - Magnetic Encoder Pair Kit for Micro Metal Gearmotors, 12 CPR, 2.7-18V.” [Online]. Available: https://www.pololu.com/product/2598. [Accessed: 14-Oct-2015].

[23] Digi, “XBee RF Modules.” .

[24] Fairchild Semiconductory, “QRD1114 Reflective Object Sensor.” Jun-2013.

[25] “ZIPPY Flightmax 350mAh 2S 20C,” *HobbyKing Store*. [Online]. Available: http://hobbyking.com/hobbyking/store/uh_viewitem.asp?idproduct=9028. [Accessed: 14-Oct-2015].

15. Appendix A: Supplementary Material

This appendix describes the material can be found on the CD that was submitted with this document.

15.1 Videos

The CD contains a few videos which show the testing setup that was used.

15.2 MATLAB code

There is also a folder for the MATLAB scripts used during the project from system modelling to the generation of graphs.

15.3 Datasheets

The datasheets from the List of References has been included.

15.4 C code

The project folder used to implement the controller has also been included in the folder called iNEMOproject.

16. EBE Faculty: Assessment of Ethics in Research Projects

Any person planning to undertake research in the Faculty of Engineering and the Built Environment at the University of Cape Town is required to complete this form before collecting or analysing data. When completed it should be submitted to the supervisor (where applicable) and from there to the Head of Department. If any of the questions below have been answered YES, and the applicant is NOT a fourth year student, the Head should forward this form for approval by the Faculty EIR committee: submit to Ms Zulpha Geyer (Zulpha.Geyer@uct.ac.za; Chem Eng Building, Ph 021 650 4791).Students must include a copy of the completed form with the final year project when it is submitted for examination.

Name of Principal Researcher/Student: ____________ **Department:** ELECTRICAL ENGINEERING

If a Student: YES **Degree:** ____________ **Supervisor:** ____________

If a Research Contract indicate source of funding/sponsorship: ____________

Research Project Title: ____________

Overview of ethics issues in your research project:

Question 1: Is there a possibility that your research could cause harm to a third party (i.e. a person not involved in your project)?	YES	NO
Question 2: Is your research making use of human subjects as sources of data? If your answer is YES, please complete Addendum 2.	YES	NO
Question 3: Does your research involve the participation of or provision of services to communities? If your answer is YES, please complete Addendum 3.	YES	NO
Question 4: If your research is sponsored, is there any potential for conflicts of interest? If your answer is YES, please complete Addendum 4.	YES	NO

If you have answered YES to any of the above questions, please append a copy of your research proposal, as well as any interview schedules or questionnaires (Addendum 1) and please complete further addenda as appropriate.

I hereby undertake to carry out my research in such a way that

- there is no apparent legal objection to the nature or the method of research; and
- the research will not compromise staff or students or the other responsibilities of the University;
- the stated objective will be achieved, and the findings will have a high degree of validity;
- limitations and alternative interpretations will be considered;
- the findings could be subject to peer review and publicly available; and
- I will comply with the conventions of copyright and avoid any practice that would constitute plagiarism.

Signed by:

	Full name and signature	**Date**
Principal Researcher/Student:		14 October 2015

This application is approved by:

Supervisor (if applicable):		14 October 2015
HOD (or delegated nominee): Final authority for all assessments with NO to all questions and for all undergraduate research.	**Janine Buxey**	14 October 2015
Chair : Faculty EIR Committee For applicants other than undergraduate students who have answered YES to any of the above		

ADDENDUM 1:
Please append a copy of the research proposal here, as well as any interview schedules or questionnaires:

ADDENDUM 2: To be completed if you answered YES to Question 2:

It is assumed that you have read the UCT Code for Research involving Human Subjects (available at http://web.uct.ac.za/depts/educate/download/uctcodeforresearchinvolvinghumansubjects.pdf) in order to be able to answer the questions in this addendum.

2.1 Does the research discriminate against participation by individuals, or differentiate between participants, on the grounds of gender, race or ethnic group, age range, religion, income, handicap, illness or any similar classification?	YES	NO
2.2 Does the research require the participation of socially or physically vulnerable people (children, aged, disabled, etc) or legally restricted groups?	YES	NO
2.3 Will you not be able to secure the informed consent of all participants in the research? (In the case of children, will you not be able to obtain the consent of their guardians or parents?)	YES	NO
2.4 Will any confidential data be collected or will identifiable records of individuals be kept?	YES	NO
2.5 In reporting on this research is there any possibility that you will not be able to keep the identities of the individuals involved anonymous?	YES	NO
2.6 Are there any foreseeable risks of physical, psychological or social harm to participants that might occur in the course of the research?	YES	NO
2.7 Does the research include making payments or giving gifts to any participants?	YES	NO

If you have answered YES to any of these questions, please describe below how you plan to address these issues:

ADDENDUM 3: To be completed if you answered YES to Question 3:

3.1 Is the community expected to make decisions for, during or based on the research?	YES	NO
3.2 At the end of the research will any economic or social process be terminated or left unsupported, or equipment or facilities used in the research be recovered from the participants or community?	YES	NO
3.3 Will any service be provided at a level below the generally accepted standards?	YES	NO

If you have answered YES to any of these questions, please describe below how you plan to address these issues:

ADDENDUM 4: To be completed if you answered YES to Question 4

4.1 Is there any existing or potential conflict of interest between a research sponsor, academic supervisor, other researchers or participants?	YES	NO
4.2 Will information that reveals the identity of participants be supplied to a research sponsor, other than with the permission of the individuals?	YES	NO
4.3 Does the proposed research potentially conflict with the research of any other individual or group within the University?	YES	NO

If you have answered YES to any of these questions, please describe below how you plan to address these issues: