

Faculty of Engineering and Built Environment

School of Engineering

Mechanical Engineering Project B

Semester 2 - 2020



PROJECT TITLE

Disk on Disk System

NAME

Timothy Dunn

SUPERVISOR

Joel Ferguson



FINAL YEAR PROJECT

I. Dot-Point Summary

- I applied the Euler-Lagrange method to develop a mathematical model
- I designed a state estimator suitable for use with mechatronic control
- I performed a simulation of a mechanical system using a mathematical model
- I designed an experiment apparatus using results of a simulation
- I designed an electrical system to control actuators in an experiment
- I implemented control of said actuators using an embedded system
- I implemented a method of feature recognition to obtain measurements
- I researched methods of image capture
- I performed system identification and developed a smooth friction model to model non-linear dynamics suitable for use with numerical solving tools
- I implemented a novel method of control allocation
- I transformed the Euler- Lagrange model into a Hamiltonian model for a new control allocation scheme
- I tested camera arrangements and configurations to determine suitability for low-latency image capture
- I determined the experiment would not exhibit stability at achievable control frequencies due to image acquisition latency, and proved this with experimental results

Declaration

I declare that this thesis is my own work unless otherwise acknowledged and is in accordance with the University's academic integrity policy available from the Policy Library on the web at <http://www.newcastle.edu.au/policylibrary/000608.html>

I certify that this assessment item has not been submitted previously for academic credit in this or any other course.

I acknowledge that the Faculty of Engineering may, for the purpose of assessing this thesis:

- Reproduce this thesis and provide a copy to another member of the Faculty; and/or
- Communicate a copy of this assessment item to a plagiarism checking service (which may then retain a copy of the item on its database for the purpose of future plagiarism checking).
- Submit the assessment item to other forms of plagiarism checking.
- Where appropriate, provide a copy of this thesis to other students where that student is working in/on a related project.

I further acknowledge that the Faculty of Engineering may, for the purpose of sector wide Quality Assurance:

- Reproduce this thesis to provide a copy to a member of another engineering faculty.

I certify that the electronic versions of this thesis I have submitted are identical to this hardcopy version. Furthermore, I have checked that the electronic version is complete and intact on the submitted location.

Student: Timothy Dunn

Signature: 


Lab Cleanliness Declaration

I, Timothy Dunn having completed the project Disk on Disk System, and using laboratory spaces (none), hereby submit that the area has been cleaned to level equal to or better than when I commenced my work. All items I borrowed have been returned, and the area is now suitable for inspection and sign off by the appropriate laboratory manager.

Signed: 

Date: 9/11/20

Appropriate Manager

Signed: 

Date: 11/11/20

Originality Report

Timothy Dunn - C3234207 - FYP Part B Report

ORIGINALITY REPORT

6%

SIMILARITY INDEX

3%

INTERNET SOURCES

3%

PUBLICATIONS

4%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to University of Newcastle

Student Paper

2%

2

hdl.handle.net

Internet Source

1%

3

my.nps.edu

Internet Source

<1%

4

W. E. Dixon. "Asymptotic Tracking for Uncertain Dynamic Systems via a Multilayer NN Feedforward and RISE Feedback Control Structure", 2007 American Control Conference, 07/2007

Publication

<1%

5

Ruoqi Wei, Cesar Garcia, Ahmed El-Sayed, Viyaleta Peterson, Ausif Mahmood. "Variations in Variational Autoencoders - A Comparative Evaluation", IEEE Access, 2020

Publication

<1%

Abstract

This project has the objective of researching, modelling, simulating, and implementing control techniques on a rolling-balancing mechanical system. By applying the Euler-Lagrange method to develop a mathematical model, which is then implemented in the Simulink toolbox and controlled using both the Linear Quadratic Regulator and Model Predictive Controller, this objective is partially achieved. An investigation into building control schemes around effort source actuators and flow source actuators is undertaken. Promising results were achieved using Proportional-Integral-Derivative (PID) control allocation when treating the DC motor as a source of flow, an unconventional approach that negates the requirement to perform system identification. The last objective, which is unable to be completed, is the implementation of the control techniques using the designed experiment apparatus. During the implementation phase, a significant and unavoidable delay in image acquisition was discovered which reduced achievable control and measurement rates to frequencies which are inherently unstable.

Underactuated mechanical systems are extremely common engineering scenarios with no general solution. Systems such as the disk-on-disk (DoD) present further complications beyond under-actuation. The system is suitable for Linear-Quadratic Regulator (LQR) control schemes to be used. Reference tracking can also be employed using reference feedforward structures.

The DoD system presents complications when measurements of system states is required, particularly the states associated with the unactuated disk. A contactless method of measuring position is employed, utilising colour saturation and blob analysis. The uncertainty associated with the measurements is accounted for with the Kalman Filter.

Acknowledgements

I would like to thank Joel Ferguson for his excellent work as a supervisor. His guidance, support, and encouragement were greatly appreciated, as was his dismay at my use of PID.

I would also like to thank Alex Fairclough for his constant enthusiasm and support throughout this project. The importance of an engaged listener for a conversation about even the most trivial aspects of my work can not be understated.

Table of Contents

I.	Dot-Point Summary	2
	Abstract	6
	Acknowledgements	7
II.	List of Figures	10
1	Introduction	12
2	Background	15
2.1	Euler-Lagrange Method	15
2.2	Underactuation	16
2.3	Estimation and Control	17
2.3.1	Linear-Quadratic Regulator	17
2.3.2	Model Predictive Control	19
2.4	Sensing Technologies	21
2.5	The Kalman Filter	21
3	Modelling and Simulation	24
3.1	The Disk-on-Disk System	24
3.2	Simulation	27
4	Estimation and Control	29
4.1	Controller Design	29
4.1.1	Torque Control	30
4.1.2	Velocity Control	30
4.1.3	Comparison of Torque Control and Velocity Control	31
4.1.4	Comparison of LQG and MPC	33
4.2	State estimation	33
4.2.1	Computer Vision	34

4.2.2	Implementation of the Kalman Filter.....	39
5	Experiment Apparatus Design	41
5.1	Mechanical Design	41
5.2	Electrical Design	45
5.3	Safety Considerations.....	45
6	Implementation.....	46
6.1	DC Motor System Identification	46
6.1.1	Estimation of Resistance and Motor Constant	47
6.1.2	Estimation of Armature Inductance	48
6.1.3	Estimation of Torque Curve	50
6.1.4	Estimation of Friction.....	52
6.2	Control Allocation	59
6.2.1	Torque Control	59
6.2.2	Velocity Control	61
6.3	Implementation of Computer Vision Video Stream	62
6.3.1	Microsoft Webcam	63
6.3.2	Raspberry Pi Camera.....	65
6.3.3	Measuring Image Acquisition Latency.....	66
7	Results.....	69
8	Conclusion and Recommendations	72
9	References	73
	Appendix A – Time Log Graphic	75
	Appendix B – Code, Drawings and Documents Generated.....	76

II. List of Figures

Figure 1. The disk-on-disk system.....	12
Figure 2. A robotic gripper [1].....	13
Figure 3. Examples of prehensile and non-prehensile manipulations [2].	13
Figure 4. Linear-Quadratic Regulator with reference feedforward control structure [8].....	19
Figure 5. Layout of ideal DoD system.	25
Figure 6. Layout of Simulink model.	27
Figure 7. Plots of hand and object angle and angular velocity.	28
Figure 8. Torque control in simulation.	32
Figure 9. Velocity control in simulation.	32
Figure 10. Comparison of LQG and MPC controllers.....	33
Figure 11. Checkerboard pattern used for camera calibration.....	34
Figure 12. The original distorted image (left) and the undistorted image after estimation of camera parameters (right).....	35
Figure 13. Reprojection error for each image in the calibration set.	36
Figure 14. HSV version of test image.	37
Figure 15. Segmented circles.	37
Figure 16. Result of feature recognition using saturation thresholding.	38
Figure 17. Performance of Kalman Filter at 100Hz.	40
Figure 18. Performance of Kalman Filter at 20Hz.....	40
Figure 19. Isometric view of experiment apparatus general assembly.	42
Figure 20. Experiment Apparatus.	43
Figure 21. Angle, angular velocity and demanded torque plots for applied control law at $T=0.01s$. 44	
Figure 22. Angle, angular velocity and demanded torque plots for applied control law at $T=0.15s$. 44	
Figure 23. Standard permanent magnet brushed DC motor model [15].	46

Figure 24. Result of first stage parameter estimation.	48
Figure 25. Result of second stage parameter estimation.	49
Figure 26. Current vs resistance for nonlinear resistance model.....	50
Figure 27. Experiment setup to determine motor torque characteristics.	51
Figure 28. Current-Torque curve.....	51
Figure 29. Parameter estimation using viscous friction model.	52
Figure 30. Stribeck friction models.....	53
Figure 31. Smooth alternatives to the sign(x) function.....	54
Figure 32. Parameter estimation using combined Stribeck-viscous friction model.....	55
Figure 33. Friction profiles of component models and total model.....	56
Figure 34. Parameter estimation using continuously differentiable friction model.	56
Figure 35. Smooth and non-smooth Heaviside functions.....	57
Figure 36. Parameter estimation using asymmetrical continuously differentiable friction model.....	58
Figure 37. Results of deadzone test showing large initial friction in mechanical subsystem.	60
Figure 38. Comparison of demanded and measured velocity for 1hz sine wave input.	61
Figure 39. Resultant velocity for demanded torque, showing erratic behaviour of system under torque control schemes.	62
Figure 40. Microsoft Life-Cam VX3000 [18].	63
Figure 41. Raspberry Pi Camera Board [20].....	65
Figure 42. Image acquisition latency test. Frame 1 (top left), frame 2 top right), frame 3 (bottom left) and frame 4 (bottom right).	68
Figure 43. Results of simulation at 10Hz, showing signs of the limit of stability.	69
Figure 44. Results of experiment at 10Hz control and measurement frequency.	70
Figure 45. Results of simulation at 9Hz, showing instability.	70
Figure 46. Results of simulation at 20Hz.....	71
Figure 47. Breakdown of time spent.....	75

1 Introduction

The DoD system is an underactuated mechanical system of two disks. The top disk, called the object, is situated on top of the bottom disk, the hand. The top disk is free to roll but not translate out of plane with the bottom disk. The bottom disk is actuated.

The goal of the DoD system is to control the position of the top disk. In this case the desired position is the upright balancing position. There is no limitation currently in place for the position of the bottom disk.

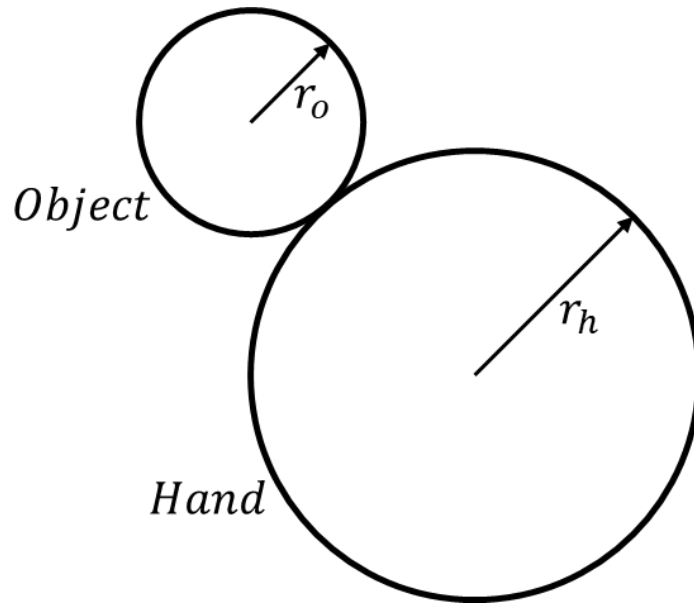


Figure 1. The disk-on-disk system.

Underactuated systems are systems which cannot be commanded to follow an arbitrary trajectory in a configuration space. This situation typically arises from a lesser number of actuators than degrees of freedom, or redundant actuators causing degrees of freedom to not have an associated actuator. The DoD system falls into the first category, with only one actuator attached to the hand and two degrees of freedom.

In the context of robotic manipulators, constraining the system to fully actuated scenarios severely limits their ability to achieve the performance they are capable of. Without exploiting the dynamics of the system during control actions, the full kinematic envelope can never be utilised, which motivates research into developing further understanding of underactuated systems.

Manipulation and control of a system where gripping of the object is possible is relatively straightforward as manipulators can directly influence the position or orientation of the object, and

this type of manipulation is called prehensile manipulation. The familiarity of prehensile manipulation, alongside the simplicity it provides in control the pose of an object, has led to robotic systems and manipulators that rely heavily on prehensile manipulations, such as the robotic gripper in Figure 2.



Figure 2. A robotic gripper [1].

Non-prehensile manipulation is any type of interaction that does not involve gripping, as shown in Figure 3. Non-prehensile manipulations are common for humans, who will happily push, slide, or throw an object to change its position, as they are able to monitor the process visually and adjust movements accordingly.

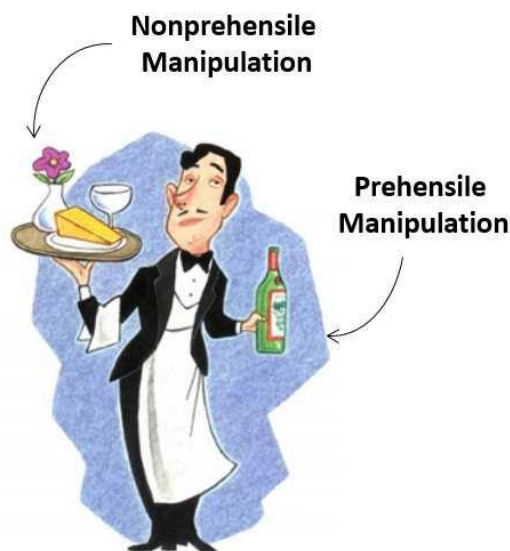


Figure 3. Examples of prehensile and non-prehensile manipulations [2].

Using robotic manipulators to perform these actions is far more difficult than prehensile ones. The manipulation is not a closed kinematic chain and the changes of state are generally non-smooth.

Additionally, as the object is free to move relative to the manipulator, many of these systems become underactuated. Manipulation of the top disk in the DoD system is non-prehensile manipulation.

The DoD system presents a system that requires a specific type of non-prehensile manipulation: rolling. Rolling is generally more complex than a pushing or grasping motion due to the nonlinearity of the motion. The DoD system is further broken down to exclude movement in the axial direction, isolating the rolling movement as the only form. Control of this system presents unique insights into understanding non-prehensile motion in the general sense.

This report investigates one method of modelling, simulating, and controlling the disk-on-disk system shown in Figure 1. Energy based methods of modelling are employed to develop a mathematical model, which is then simulated using Simulink. A contactless method of measuring position is developed using image processing and feature recognition. This process can aid in the understanding of the general non-prehensile control problem.

The DoD system has been studied by several groups, alongside studies of related problems. [3] investigates the control of a DoD system in the same configuration, where control is achieved through passivity-based methods. Control of the object position and velocity is achieved, and the addition of injection damping allows control of the hand position and velocity. The system is modelled using port-Hamiltonian theory, which differs from the Euler-Lagrange methods applied in this report. This paper presents a much more sophisticated approach to both modelling and control. The computer vision used runs at more than double the frequency of this report, allowing higher quality measurements to be taken. Additionally, the motor used is a continuous rotation servomotor, which reduces the complexity of applying the control law by removing the need for control allocation. The DC motors used in this report do not present such finite control over position.

In [4] the same DoD system is modelled and controlled, both in simulation and experiments. The modelling process expresses the model using the Euler-Lagrange equations, but characterises the movements in terms of arc lengths. Control is achieved with state feedback. Again, the measurement frequency of 800Hz far exceeds the control frequency used in this report.

These papers show that control of the DoD system is achievable through a range of other methods than the one presented in this report.

2 Background

The study of classical mechanics, and in turn dynamical systems, is one of the main focus areas of any mechanical or mechatronic engineering degree. Analytical mechanics is focused on the scalar properties of motion which represent the system. As opposed to vector properties, which have both magnitude and direction, scalar properties have only a magnitude. Typically, the scalar properties observed in analytical mechanics is the total kinetic and potential energy.

Analytical mechanics easily describe most mechanical systems encountered. By observing the constraints of a system, the number of degrees of freedom can be limited, reducing the complexity of the problem. Intelligent selection of the generalized coordinates to describe a system can greatly simplify the modelling process further, and often leads to very intuitive descriptions of real-world systems. There are two major branches of analytical mechanics: Lagrangian mechanics and Hamiltonian mechanics.

2.1 Euler-Lagrange Method

The Euler-Lagrange method is an energy-based method of modelling. The Euler-Lagrange method is often used for rigid body dynamic systems as it can be immediately apparent which components of the system are energy storing elements, and what their associated degrees of freedom are. In the case of the DoD system there are two main energy storing elements: the hand and the object. The driveshaft of the hand may also be considered to be energy storing.

Once the energy storing elements of the system are identified, the kinetic co-energy, \mathcal{T}^* , and potential energy, \mathcal{V} , can be determined. The kinetic co-energy and potential energy must be determined for each energy storing element in each associated degree of freedom. By taking the difference of the kinetic co-energy and the potential energy, the Lagrangian of the system is found [5].

$$\mathcal{L} = \mathcal{T}^*(\mathbf{f}, \mathbf{q}) - \mathcal{V}(\mathbf{q}) \quad (1)$$

Where \mathbf{q} is column vector of position coordinates. The kinetic co-energy can be factored into quadratic form, (2), if the component constitutive relationships are linear in velocity¹. The component constitutive relationships describe how a component of the system relates certain magnitudes [6].

¹ Linearity in velocity occurs when the magnitudes are significantly below the speed of light.

$$\mathcal{T}^*(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{M}(\mathbf{q}) \dot{\mathbf{q}} \quad (2)$$

Where $\mathbf{M}(\mathbf{q})$ is the symmetric positive definite mass matrix.

If the effects of damping are to be considered, the impact of the generalised resistors of the system can be accounted for using the Rayleigh dissipation function \mathcal{D} . This function is also factored into quadratic form. The dissipation function must satisfy the inequality $\frac{\partial \mathcal{D}}{\partial \dot{\mathbf{q}}} \dot{\mathbf{q}} \geq 0$ for all $\dot{\mathbf{q}}$.

$$\mathcal{D}(\dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^T \mathbf{D} \dot{\mathbf{q}} \quad (3)$$

Where \mathbf{D} is the symmetric positive semi-definite damping matrix.

If the system in question has non-conservative input forces, these can be included in the Euler-Lagrange equation alongside the dissipation forces.

$$\frac{d}{dt} \left(\frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} \right) - \frac{\partial \mathcal{L}}{\partial \mathbf{q}} = \boldsymbol{\tau} - \frac{\partial \mathcal{D}}{\partial \dot{\mathbf{q}}} \quad (4)$$

By substituting (1), (2) and (3) into (4), the following is found,

$$\frac{d}{dt} (\mathbf{M}(\mathbf{q}) \dot{\mathbf{q}}) - \frac{\partial \mathcal{T}^*}{\partial \mathbf{q}} + \frac{\partial \mathcal{V}}{\partial \mathbf{q}} = \boldsymbol{\tau} - \mathbf{D} \dot{\mathbf{q}} \quad (5)$$

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}} + \dot{\mathbf{M}}(\mathbf{q}) \dot{\mathbf{q}} - \frac{\partial \mathcal{T}^*}{\partial \mathbf{q}} + \frac{\partial \mathcal{V}}{\partial \mathbf{q}} + \mathbf{D} \dot{\mathbf{q}} = \boldsymbol{\tau} \quad (6)$$

$\dot{\mathbf{M}}(\mathbf{q}) \dot{\mathbf{q}} - \frac{\partial \mathcal{T}^*}{\partial \mathbf{q}}$ is grouped to form the centripetal-Coriolis matrix and the gradient of the potential energy function is represented by $\mathbf{g}(\mathbf{q})$, giving

$$\mathbf{M}(\mathbf{q}) \ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \dot{\mathbf{q}} + \mathbf{D} \dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \quad (7)$$

A model of this form for the DoD system is developed in section 3.1.

2.2 Underactuation

A system is said to be underactuated if it cannot be commanded to follow an arbitrary trajectory in the configuration space. There are many causes for underactuation, the most common being a system possessing less actuators than degrees of freedom, or not possessing an actuator associated with every degree of freedom.

The model in the form of (7) is second order and the acceleration states can be described as a function of the positions, \mathbf{q} , and the velocities, $\dot{\mathbf{q}}$. With an input vector \mathbf{u} , the general second order form of a dynamical system which is affine in commanded torque is given by

$$\ddot{\mathbf{q}} = \mathbf{f}_1(\mathbf{q}, \dot{\mathbf{q}}, t) + \mathbf{f}_2(\mathbf{q}, \dot{\mathbf{q}}, t)\mathbf{u} \quad (8)$$

To be fully actuated in the configuration $(\mathbf{q}, \dot{\mathbf{q}}, t)$ the system must satisfy

$$\text{rank}[\mathbf{f}_2(\mathbf{q}, \dot{\mathbf{q}}, t)] = \dim[\mathbf{q}] \quad (9)$$

An underactuated system will instead satisfy

$$\text{rank}[\mathbf{f}_2(\mathbf{q}, \dot{\mathbf{q}}, t)] < \dim[\mathbf{q}] \quad (10)$$

Underactuation is then a property of the configuration $(\mathbf{q}, \dot{\mathbf{q}}, t)$, and not inherently global, although most systems that satisfy (10) in a configuration are in fact globally underactuated [7].

While underactuation inherently causes difficulty in controlling a system as there are trajectories that cannot be directly controlled, underactuated systems are extremely common, with many everyday systems such as cars, boats, and aeroplanes being underactuated. It is these applications, among many other far less complex ones, that drive the need for control.

2.3 Estimation and Control

The desired outcome of control is the ability to achieve desired outcomes in a system, particularly underactuated systems. Controllers monitor the state of the system, and the desired reference behaviour, and determine the difference between the two, known as the error. This error is used as feedback to determine the next control action. Control methods vary from primitive, such as proportional-integral-derivative controllers (PID), to complex methods, such as model predictive control (MPC). PID control uses only the state error to determine the next control action, whereas more sophisticated methods employ state estimation techniques. Two common methods of state-feedback control are the Linear-Quadratic Regulator (LQR) and Model Predictive Control (MPC).

2.3.1 Linear-Quadratic Regulator

LQR requires that the dynamics of the system be wholly described by a set of linear differential equations. The core of the LQR method is minimising a quadratic cost function.

Consider the system

$$\mathbf{x}_{k+1} = \mathbf{A}_d \mathbf{x}_k + \mathbf{B}_d \mathbf{u}_k \quad (11)$$

$$\mathbf{y}_k = \mathbf{C} \mathbf{x}_k + \mathbf{D} \mathbf{u}_k \quad (12)$$

which is linear time-invariant and stabilisable. The desired feedback control law is of the form

$$\mathbf{u}_k = -\mathbf{K}\mathbf{x}_k \quad (13)$$

which minimises the cost function

$$\mathcal{V} = \sum_{k=0}^{\infty} \mathbf{x}_k^T \mathbf{Q}_d \mathbf{x}_k + \mathbf{u}_k^T \mathbf{R}_d \mathbf{u}_k \text{ where } \mathbf{Q}_d \geq 0, \mathbf{R}_d > 0 \quad (14)$$

This cost function is the sum of the quadratic function in the states and the quadratic function in the inputs. The matrix \mathbf{Q} influences the control response to the state deviations from zero. If the magnitude of \mathbf{Q} is large, the control force will increase to minimise the deviation of the state from zero. Similarly, the matrix \mathbf{R} influences the control effort. If the magnitude of \mathbf{R} is large, the control effort will be minimised.

The solution to (14) is given when

$$\mathbf{K} = (\mathbf{R}_d + \mathbf{B}_d^T \mathbf{S} \mathbf{B}_d)^{-1} \mathbf{B}_d^T \mathbf{S} \mathbf{A}_d \quad (15)$$

\mathbf{S} is the solution to the Discrete-time Algebraic Ricatti Equation

$$\mathbf{0} = \mathbf{A}_d^T (\mathbf{S} - \mathbf{S} \mathbf{B}_d (\mathbf{R}_d + \mathbf{B}_d^T \mathbf{S} \mathbf{B}_d)^{-1} \mathbf{B}_d^T \mathbf{S}) \mathbf{A}_d + \mathbf{Q}_d - \mathbf{S} \quad (16)$$

The closed-loop system is stable if \mathbf{A}_d and \mathbf{B}_d are stabilisable, and \mathbf{A}_d and $\mathbf{Q}_d^{1/2}$ are detectable [8].

Fortunately, MATLAB has an inbuilt function to calculate the discrete-time LQ control gain \mathbf{K} given the continuous-time state space equations. The control system toolbox functions ‘lqrd’ and ‘c2d’ use the method outline above and the zero-order hold approximation, respectively.

LQR can be used with a reference feedforward structure, allowing a state to be regulated to a desired setpoint, \mathbf{r} . This involves the introduction of a non-zero steady state and input into the control law derived above.

$$\mathbf{u}_k - \mathbf{u}_{ss} = -\mathbf{K}(\mathbf{x}_k - \mathbf{x}_{ss}) \quad (17)$$

As $k \rightarrow \infty$, then $\mathbf{x}_k \rightarrow \mathbf{x}_{ss}$ and $\mathbf{u}_k \rightarrow \mathbf{u}_{ss}$, the desired output is \mathbf{r} . Therefore, the following must be satisfied

$$\mathbf{x}_{ss} = \mathbf{A}_d \mathbf{x}_{ss} + \mathbf{B}_d \mathbf{u}_{ss} \quad (18)$$

$$\mathbf{r} = \mathbf{C}\mathbf{x}_{ss} + \mathbf{D}\mathbf{u}_{ss} \quad (19)$$

which can be rearranged into a matrix equation

$$\begin{bmatrix} \mathbf{A}_d - \mathbf{I} & \mathbf{B}_d \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{ss} \\ \mathbf{u}_{ss} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \mathbf{r} \quad (20)$$

The solution is linear in \mathbf{r} , allowing \mathbf{x}_{ss} to be set to $\mathbf{N}_x \mathbf{r}$ and \mathbf{u}_{ss} to be set to $\mathbf{N}_u \mathbf{r}$. The equation can then be solved to find

$$\begin{bmatrix} \mathbf{N}_x \\ \mathbf{N}_u \end{bmatrix} = \begin{bmatrix} \mathbf{A}_d - \mathbf{I} & \mathbf{B}_d \\ \mathbf{C} & \mathbf{D} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{I} \end{bmatrix} \quad (21)$$

This gives the control law \mathbf{u}_k

$$\mathbf{u}_k = (\mathbf{N}_u + \mathbf{K}\mathbf{N}_x)\mathbf{r} - \mathbf{K}\mathbf{x}_k \quad (22)$$

The entire control law, consisting of the reference feedforward terms and the state feedback terms, can be implemented with the structure shown in Figure 4 [8].

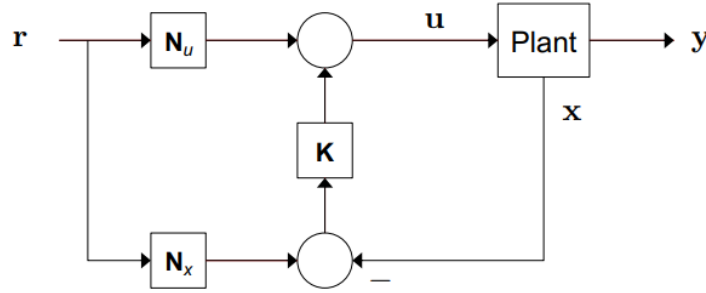


Figure 4. Linear-Quadratic Regulator with reference feedforward control structure [8].

2.3.2 Model Predictive Control

Model predictive control calculates the optimal control law for the each timestep in a nominated control horizon by minimizing a cost function with the knowledge of current state estimates and a desired outcome.

Again, consider the system described in (11), and the cost function

$$\mathcal{V} = \frac{1}{2}(\mathbf{x}_{N+1} - \mathbf{x}_{N+1}^*)^T \mathbf{Q}_f (\mathbf{x}_{N+1} - \mathbf{x}_{N+1}^*) + \sum_{k=1}^N \frac{1}{2}(\mathbf{x}_k - \mathbf{x}_k^*)^T \mathbf{Q} (\mathbf{x}_k - \mathbf{x}_k^*) + \frac{1}{2} \mathbf{u}_k^T \mathbf{R} \mathbf{u}_k \quad (23)$$

we seek to minimise the cost function to determine the control law:

$$\mathbf{U}^* = \arg \min \mathcal{V}(\mathbf{x}_1, \mathbf{U}^N) \quad (24)$$

$$\mathbf{U}_N^* = \{\mathbf{u}_1^*, \mathbf{u}_2^*, \dots, \mathbf{u}_N^*\} \quad (25)$$

A more desirable cost function is

$$\mathcal{V} = \sum_{k=1}^{\infty} \frac{1}{2} (\mathbf{y}_k - \mathbf{y}_k^*)^T \mathbf{Q}_y (\mathbf{y}_k - \mathbf{y}_k^*) + \sum_{k=1}^{\infty} \frac{1}{2} (\mathbf{u}_k - \mathbf{u}_{k-1}^*)^T \mathbf{R}_u (\mathbf{u}_k - \mathbf{u}_{k-1}^*) \quad (26)$$

Where \mathbf{Q}_y and \mathbf{R}_u represent the output error and actuator change penalty, respectively. Increasing the penalty on the output error drives the controller to force the system to respond more aggressively to deviations in state from the desired reference; increasing the penalty on actuator change drives the controller to limit the change in control law between timesteps. The inverse of these rules also applies. This cost function allows tuning of the response of the system via two methods, depending on the desired behaviour.

MPC also allows constraints to be included in the minimisation problem. This is particularly convenient for physical systems, as motor slew rate and velocity constraints are directly included in the controller. The importance of constraints included in the control calculation cannot be understated, in particular the inclusion of hard constraints. LQR has no such ability to include rigid actuator constraints, allowing MPC to be expanded into control of a nonlinear system beyond the operating point, which must be linearized for LQG control.

Slew rate and velocity saturation input constraints can be expressed as follows.

$$\mathbf{u}_{\min} \leq \mathbf{u}_k \leq \mathbf{u}_{\max}, \quad k = 1, \dots, N \quad (27)$$

$$\Delta \mathbf{u}_{\min} \leq \mathbf{u}_k - \mathbf{u}_{k-1} \leq \Delta \mathbf{u}_{\max}, \quad k = 1, \dots, N \quad (28)$$

Alternatively expressed as $\mathcal{L}\mathbf{u} \leq \mathbf{m}$ for use in MPC control design

$$\begin{aligned} \mathcal{L} &= \begin{bmatrix} \mathbf{I} \\ -\mathbf{I} \\ \mathcal{W} \\ -\mathcal{W} \end{bmatrix}, \quad \mathbf{m} = \begin{bmatrix} \mathbf{u}_{\max} \\ -\mathbf{u}_{\min} \\ \Delta \mathbf{u}_{\max} \\ -\Delta \mathbf{u}_{\min} \end{bmatrix}, \quad \mathbf{u} \triangleq \begin{bmatrix} \mathbf{u}_1 \\ \mathbf{u}_2 \\ \vdots \\ \mathbf{u}_N \end{bmatrix}, \quad \mathbf{u}_{\min} \triangleq \begin{bmatrix} \mathbf{u}_{\min} \\ \mathbf{u}_{\min} \\ \vdots \\ \mathbf{u}_{\min} \end{bmatrix}, \quad \mathbf{u}_{\max} \triangleq \begin{bmatrix} \mathbf{u}_{\max} \\ \mathbf{u}_{\max} \\ \vdots \\ \mathbf{u}_{\max} \end{bmatrix} \\ \mathcal{W} &\triangleq \begin{bmatrix} \mathbf{I} & \mathbf{0} & \dots & \dots & \mathbf{0} \\ -\mathbf{I} & \mathbf{I} & \ddots & & \vdots \\ \mathbf{0} & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \dots & \mathbf{0} \\ \mathbf{0} & \dots & \mathbf{0} & -\mathbf{I} & \mathbf{I} \end{bmatrix}, \quad \Delta \mathbf{u}_{\min} \triangleq \begin{bmatrix} \mathbf{u}_0 + \Delta \mathbf{u}_{\min} \\ \Delta \mathbf{u}_{\min} \\ \vdots \\ \Delta \mathbf{u}_{\min} \end{bmatrix}, \quad \Delta \mathbf{u}_{\max} \triangleq \begin{bmatrix} \mathbf{u}_0 + \Delta \mathbf{u}_{\max} \\ \Delta \mathbf{u}_{\max} \\ \vdots \\ \Delta \mathbf{u}_{\max} \end{bmatrix} \end{aligned} \quad (29)$$

No matter the control method chosen, accurate measurements of state are required.

2.4 Sensing Technologies

Sensing technologies in the context of mechatronics refers to transducers which convert a physical property into an electrical signal. The conversion may be straightforward as in a potentiometer, which converts a physical angle of rotation between two points into a voltage between two points. A camera or image sensor does not have such a straightforward path, but is a sensor nonetheless. In the case of mechanical systems, measurements of position are often desired. For motors, acquiring a position measurement of the shaft is often achieved using encoders, which also allows for an estimation of velocities. Unfortunately, it is not always possible to physically interact with the object to be measured, driving the need for contactless measurements systems such as LIDAR and computer vision-based systems, which present significant challenges for real-time control.

2.5 The Kalman Filter

A Kalman Filter can be used for state estimation. The Kalman Filter is the closed form solution to the Bayes Filter when it is assumed the state-transition model and measurement model are multivariate normally distributed. A Bayes Filter relies on two assumptions: the state transition model is a Markov Chain, and the measurement model is conditionally independent. These assumptions ensure that the probability density function (PDF) of the next state is only dependant on the previous state, and the measurement at a given time is only dependent on the state at that time. The Bayes Filter has the general form of Algorithm 1 [9].

Algorithm 1 – Bayes Filter

Choose initial state PDF (prior)

$$p(x_0|y_0) = p(x_0) \quad (30)$$

for $t = 1, 2 \dots T$

 Measurement update: update the prior with new information

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{t-1})dx_t} \quad (31)$$

 Prediction update: compute the state distribution at $t+1$

$$p(x_{t+1}|y_{1:t}) = \int p(x_{t+1}|x_t)p(x_t|y_{1:t})dx_t \quad (32)$$

end

The Bayes Filter has two implementation issues: the measurement update and prediction update both contain the product of PDF's and an integration [9]. The Kalman Filter does not have these issues due to the assumptions on the state and measurement PDF's. For a system with the form

$$\mathbf{x}_{t+1} = \mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{u}_t + \mathbf{w}_t \quad (33)$$

$$\mathbf{y}_t = \mathbf{C}_t \mathbf{x}_t + \mathbf{D}_t \mathbf{u}_t + \mathbf{v}_t \quad (34)$$

where $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_t)$ and $\mathbf{v}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_t)$, then the Bayesian Filter has the closed form solution

$$p(\mathbf{x}_{t+1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{A}_t \mathbf{x}_t + \mathbf{B}_t \mathbf{u}_t, \mathbf{Q}_t) \quad (35)$$

$$p(\mathbf{y}_t|\mathbf{x}_t) = \mathcal{N}(\mathbf{C}_t \mathbf{x}_t + \mathbf{D}_t \mathbf{u}_t, \mathbf{R}_t) \quad (36)$$

$$p(\mathbf{x}_1) = \mathcal{N}(\boldsymbol{\mu}_{1|0}, \mathbf{P}_{1|0}) \quad (37)$$

This reduces the measurement and prediction step of the Bayes Filter to produce multivariate normal distributions, giving Algorithm 2 [9].

Algorithm 2 – Kalman Filter

Choose initial state PDF (prior)

$$p(x_1) = \mathcal{N}(\mu_{1|0}, P_{1|0}) \quad (38)$$

for $t = 1, 2, \dots, T$

Measurement update: update the prior with new information

$$p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_t} = \mathcal{N}(\mu_{t|t}, P_{t|t}) \quad (39)$$

Prediction update: compute the state distribution at $t + 1$

$$p(x_{t+1}|y_{1:t}) = \int p(x_{t+1}|x_t)p(x_t|y_{1:t})dx_t = \mathcal{N}(\mu_{t+1|t}, P_{t+1|t}) \quad (40)$$

end

where

$$\mu_{t|t} = \mu_{t|t-1} + P_{t|t}C_t^T(C_tP_{t|t-1}C_t^T + R_t)^{-1}(y_t - C_t\mu_{t|t-1} - D_tu_t) \quad (41)$$

$$P_{t|t} = P_{t|t-1} - P_{t|t-1}C_t^T(C_tP_{t|t-1}C_t^T + R_t)^{-1}C_tP_{t|t-1} \quad (42)$$

$$\mu_{t+1|t} = A_t\mu_{t|t} + B_tu_t \quad (43)$$

$$P_{t+1|t} = A_tP_{t|t}A_t^T + Q_t \quad (44)$$

If the uncertainty on the states is assumed to be normally distributed, then the state space model derived in (11) and (12) is of the form required for a Kalman Filter.

Use of a Kalman Filter for state estimation ensures that the state estimates are suitable for control law calculations. The Kalman filter ensures that erroneous measurements that are extremely unlikely given the state transition model are correctly treated as noise.

There are effectively three tuning parameters associated with a Kalman Filter: the state transition model uncertainty, the measurement uncertainty, and the initial prediction density. These parameters are expressed as covariance matrices. The state transition model uncertainty captures the confidence in the model. A model that does not accurately reflect the true system dynamics should have a large model uncertainty.

3 Modelling and Simulation

Simulation and modelling are the foundation upon which a mechatronic system is built. A model of a system is a mathematical description of the output behaviour given an input. By taking this model and applying known inputs, the behaviour of the system can be recorded, and a simulation has occurred. If a particular output or system behaviour is desired, then the system must be influenced in such a way that the desired behaviour is realised. Influencing a system in this way is called control. Control without a working model and simulation is impossible.

The modelling of a system is often the first step taken in this process. The physical arrangement of the DoD system is already defined, and the physical phenomena at work are well known. The kinematics of the DoD system is adequately captured through use of the Euler-Lagrange modelling method.

3.1 The Disk-on-Disk System

A simplified model of the DoD system is considered for the modelling process. This simplification process is based upon several assumptions:

- (A1) Torque transfer between the actuator and the hand is not subject to any dissipation effects
- (A2) The object and hand interact without slipping
- (A3) The hand and object are always in contact
- (A4) The system is always operating near the upright balancing point

The model developed will be unable to perfectly describe the system dynamics if one or more of these assumptions is not met. Figure 5 shows the layout of the DoD system.

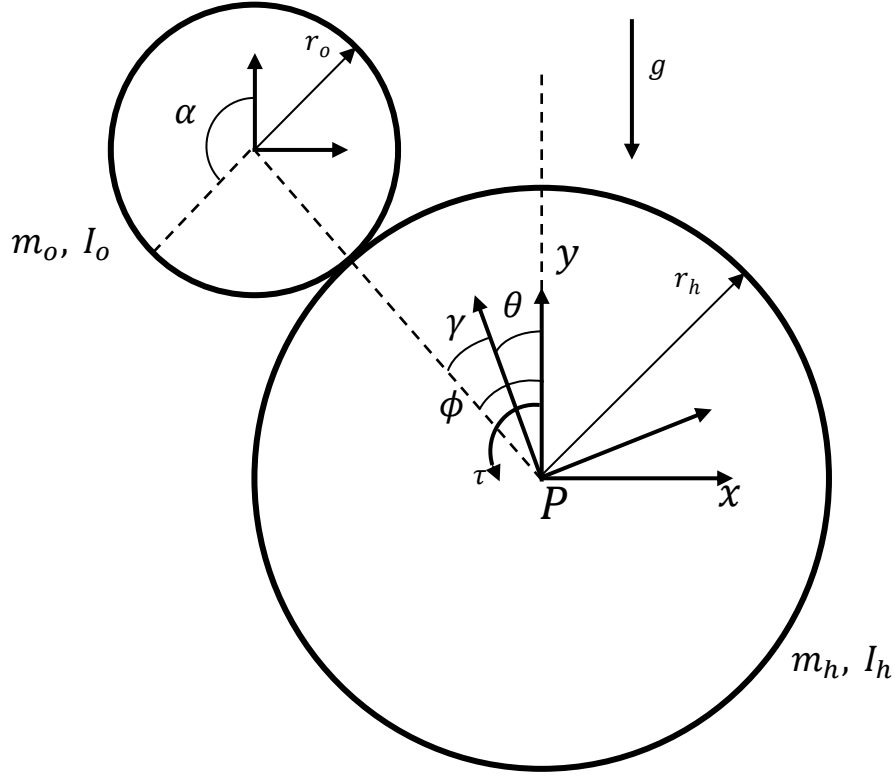


Figure 5. Layout of ideal DoD system.

The energy storing elements and their associated degrees of freedom can be identified via inspection. The hand has a single degree of freedom: the angle of rotation θ about point P. The object has two associated degrees of freedom: the angle of rotation α about its centre and the angle of rotation ϕ about the point P. Assumption (A2) allows α to be described as a function of θ and ϕ , reducing the system to two degrees of freedom.

There are two scenarios to consider when developing the kinematic model: the case where $\theta = 0$ and the case where $\phi = 0$.

In the first case, if $\theta = 0$ then

$$\alpha = \frac{r_h}{r_o} \gamma \text{ where } \phi = \gamma - \theta \quad (45)$$

$$\therefore \alpha = \frac{r_h}{r_o} \phi \quad (46)$$

Similarly, when $\phi = 0$

$$\alpha = -\frac{r_h}{r_o} \theta \quad (47)$$

This then gives the relationship:

$$\alpha = \frac{r_h}{r_o}(\theta - \phi) \text{ and } \dot{\alpha} = \frac{r_h}{r_o}(\dot{\theta} - \dot{\phi}) \quad (48)$$

The kinetic co-energy of the object can then be described as

$$\mathcal{T}_o^* = \frac{1}{2}m_o\dot{x}^2 + \frac{1}{2}m_o\dot{y}^2 + \frac{1}{2}J_o\dot{\alpha}^2 \quad (49)$$

$$\mathcal{T}_o^*(\dot{\theta}, \dot{\phi}) = \frac{1}{2}m_o\dot{\phi}^2(r_h + r_o)^2 + \frac{1}{2}J_o\left(\frac{r_h^2}{r_o^2}\dot{\phi}^2 - 2\frac{r_h^2}{r_o^2}\dot{\phi}\dot{\theta} + \frac{r_h^2}{r_o^2}\dot{\theta}^2\right) \quad (50)$$

The conversion between translational and rotational coordinates is most easily accomplished using the single-axis rotation matrices. Alternatively, the parallel axis theorem may be used to find the rotational inertia of the object about the point P and solve directly.

The kinetic co-energy of the hand is

$$\mathcal{T}_h^*(\dot{\theta}) = \frac{1}{2}J_h\dot{\theta}^2 \quad (51)$$

The total kinetic co-energy can then be factorised to find the mass matrix with the generalised coordinates $\mathbf{q} = \begin{bmatrix} \theta \\ \phi \end{bmatrix}$.

$$\mathcal{T}^*(\dot{\theta}, \dot{\phi}) = \frac{1}{2} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix} \begin{bmatrix} J_h + J_o \frac{r_h^2}{r_o^2} & -J_o \frac{r_h^2}{r_o^2} \\ -J_o \frac{r_h^2}{r_o^2} & m_o(r_h + r_o)^2 + J_o \frac{r_h^2}{r_o^2} \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix} \quad (52)$$

$$\therefore \mathbf{M} = \begin{bmatrix} J_h + J_o \frac{r_h^2}{r_o^2} & -J_o \frac{r_h^2}{r_o^2} \\ -J_o \frac{r_h^2}{r_o^2} & m_o(r_h + r_o)^2 + J_o \frac{r_h^2}{r_o^2} \end{bmatrix} \quad (53)$$

The potential energy is given by

$$\mathcal{V}(\phi) = m_o g(r_h + r_o) \cos(\phi) \quad (54)$$

$$\therefore \mathbf{g}(\phi) = \begin{bmatrix} 0 \\ -m_o g(r_h + r_o) \sin(\phi) \end{bmatrix} \quad (55)$$

There is only one input force, the torque τ applied to the hand, giving

$$\boldsymbol{\tau} = \begin{bmatrix} \tau \\ 0 \end{bmatrix} \quad (56)$$

As the mass matrix is not time varying and the kinetic co-energy is not a function of \mathbf{q} , the centripetal-Coriolis matrix is zero. This leaves the final Euler-Lagrange equation as

$$\begin{bmatrix} J_h + J_o \frac{r_h^2}{r_o^2} & -J_o \frac{r_h^2}{r_o^2} \\ -J_o \frac{r_h^2}{r_o^2} & m_o(r_h + r_o)^2 + J_o \frac{r_h^2}{r_o^2} \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ -m_o g(r_h + r_o) \sin(\phi) \end{bmatrix} = \begin{bmatrix} \tau \\ 0 \end{bmatrix} \quad (57)$$

3.2 Simulation

Simulink is a graphical block diagram simulation program. It enables the user to build a model using distinct block components in order to simulate the behaviour of a dynamic system. Simulink is often useful when a system is a combination of several distinct subsystems as it allows the user to add or remove subsystems easily, with interactions being visually displayed.

The DoD system can be built and simulated in Simulink using integration blocks. By rearranging the derived Euler-Lagrange equation so that the second state derivative is the subject of the equation, the system can be set up with sequential integration blocks, allowing the position states and the velocity derivatives to be calculated.

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\boldsymbol{\tau} - \mathbf{g}(\mathbf{q})) \quad (58)$$

These states and derivatives are then used to calculate the next state. The layout of this model is shown in Figure 6 below.

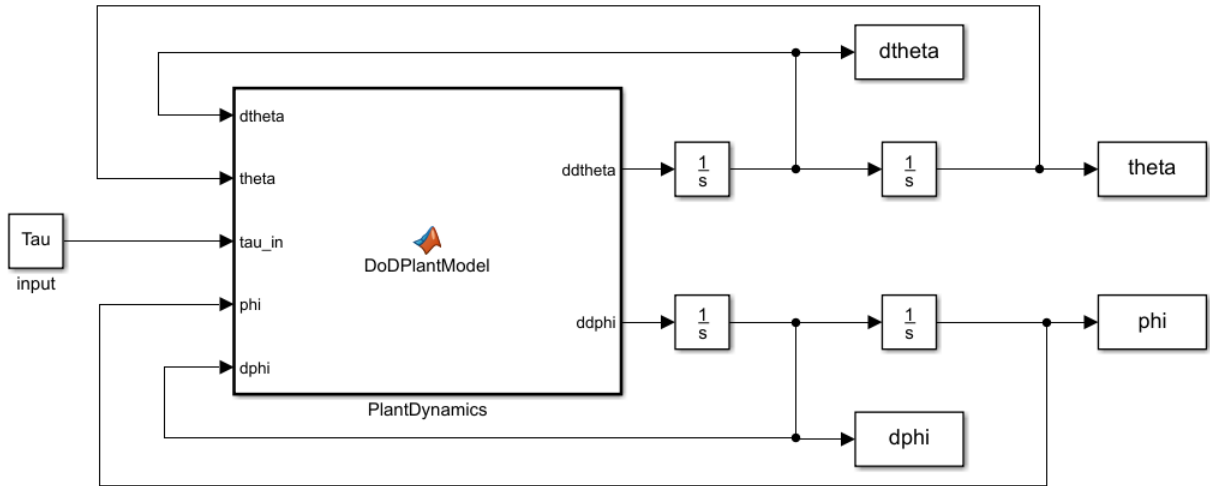


Figure 6. Layout of Simulink model.

The derived model can now be verified using this simulation. If the object is initially positioned at small angle offset from the upright balancing position, the expectation is that it would fall toward the ground. Due to (A3) and the absence of energy dissipation, the object is expected to swing with sinusoidal motion without decay. The hand is also expected to rotate due to (A2). Figure 7 below

shows the results of running the simulation for these conditions, verifying that the model is a valid representation of the DoD system.

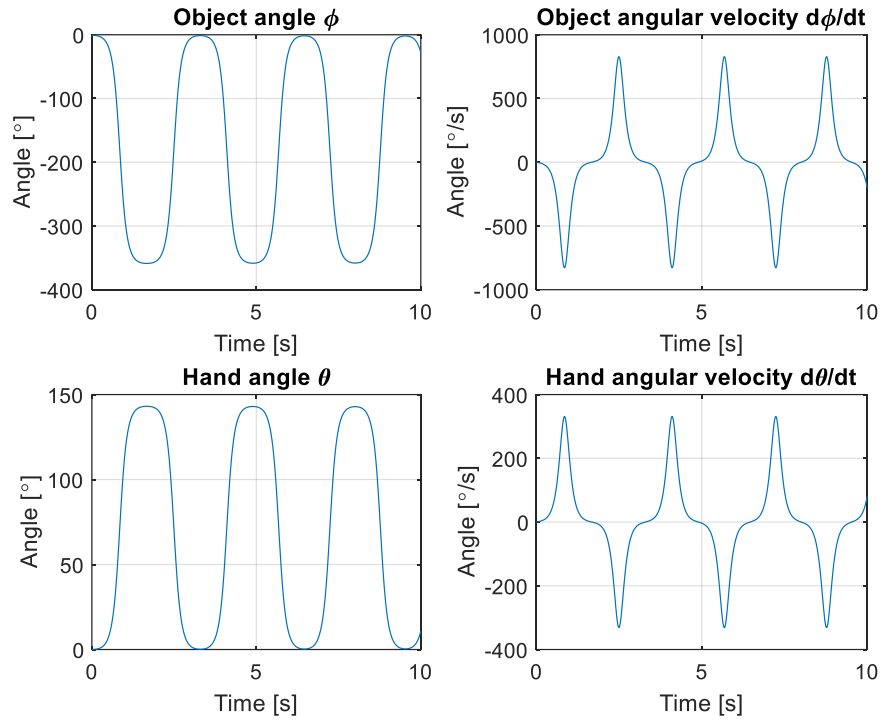


Figure 7. Plots of hand and object angle and angular velocity.

4 Estimation and Control

The ability to control the output behaviour of a system is often desired. When a system is controlled, the output of the system follows some specified reference input, regardless of the details of the dynamics in the system. Robust control methods allow reference input tracking in the presence of input disturbances, errors introduced to the model by assumptions, and errors introduced in measurements by the sensors [10].

Digital control methods are, by nature, discrete-time problems. Despite continual advancements in technology, calculations must ultimately be performed at specified time intervals. The models so far developed have been in continuous-time and must be discretised in order to develop the discrete-time control algorithms.

4.1 Controller Design

Typically, when using DC motors, the model of the system is developed such that the input to the system and output of the controller is torque. This is due to the nature of the DC motor model itself and the relationship between current and torque, and the simplicity of modelling using the Euler Lagrange method. An alternative is to develop a system model with a velocity input and develop a corresponding controller with velocity or acceleration output. Control models with velocity inputs are far more common for flow source actuators, such as stepper motors, but can be used with effort source actuators in certain circumstances.

Torque based control using a DC motor requires the motor to be characterised with a system identification process, as seen in section 6.1, and has a far more natural derivation of the control and observer models from the Euler Lagrange equations. Velocity based control requires no system identification but requires that control allocation can be implemented at significantly higher frequencies than the control frequency. Velocity based control and observer models are unable to be directly derived from the Euler Lagrange equation due to a causality issue.

Regardless of the input to the system and output of the controller, the plant model must be expressed in the state space form. For both velocity and torque control, the state to be regulated is the velocity of the hand. The torque controller achieves this by calculating the required torque input to achieve the desired velocity, whilst the velocity controller simply controls the velocity.

4.1.1 Torque Control

A continuous-time state space representation of the system described in Lagrangian model in (57) is found via the following process. Upon linearisation about the upright balancing position, (57) is of the form

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \boldsymbol{\tau} \quad (59)$$

where \mathbf{M} , \mathbf{D} and \mathbf{K} are the mass, damping and stiffness matrices, respectively. The generalised positions are chosen for the state vector $\mathbf{q} = \begin{bmatrix} \theta \\ \phi \end{bmatrix}$. By choosing $\mathbf{v} \triangleq \dot{\mathbf{q}}$, (59) can be rearranged into the standard state space form

$$\begin{bmatrix} \dot{\mathbf{v}} \\ \dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} -\mathbf{M}^{-1}\mathbf{D} & -\mathbf{M}^{-1}\mathbf{K} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{q} \end{bmatrix} + \begin{bmatrix} \mathbf{M}^{-1}\mathbf{e} \\ \mathbf{0} \end{bmatrix} \boldsymbol{\tau} \text{ where } \mathbf{e} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad (60)$$

As the goal is to achieve setpoint regulation for a specified velocity, the position of the bottom disk, θ , is potentially unbounded and is therefore removed from the controller state vector.

There is only one output to be regulated, the velocity of the hand $\dot{\theta}$, making the output regulation equations trivial.

$$\mathbf{y}_r = \dot{\theta} = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{v} \\ \mathbf{q} \end{bmatrix} \quad (61)$$

This control model, in state space form, can be used to derive the observer model by the exclusion of the object angular velocity state, which is not observable.

4.1.2 Velocity Control

Derivation of a control model for velocity control is not possible by rearranging the Euler Lagrange equations. The Lagrangian model developed in section 3.1 uses hand velocity as a generalized coordinate, which in a velocity control system, would also be the plant input. This causes a causality issue in the model which must be corrected. Fortunately, Hamiltonian models are closely linked to Lagrangian models and offer a suitable change of coordinates. The Hamiltonian is given by

$$\mathcal{H}(\mathbf{p}, \mathbf{q}) = \frac{1}{2} \mathbf{p}^T \mathbf{M}^{-1}(\mathbf{q}) \mathbf{p} + V(\mathbf{q}) \quad (62)$$

where

$$\mathbf{p} = \frac{\partial \mathcal{L}}{\partial \dot{\mathbf{q}}} = \mathbf{M}(\mathbf{q}) \dot{\mathbf{q}} \quad (63)$$

Choosing the states p_ϕ and ϕ results in the Hamiltonian plant model

$$\begin{bmatrix} \dot{p}_\phi \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} m_o g(r_h + r_o) \sin(\phi) \\ p_\phi + J_o \frac{r_h^2}{r_o^2} \dot{\theta} \\ m_o(r_h + r_o)^2 + J_o \frac{r_h^2}{r_o^2} \end{bmatrix} \quad (64)$$

The behaviour of the Hamiltonian model and the Lagrangian model are compared under a zero-input scenario, with similar behaviour observed. Slight discrepancies are expected between the models as the hand is fixed due to a zero velocity input in the Hamiltonian model, whilst the hand is able to move in the Lagrangian model due to a zero torque input.

To design a control model with a velocity output the Lagrangian model is linearised assuming the plant has an acceleration input, and the output of the controller is integrated prior to the plant. This method is favourable as it does not require derivation of the Hamiltonian of the system. Linearisation of the Lagrangian about the upright balancing position results in the control model

$$\begin{bmatrix} \ddot{\theta} \\ \ddot{\phi} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & \frac{m_o g(r_h + r_o)}{H} \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \\ \phi \end{bmatrix} + \begin{bmatrix} 1 \\ J_o(r_h^2 + r_o^2) \\ H \\ 0 \end{bmatrix} \ddot{\theta} \quad (65)$$

where $H = m_o(r_h + r_o)^2 + J_o \frac{r_h^2}{r_o^2}$

The observer model is built off the controller model with the addition of hand position θ as a state, which is observable but not controllable,

$$\begin{bmatrix} \ddot{\theta} \\ \ddot{\phi} \\ \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{m_o g(r_h + r_o)}{H} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \\ \theta \\ \phi \end{bmatrix} + \begin{bmatrix} 1 \\ J_o(r_h^2 + r_o^2) \\ H \\ 0 \end{bmatrix} \ddot{\theta} \quad (66)$$

With the C matrix

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (67)$$

These models are both in the state space form and suitable for use with LQG or MPC and the Kalman Filter.

4.1.3 Comparison of Torque Control and Velocity Control

There is little difference between the torque control and velocity control methods in simulation, as to be expected. The controllers developed are both perfectly adequate of controlling the system when formulated as a state space model, with Figure 8 and Figure 9 showing the result of balancing from

an initial non-zero position and regulating velocity to some set reference using LQG. It is determined that either method of control is suitable for implementation, with torque control chosen as the initial preference.

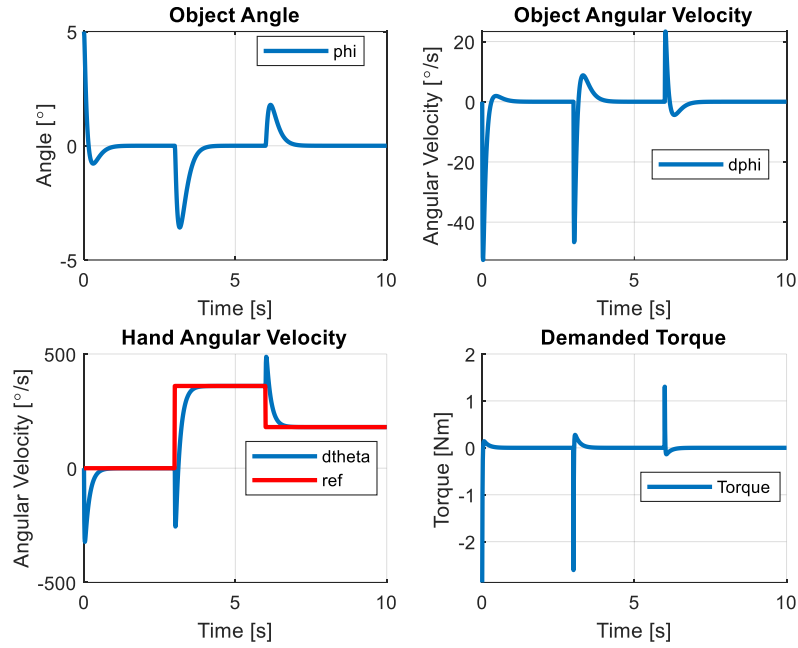


Figure 8. Torque control in simulation.

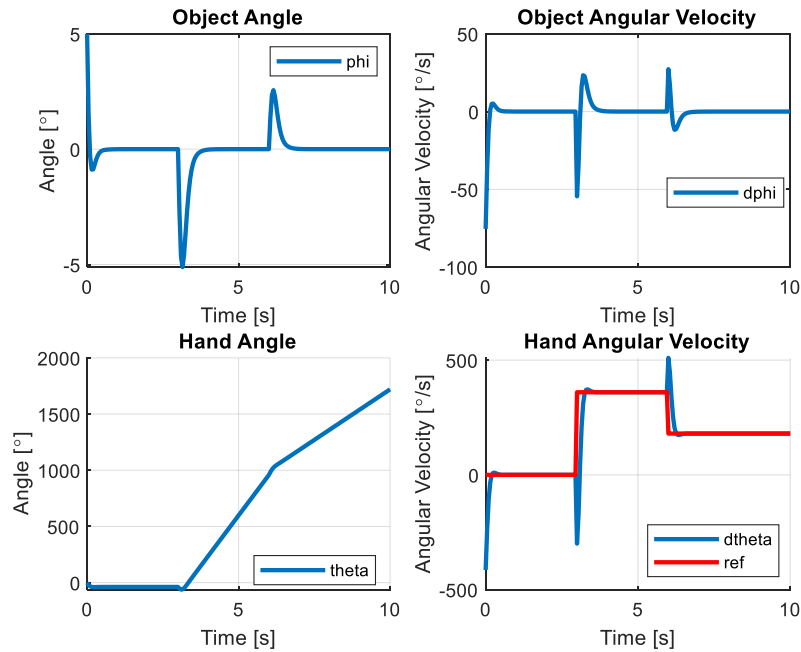


Figure 9. Velocity control in simulation.

4.1.4 Comparison of LQG and MPC

A MPC controller is also designed and implemented in simulation, with a comparison between the LQG and MPC controllers shown in Figure 10. MPC is a more sophisticated control method, with actuator limits able to be included in the controller itself. The LQG controller is chosen over the MPC for several reasons. The simplicity enhances debugging capabilities in implementation and reduces computation time required to calculate each control action. Computation time at each timestep is a significant consideration, as addressed in later sections. Additionally, the performance between the two controllers is similar, as shown in Figure 10.

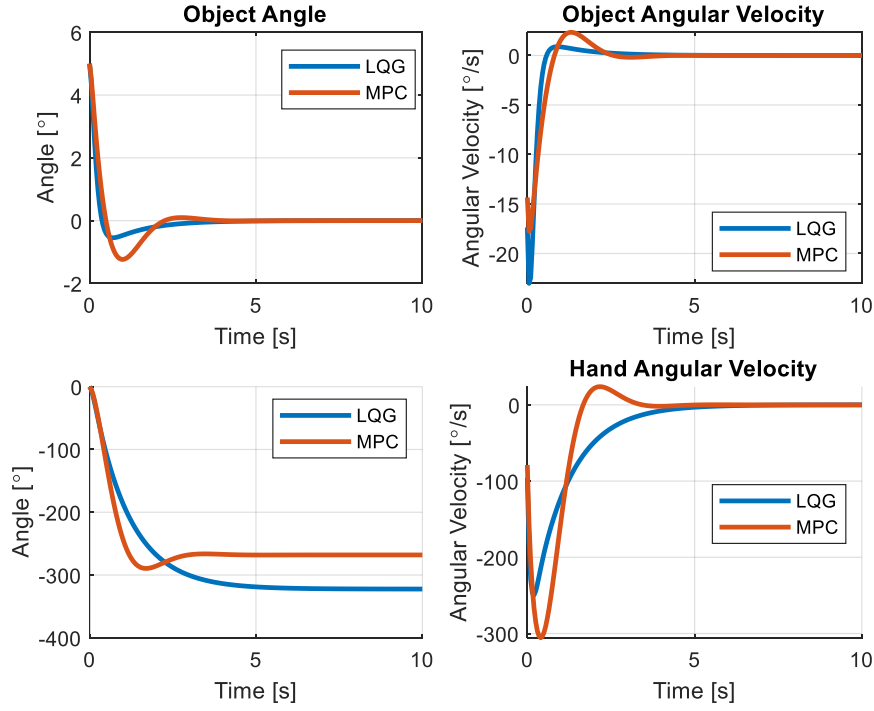


Figure 10. Comparison of LQG and MPC controllers.

4.2 State estimation

As the name implies, state feedback control requires knowledge of the states. Measurement of the states is often unavailable or subject to a level of noise. In the case of the DoD system, angular position measurements of the hand can be directly taken via an encoder on the motor drive shaft. An approximation of the hand velocity can be calculated using the difference in the position measurements between timesteps.

Critically, the position and velocity of the object is not easily measured. It is possible to attach an inertial measurement unit to the object, which would allow calculation of the angle and angular velocity. These measurements would need to be communicated, which presents significant problems.

Wired methods of communication would present significant risk of entanglement with rotating components, whilst wireless methods would involve embedding power and communication electronics inside of the object. A more elegant solution is the use of contactless measurement.

4.2.1 Computer Vision

Computer vision is the process of extracting information from still images or video, particularly the identification of features or objects. Using image processing techniques, a stream of images captured by a low-end webcam can be used to make relatively accurate measurements of the position of the upper disk in the DoD system. The image capturing device often introduces distortion to the image which requires a correction process to negate the error in measurements.

The errors in measurements are caused by the camera lens distorting the light as it passes through, and the projection of the three-dimensional environment onto a two-dimensional plane. The nature of the DoD system reduces the impact of these errors. As the location of the camera relative to the plane containing the front of the two disks is constant, the distortion due to conversion between three-dimensional and two-dimensional space does not need to be recalculated each time a measurement is taken.

Calibration of the camera is a straightforward process with the aid of the MATLAB computer vision toolbox. A ‘checkerboard’ pattern shown in Figure 11 is photographed in several different positions in the three-dimensional space with the camera fixed in place.

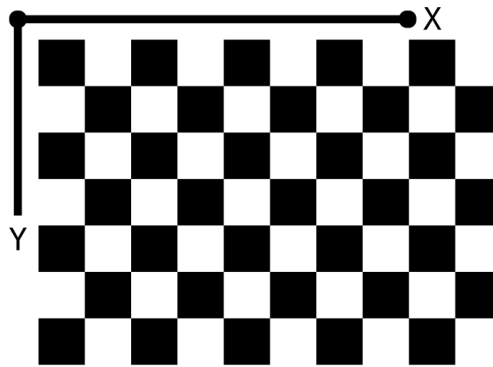


Figure 11. Checkerboard pattern used for camera calibration.

This pattern is used as it provides many well-defined cross-corners² that are easily identified by the corner detection algorithm. The detected corners in image coordinates are then used, in conjunction

² A cross-corner is the intersection of two white and two black squares.

with the measured edge length of the squares, to solve for the intrinsic and extrinsic camera parameters in closed form using the method outlined in [11]. Initially the lens distortion is assumed to be zero. This closed form solution is then used to estimate all of the camera parameters simultaneously using the nonlinear least-squares minimisation method outlined in [12]. The closed form solution is used as the initial estimate for the camera parameters and the initial estimate of the distortion is zero [13].

Using these estimated camera parameters, an input image can be undistorted. A summary of the reprojection errors is also calculated during this process. In the absence of experiment apparatus to test the camera calibration process, a series of seven test images are used. Figure 12 (left) shows an original distorted image captured with the camera. The image includes both the checkerboard pattern and two regions of interest. Figure 12 (right) shows the undistorted image after the camera calibration process. The comparison shows the effects of radial lens distortion on the original image. If this distortion is not accounted for it is impossible to obtain accurate measurements of the locations of points of interest.

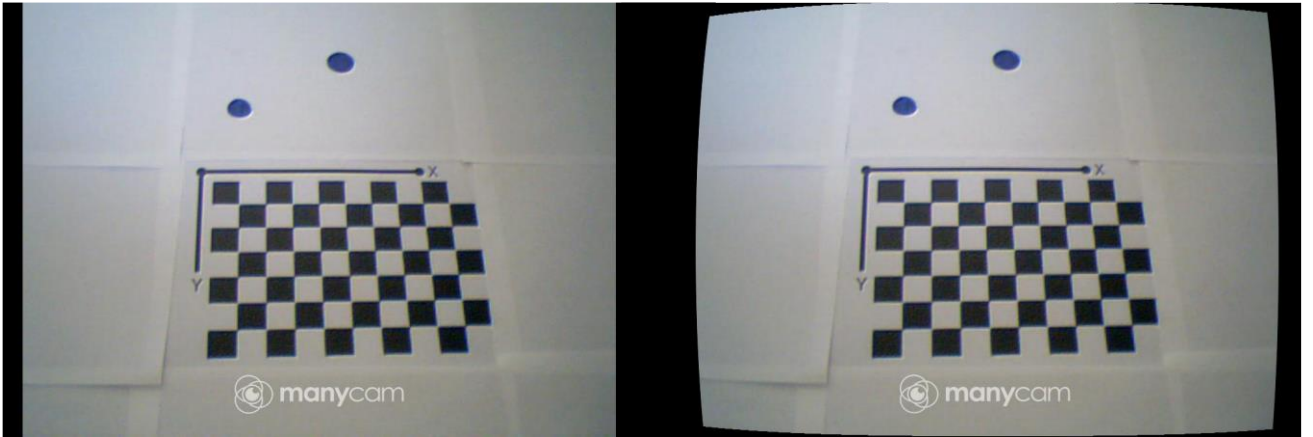


Figure 12. The original distorted image (left) and the undistorted image after estimation of camera parameters (right).

Figure 13 shows a plot magnitude of the reprojection errors. A reprojection error is a measure, in pixels, of how precise the estimation of the point's location is in the two-dimensional projection compared to its position in three-dimensional space.

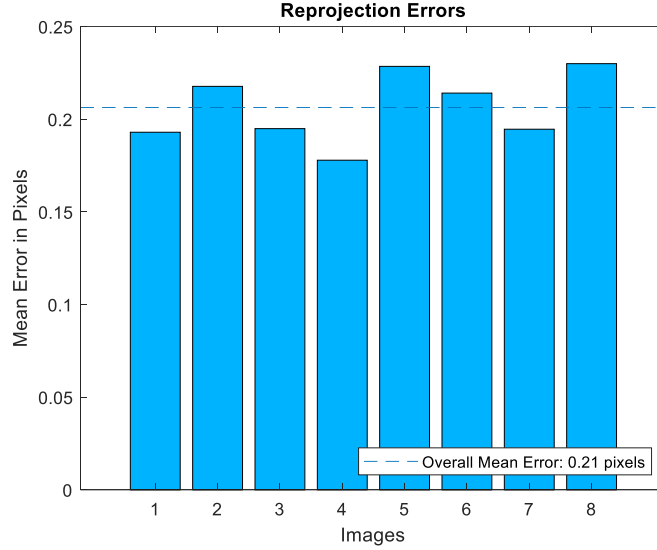


Figure 13. Reprojection error for each image in the calibration set.

The reprojection errors for each image are less than a pixel, which is generally considered suitable.

Using the calibrated camera, accurate feature recognition is possible. The DoD system can be configured to reduce the complexity of this process by specifying that points of interest have a consistent colour, and all other regions are consistently transparent or coloured differently. This specification allows for saturation thresholding to be used.

Images captured by the camera are initially in RGB (red, green, and blue) format. Each pixel in the image has three numbers associated with it which describe the red, green, and blue colour intensity, respectively. These values can be converted into HSV (hue, saturation, and value). Hue describes the wavelength of the colour of the pixel and value describes how light or dark the colour of a pixel is. Saturation is a measure of the intensity of the colour of the pixel. In an image that has clearly defined regions of colour, such as the image in Figure 12, would ideally have three dominating levels of saturation: one for the dark areas, a second for the light areas, and a third for the blue circles. In practice, low quality or poorly lit images have far less consistent saturation values. The computer vision toolbox in MATLAB has a function for converting RGB images to HSV, named ‘rgb2hsv’, which uses the method derived in [14]. Figure 14 shows a HSV representation of an image captured during the experiment apparatus calibration. In this representation of the image the two blue circles can be clearly identified as having significantly different saturation levels to their surroundings. There is some overlap between the saturation levels of the black squares in the checkerboard pattern, most likely due to low camera quality and poor lighting.

The image can now be converted into grayscale using image thresholding. ‘graythresh’ is another MATLAB function which computes the normalised threshold level for an image, allowing it to be converted to binary. By extracting the saturation channel of the HSV image and thresholding with respect to this channel, the image can be converted into a binary representation based on saturation values, shown in Figure 15. The blue circles are now segmented from their surroundings.

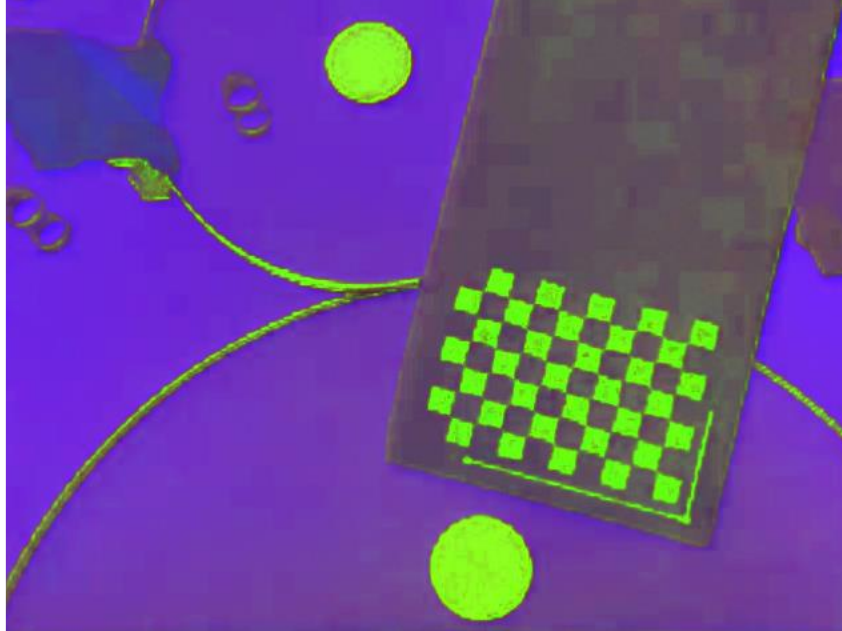


Figure 14. HSV version of test image.

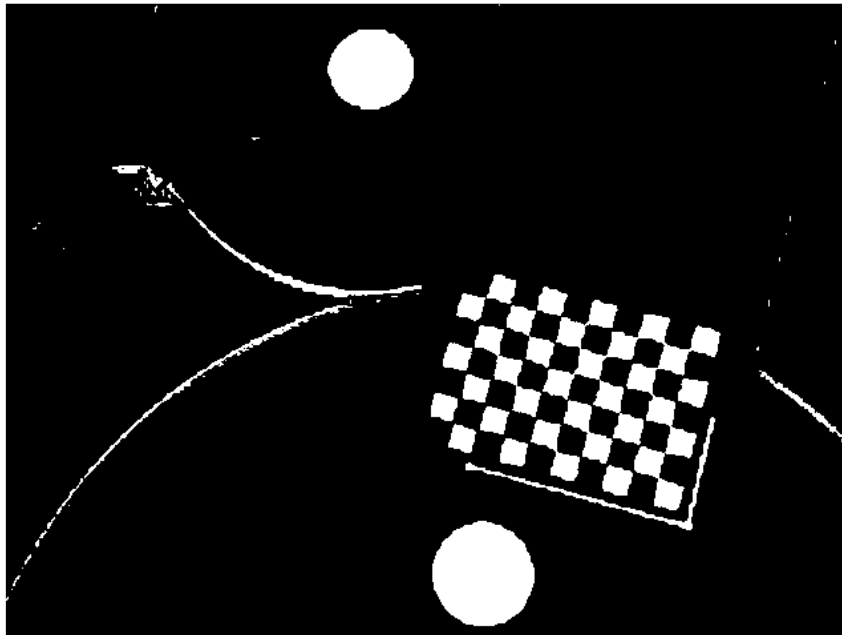


Figure 15. Segmented circles.

After the thresholding process each pixel has a value of either zero or one, representing whether the saturation of that pixel was below or above the mean saturation level of the HSV image. The blue circles are the only continuous region of consistent value, allowing the use of blob detection algorithms. The inbuilt blob detection algorithms in the MATLAB computer vision toolbox are able to compute statistics for regions of connected pixels in binary images, such as the one shown in Figure 15. The detection algorithm calculates the centroid of the detected blob and the bounding box. By setting a minimum and maximum size of the blob to be near the known size of the circles the blob detection result is very accurate. The circles are detected by the algorithm and the bounding region reflects the shape of the circle. Figure 16 shows the result of the feature detection using the process outlined.

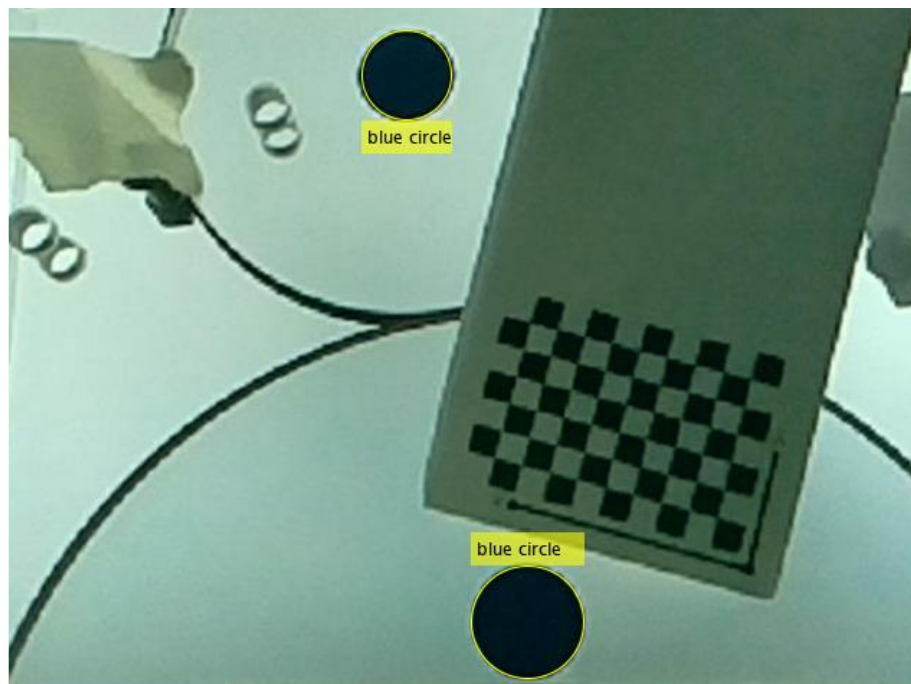


Figure 16. Result of feature recognition using saturation thresholding.

Using the statistics returned from the blob detection the size of the circles can be calculated. For the test image the measurement of diameter was accurate to within 0.5mm. The distance between the centroids can also be calculated using basic geometric relations, alongside the angle between the two circles.

In the final experiment apparatus, the camera will not move relative to the two disks during an experiment. Additionally, the object will only move in two dimensions at a fixed distance from the camera. Due to the combination of these factors the camera calibration parameters need only be computed once. The process of segmenting the points of interest from the surrounding environment is carried out only using matrix and binary operations, which are both efficient operations in the

MATLAB environment. This enables the feature recognition process to be carried out for each frame of a video stream in real time. The limiting factor on measurement frequency is assumed to be the frame rate of the camera.

The measurements taken from the computer vision algorithm are subject to uncertainty from several areas. The large pixel size in comparison to object size introduces errors, as well as the estimation process of camera parameters. Additionally, if either of the points of interest are not correctly identified, erroneous measurement values can be generated which greatly impacts the performance of the controller.

4.2.2 Implementation of the Kalman Filter

The state feedback with reference feedforward control structure shown in Figure 4 requires knowledge of the system states in order to calculate the required control law. In the DoD system only the position states, θ and ϕ , can be directly measured with a rotary encoder and computer vision, respectively. The velocity states cannot be directly measured. Additionally, the measurements of the position states are subject to uncertainty. To ensure the controller performs well an accurate estimation of the states must be computed. For this purpose, a Kalman Filter is used.

For the DoD model shown in (60), there is very little uncertainty associated with the derivatives of the position states, as the time derivative of position is velocity. There is a much more significant uncertainty associated with the velocity states. The initial prediction uncertainty describes the confidence in the initial state of the system, which is required to be experimentally determined. Measurement model uncertainty is a function of the camera. This is determined by taking a series of measurements of the positions of the disks while they are stationary. The measurement uncertainty is calculated by taking the square of the variance of the measurements from the mean.

Application of the Kalman Filter to the measurements obtained from the computer vision process generates a state estimation suitable for control, as shown in Figure 17. There is extremely good agreement between the predicted and measured states for a measurement frequency of 100Hz. For a more realistic 20Hz measurement frequency, shown in Figure 18, there is still good agreement between most states, with less strong agreement between the measured and estimated object angular velocity. This is due to the nature of the system, as this is the only state without a direct or inferred measurement, and thus the estimator has the least information to make predictions.

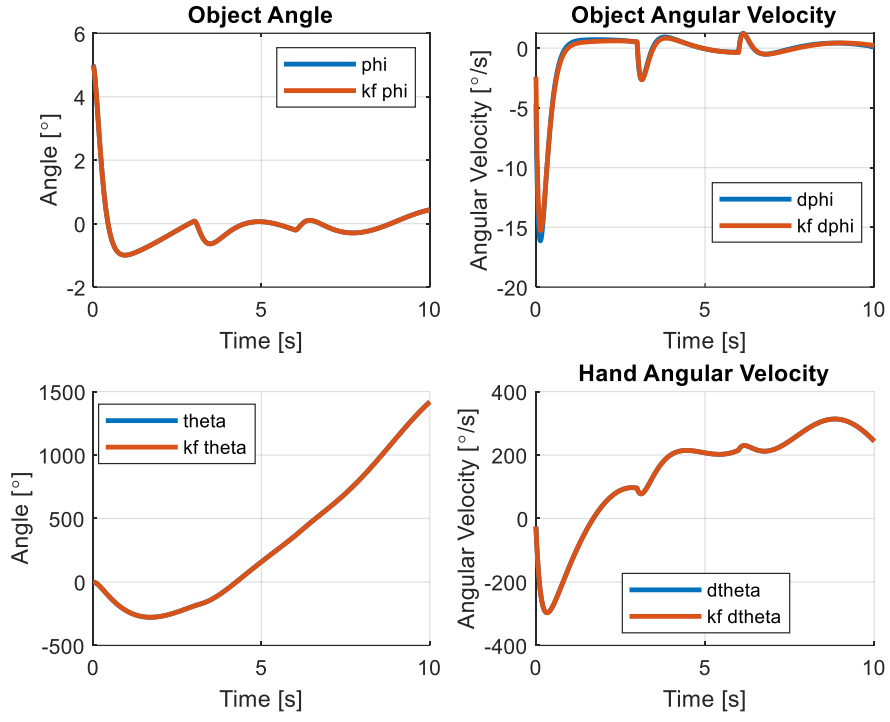


Figure 17. Performance of Kalman Filter at 100Hz.

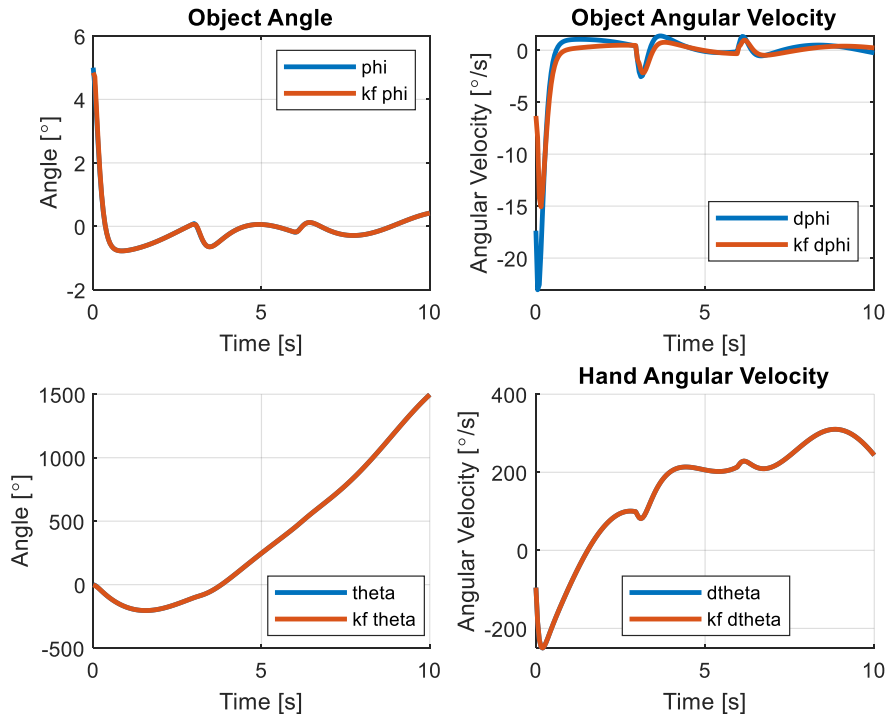


Figure 18. Performance of Kalman Filter at 20Hz.

5 Experiment Apparatus Design

While modelling and simulation are suitable for a proof of method, they do not reflect the complexities of a fully realised system. A real system will have a non-ideal motor that is subject to gearbox friction and non-linear torque curves; the disks may not be perfectly round, or have homogenous material properties leading to a centre of gravity that is not located at the geometric centre. The interaction between the two disks may be subject to slipping or skidding. These factors warrant the manufacture of an experiment apparatus to validate the control process. The design of the apparatus is based on the results of the simulation.

5.1 Mechanical Design

There are few constraints on the design of the system to preserve model accuracy. The primary constraints are that the disks be as round as possible, and that they are held in plane perpendicular to the ground. Most other design choices are driven by ease of construction, simplicity, and material availability.

The disks are held between two clear enclosure plates. The front plate must be clear to allow the computer vision process to see the object.

Enclosure plates are milled from 10mm polycarbonate sheet. Polycarbonate is a thermoplastic with good machinability and optical properties. The disks are also milled from this sheet. Perspex is an alternative material but is less flexible, more prone to shattering, and demands a significantly higher financial investment.

A frame suspends the enclosure plates and disks from the ground, provides stability against tipping, and provides mounting points for the hardware and camera. The frame is constructed from 20 x 20 x 2mm aluminium square section, with a custom bent U-channel section running the length of the centre of the frame. This section allows mounting of the PCB and motor. The use of a U-channel section ensures the mounting, and in turn the frame, are resistant to torsional forces imparted by the motor. As the inertia of the lower disk in relation to the size of the system is not insignificant, this interface between motor and frame is of particular importance.

Torque is transferred from the motor to the hand via a keyed shaft. Keyed shafts are appropriate for a significant level of torque transfer, and double as a method of axial locating of components. As polycarbonate is a brittle material, the key and keyway were manufactured larger than standard

woodruff keys in order to dissipate loads and reduce risk of shattering. The shaft diameter is 12mm, with a rigid coupling used to connect the motor output shaft and main drive shaft.

The shaft supports the load of the two disks and interacts with the enclosure plates through a set of bearings. Bearings and shafts are off-the-shelf items appropriate for the size of the components.

Figure 19 shows a general assembly of the apparatus.

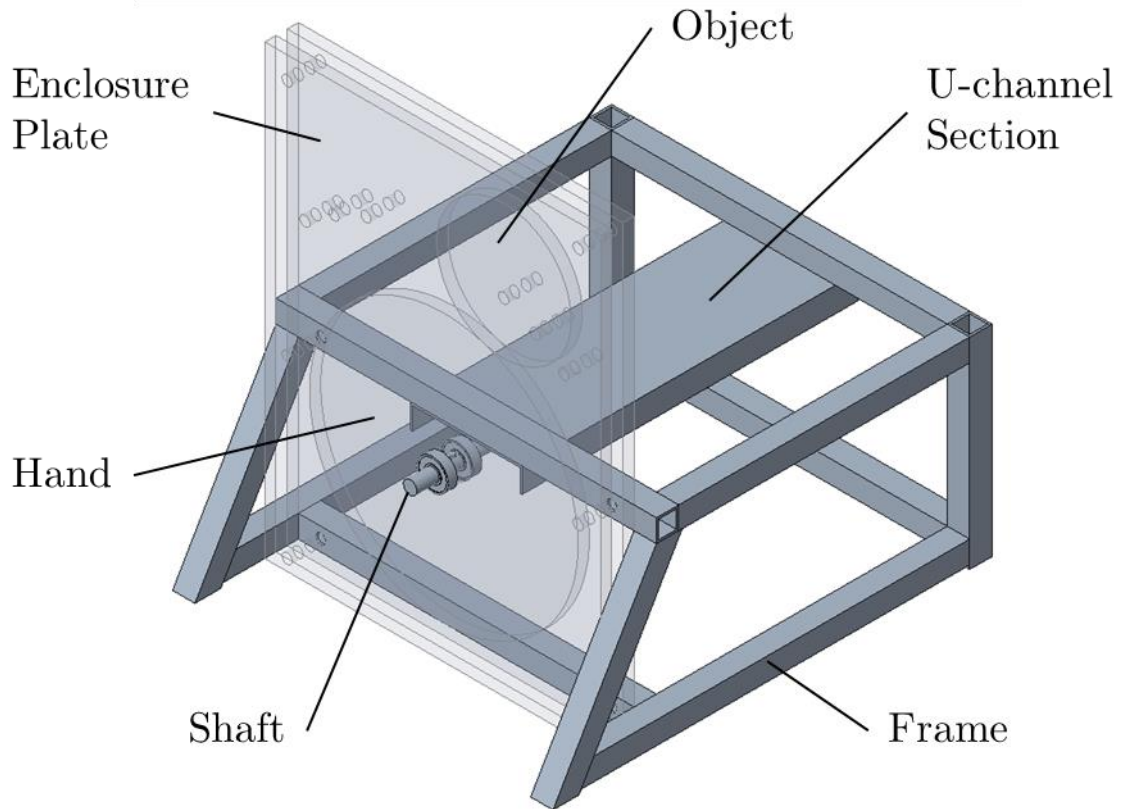


Figure 19. Isometric view of experiment apparatus general assembly.

Figure 20 shows the completed experiment apparatus.

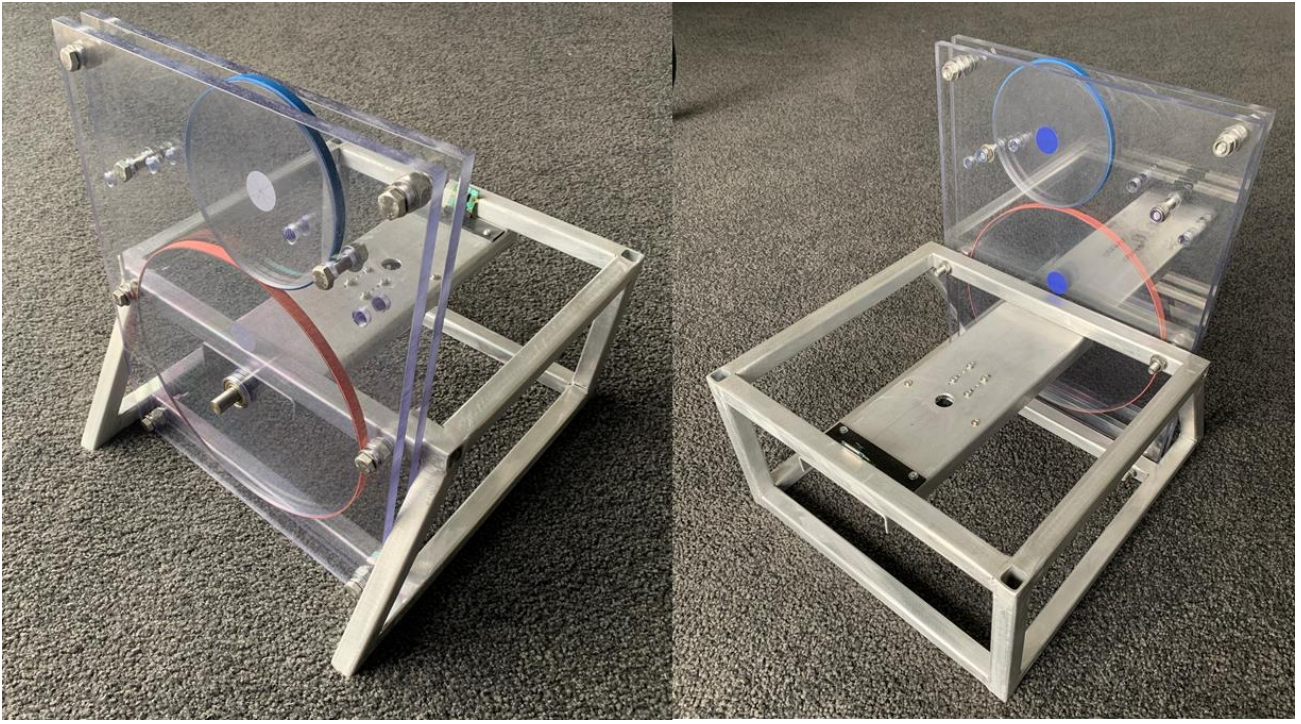


Figure 20. Experiment Apparatus.

Object and hand size determine the speed of the system dynamics and the required control torque. Geometrically smaller systems exhibit much faster system dynamics and demand higher frequency control and measurements. Larger systems do not have such high frequency dynamics but require larger control forces due to greater inertia. Hobbyist electronic motors are often available in high-torque or high-speed options as these are inversely related. The model and simulation developed allow the torque and angular velocity demands of the system to be estimated. Figure 21 shows the angular velocity of the hand and object subject to a control law. With an object diameter of 120mm and a hand diameter of 240mm, the torque demanded for a 120RPM change are below the maximum output of a hobbyist DC motor such as the Pololu 4846.

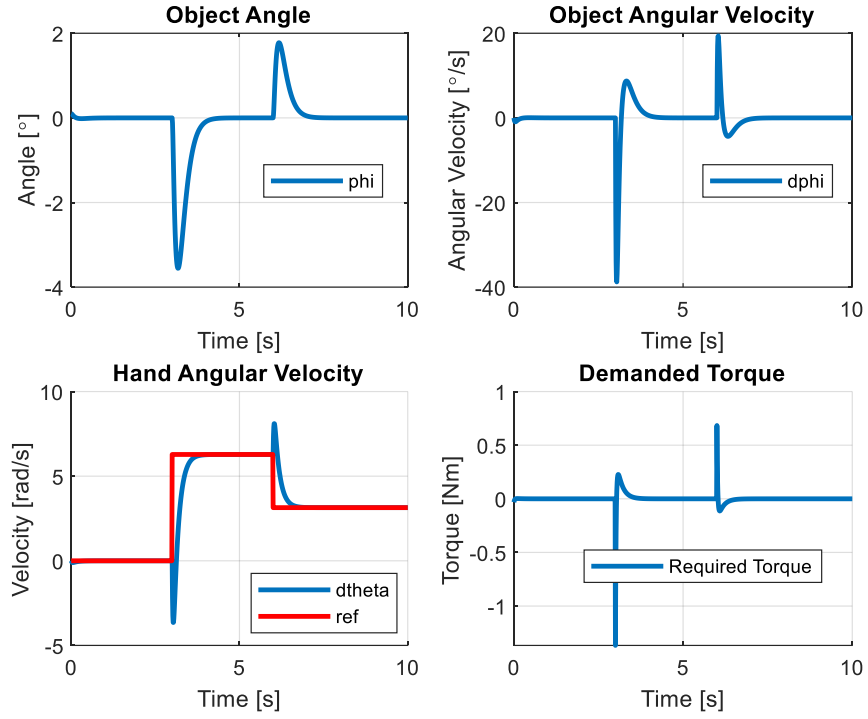


Figure 21. Angle, angular velocity and demanded torque plots for applied control law at $T=0.01s$.

Stability of the system is also significantly affected by the discrete timestep of the system. All simulations are conducted with a timestep of 0.01 seconds. This timestep is also likely to be significantly faster than the dynamics of the system. If the timestep is reduced to 0.15 seconds, the system becomes unstable, as shown in Figure 22.

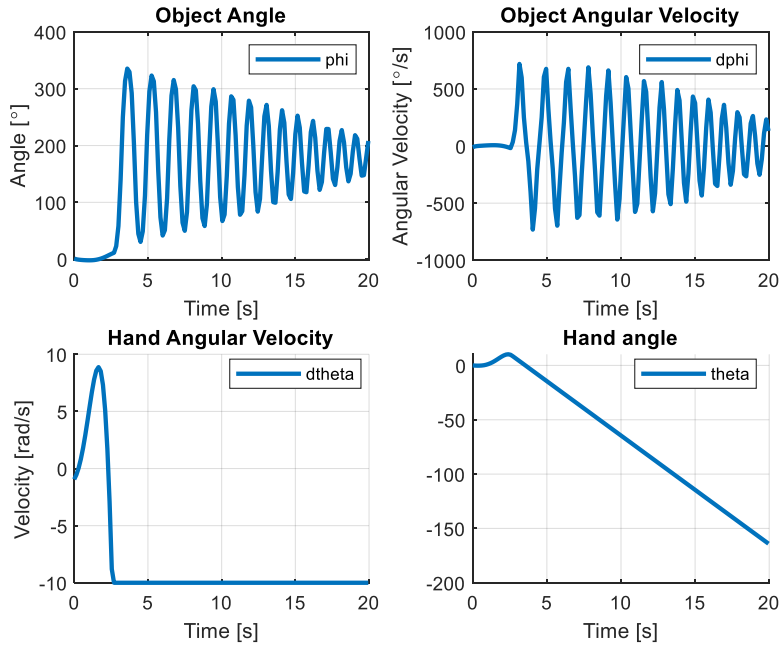


Figure 22. Angle, angular velocity and demanded torque plots for applied control law at $T=0.15s$

The same effect can be achieved by changing the size of the disks, with smaller disks requiring a faster controller frequency, and the inverse for larger disks.

A hand diameter of 240mm and object diameter of 120mm, coupled with a control frequency of 30Hz exhibits good stability in simulation, indicating that the system dynamics for the chosen design scale will not impede controllability.

5.2 Electrical Design

For the experiment to function there must be communication between the sensors, the actuator, and the controller. This communication should be as close to real-time as possible, with minimal delays or interruptions.

A STM32 microcontroller is selected to handle sensor interfacing and control of the DC motor. The STM32F446RE can run at 100MHz and offers ample GPIO pins for control of a motor driver and features onboard analogue to digital converters (ADC). One channel of the ADC is used in conjunction with a voltage divider to determine accurate motor armature voltage, necessary for the system identification process.

Motor driving functions are handled by a VNH5019 fully integrated H-bridge with onboard current sense. This board is selected due to its high continuous current output and high frequency PWM capabilities.

Actuation is handled by a 12V DC motor (Pololu 4847) with a peak torque output of 2.11Nm. This is achieved through a 99:1 planetary gearbox, which also features a 48 CPR quadrature encoder on the output shaft, giving 4741.44 counts per revolution of the output shaft.

5.3 Safety Considerations

As there are several moving parts and electrical components, safety considerations must be made. All moving parts of the system are enclosed or shielded from observers. The two disks are enclosed in the enclosure plates, ensuring entanglement is highly unlikely. Additionally, stoppers are included in the plates to stop the top disk from falling out of the system when balancing is not maintained.

The motor and shaft arrangement are located under the U-channel section, providing a level of protection from entanglement in moving parts. The PCB and electronics are also affixed to the U-channel using insulated stand-off bolts to reduce risk of electrocution in the event of a short-circuit or incorrect ground.

6 Implementation

The implementation process involves assembly of the experiment apparatus, developing the software required for the embedded system, setting up sensors and hardware interfacing, and extending the computer vision process to run in a near real-time stream. This process has two main challenges: system identification for the DC motor to enable precise control, and implementation of the computer vision stream.

6.1 DC Motor System Identification

DC motors are usually sold with a datasheet containing some, or all, of the parameters necessary to complete a mathematical model. Unfortunately, these parameters are rarely experimentally determined, and are far more likely to be theoretically derived. This causes these parameters to either be inaccurate or only valid for a very small subset of possible operating conditions.

The standard brushed DC motor model, shown in Figure 23, is relatively simple and can be linearised by assuming the rate of change of current is very slow.

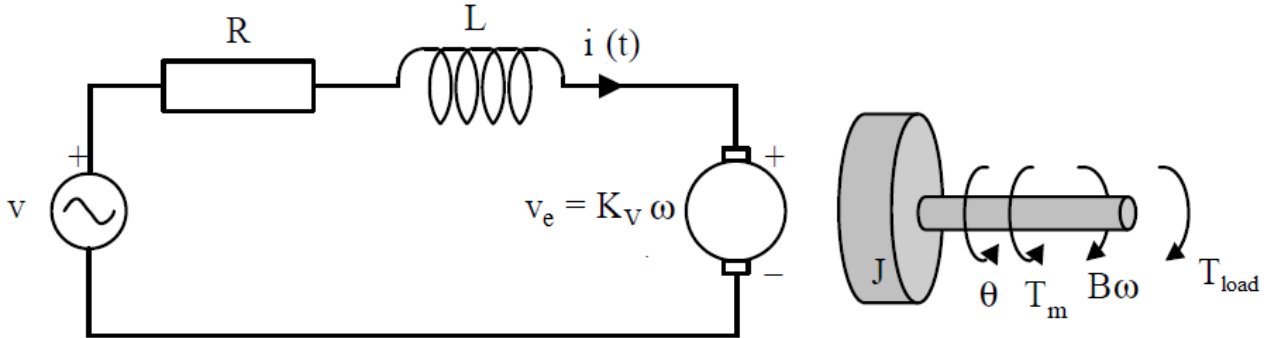


Figure 23. Standard permanent magnet brushed DC motor model [15].

This model can be used to derive the mathematical model, which is a pair of differential equations.

$$\frac{d\omega_m}{dt} = \frac{\tau_m - B\omega_m}{J} \quad (68)$$

$$\frac{dI_a}{dt} = \frac{V_a - R_a I_a - K_\omega \omega_m}{L_a} \quad (69)$$

Since a model of the motor is known, the system identification process can use a ‘grey box’ approach, which involves estimating parameters of the model using data gained from an experiment. Grey box methods are preferred to black box methods as they often allow better extension of the model to operating conditions not present in the experimental data.

The system identification process is a staged process, with model complexity being increased at each stage. Initially, only the electrical subsystem is considered.

The parameter estimation process utilises the constrained optimisation and global optimisation toolboxes present in MATLAB. The armature voltage, armature current and output velocity of the system are measured while the system receives a predetermined input. These measurements can then be used as either inputs to the simulated system, or as the true values for which the simulation outputs are tested against. The cost function being minimised is a least squares function, with weightings applied when there are multiple outputs.

6.1.1 Estimation of Resistance and Motor Constant

By considering only the electrical subsystem, and assuming the rate of change of current is approximately zero, (69) can be rearranged to give

$$V_a = R_a I_a + K_\omega \omega_m \quad (70)$$

By calculating the armature voltage at each timestep using the measured current and velocity, and minimising the least squares cost between the measured and calculated values, the parameters R_a and K_ω can be determined using a constrained optimisation routine in MATLAB.

Figure 24 shows the result of this process. The agreement between the measured and simulated values is acceptable given the assumptions made.

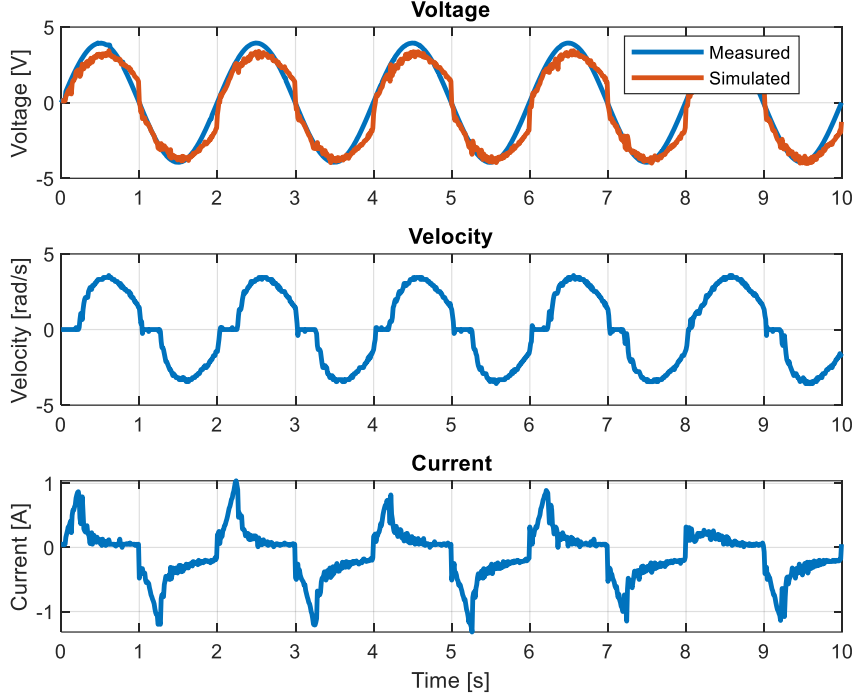


Figure 24. Result of first stage parameter estimation.

In the velocity subplot the shape of the velocity is noteworthy. Despite the sinusoidal input voltage, the output velocity exhibits significant deadzone behaviour. Such behaviour indicates the system is non-linear, and the model proposed in (68) and (69) will not be suitable without modification. This type of deadzone behaviour is common for motors utilising high-ratio gearboxes, such as the one chosen for this project.

6.1.2 Estimation of Armature Inductance

To estimate the inductance parameter a similar approach is used. With armature voltage and shaft velocity as inputs, an ODE solver can be used with (69) to determine the current at each timestep. The only unknown parameter at this stage is the inductance, which is the focus of the minimisation routine. By comparing the measured and simulated current, again using least squares, an estimate of the inductance parameter can be found. As a validation, the armature voltage is also back-calculated by rearranging (69) and using the newly simulated current values. Figure 25 shows the result of this process. There is extremely good agreement between the measured and simulated voltage, and relatively poor agreement between the measured and simulated current.

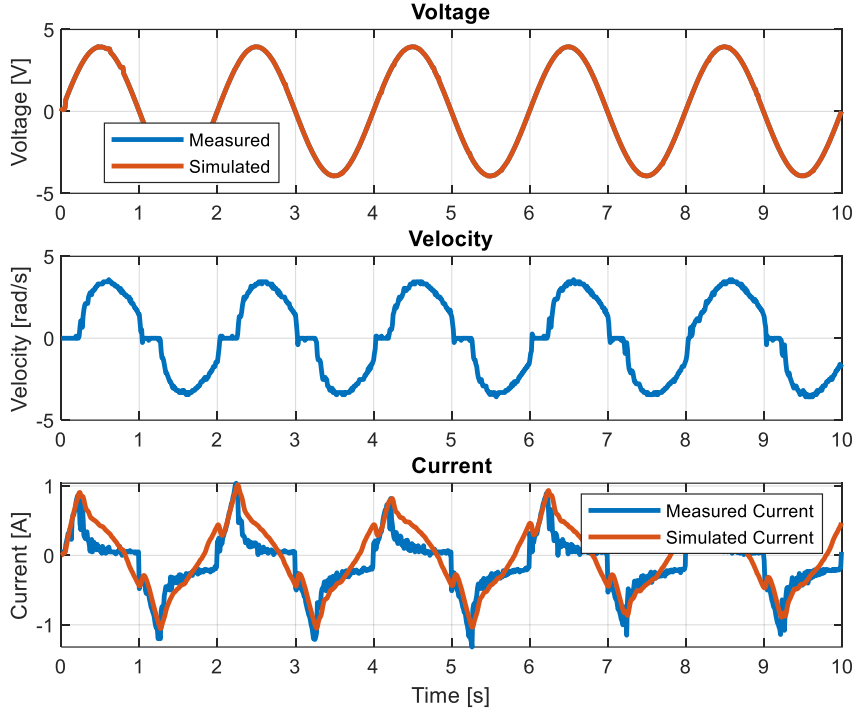


Figure 25. Result of second stage parameter estimation.

Again, the presence of a deadzone near zero velocity appears to greatly reduce the ability of the standard DC motor model to capture the behaviour of the system. To capture this behaviour as part of the electrical subsystem, a non-linear model for armature resistance can be implemented. Using

$$R_a = (R_1 + R_2 e^{-R_3 |I_a|}) I_a \quad (71)$$

attempts to account for the increase in current near zero velocity by increasing the resistance as the current approaches zero. This model for current is visualized in Figure 26 below. The exponential causes the resistance to increase sharply near the zero point, and taper to the value of R_1 outside of this region.

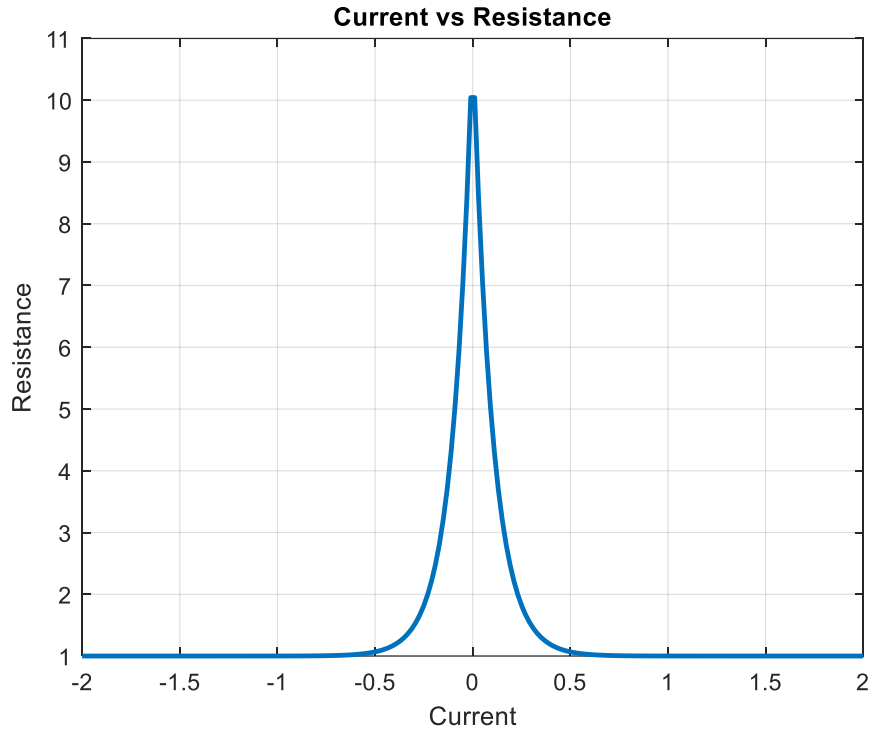


Figure 26. Current vs resistance for nonlinear resistance model.

Parameter estimation using (71) is ultimately no more successful than using a constant resistor model, indicating that the deadzone behaviour is not a result of a nonlinear resistance. A constant model for resistance is used to maintain model simplicity.

6.1.3 Estimation of Torque Curve

To determine the torque curve of the DC motor the experiment shown in Figure 27 is carried out. This experiment allows output torque for a specified voltage and current to be measured using a set of scales. The motor, with a set length arm attached, is driven with a range of input voltages until stalling, and the corresponding weight reading on the scales is recorded. The output torque can then be calculated using simple mechanics. Data is recorded for voltages between $\pm 6V$, capturing the entire required operating range. Despite the motor being capable of $\pm 12V$, early tests indicated the lower voltage range provided sufficient input torque to correct angle displacements greater than 10 degrees, and ensured the motor is operating well within the recommended load limits.

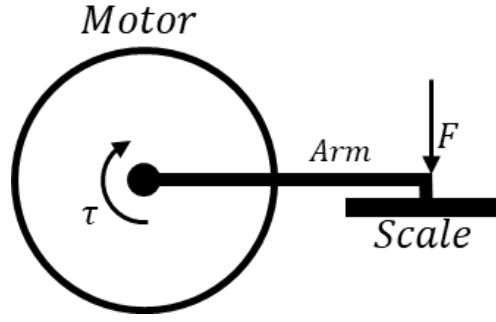


Figure 27. Experiment setup to determine motor torque characteristics.

Once the data has been collected a model that maps output torque to current can be determined. Typically, this mapping is done in the opposite way, but developing the torque-current mapping greatly simplifies a substitution required in the control allocation stage.

Figure 28 shows the collected datapoints and the fitted functions. It is clear the current-torque curve is not linear, as would be ideal, which is not unexpected due to the deadzone behaviour observed earlier. Initially, individual curves are fitted to the positive and negative sets separately, prior to fitting a polynomial to the entire data set. The final model is a fourth order polynomial. The ability to model the entire torque curve with a single smooth model is imperative to ensure the parameter estimation process can be numerically optimised as MATLAB's solvers perform extremely poorly when used on non-smooth systems due to their use of numerical integration methods.

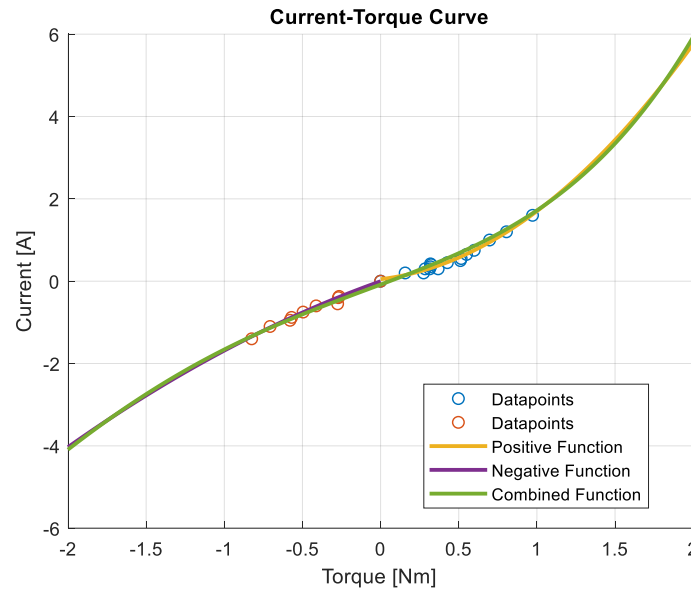


Figure 28. Current-Torque curve.

6.1.4 Estimation of Friction

As the parameters for the electrical subsystem of the DC motor are now estimated and the torque characteristics have been modelled, the fourth stage involves determining a suitable friction model and estimating the required parameters.

For the estimation of friction, the mechanical subsystem is included in the simulation. The only input to the simulation is the armature voltage, with (68) and (69) being solved using an ODE solver.

6.1.4.1 Viscous Friction

One of the least complex and most common forms of friction is viscous friction, shown in (68) as the $B\omega$ term. This model opposes motion with magnitude directly corresponding to the magnitude of the velocity and has no ability to account for the deadzone behaviour near zero. Figure 29 shows the results of the parameter estimation with a viscous friction model. As shown in velocity and current subplots, the viscous friction model does not exhibit strong ability to capture the system dynamics. The simulated velocity is out of phase and very poorly describes the behaviour near the zero-velocity point. Likewise, the simulated current only very vaguely matches the measured behaviour, with very little shape similarity.

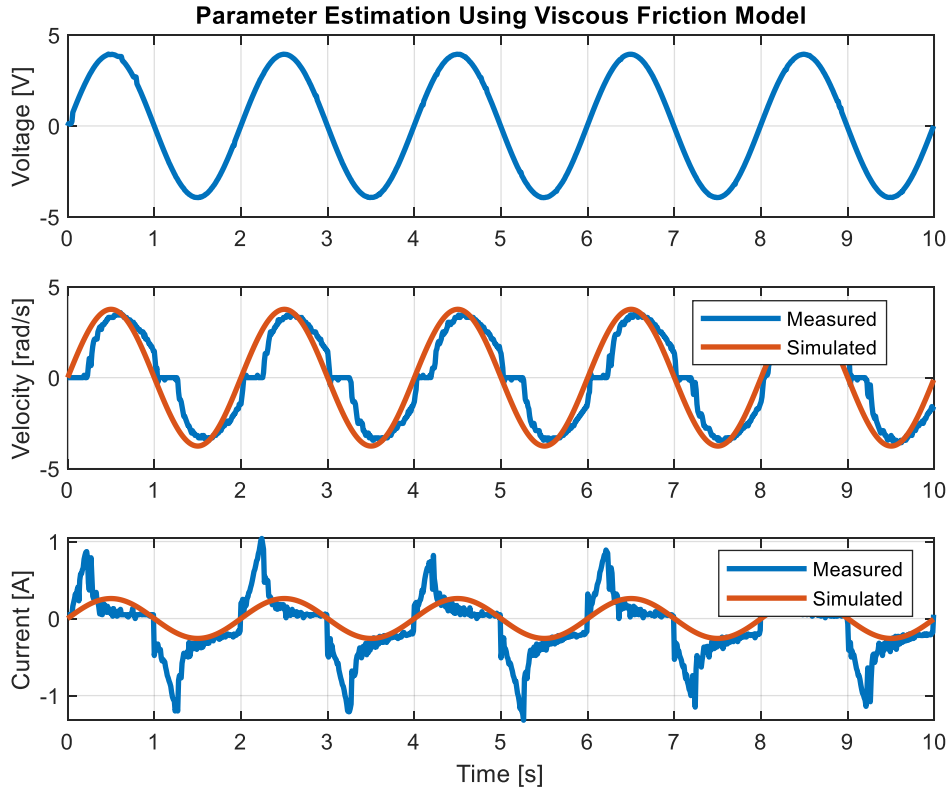


Figure 29. Parameter estimation using viscous friction model.

6.1.4.2 Stribeck Friction

Stribeck friction is a non-linear friction that includes a velocity dependence in the friction model. The model uses an exponential term to capture behaviour near zero and includes a cut-off term. When the velocity approaches the cut-off velocity, the Stribeck friction reduces to half of the peak magnitude, and tapers to zero at twice the cut-off value. Typically, Stribeck friction models include a viscous friction term to account for the region beyond the cut-off velocity.

Figure 30 shows the Stribeck friction model,

$$F_s = e^{\left(\frac{\omega}{\omega_{cutoff}}\right)^2} * \text{sign}(\omega) \quad (72)$$

and the combined Stribeck and viscous model,

$$F_{total} = F_c \omega + F_s \text{ where } F_c = B \quad (73)$$

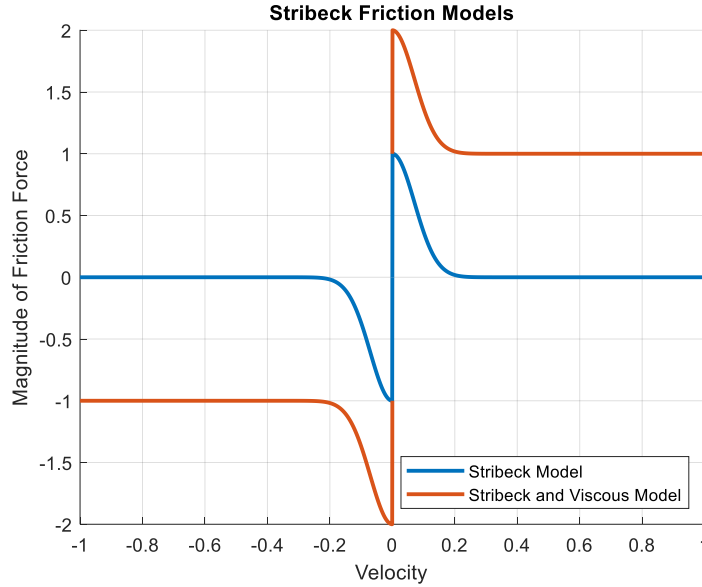


Figure 30. Stribeck friction models.

The models shown in (72) and (73) are non-smooth due to the use of the *sign* function. Non-smooth functions are unsuitable for use with MATLAB ODE solvers, as they can cause the step size to become infinitely small and greatly increase computation time. To alleviate this issue, the following alternative *sign* function is used.

$$\text{sign}(x) \approx \frac{x}{\sqrt{a + x^2}} \quad (74)$$

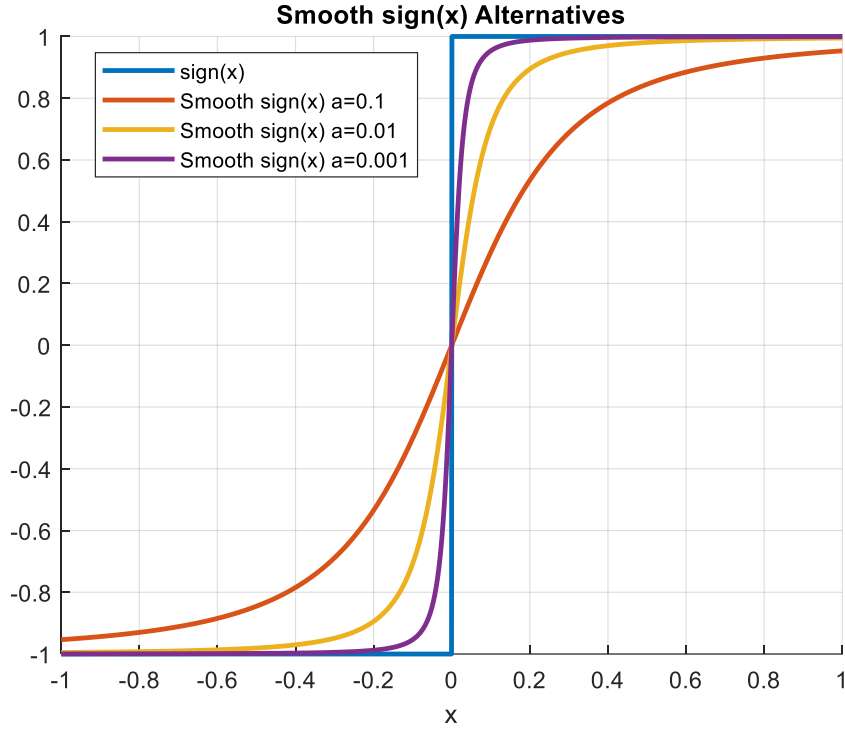


Figure 31. Smooth alternatives to the $\text{sign}(x)$ function.

As shown in Figure 31, for small values of a , the smooth approximation provides sufficient agreement with the non-smooth function.

Using the combined Stribeck and viscous friction model, alongside the smooth approximation of the *sign* function, the parameter estimation can again be carried out. Figure 32 shows the result of this process. Using the more sophisticated friction model, the simulated velocity shows far greater shape similarity to the measured velocity, particularly near zero velocity, where the Stribeck friction is most prominent. The increased friction force near zero due to the Stribeck component clearly improves the ability of the model to capture the deadzone behaviour of the DC motor. Similarity between the simulated and measured current is also improved, again likely due to the more sophisticated friction model accurately reflecting the need for greater current loads to overcome the friction present near zero velocity.

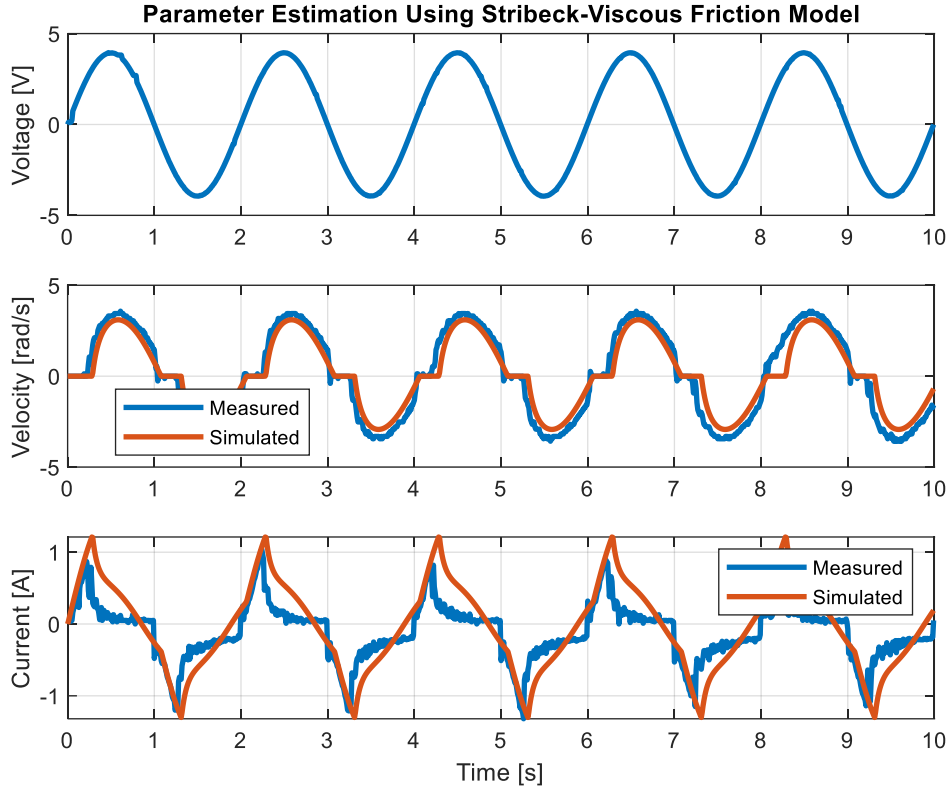


Figure 32. Parameter estimation using combined Stribeck-viscous friction model.

6.1.4.3 Continuously Differentiable Friction Model

A continuously differentiable friction model that combines static friction, Stribeck friction, Coulomb friction, and viscous friction is proposed in [16]:

$$F_{total} = a_1(\tanh(B_1\omega) - \tanh(B_2\omega)) + a_2 \tanh(B_3\omega) + a_3\omega \quad (75)$$

This model is particularly useful in this context, as it captures multiple friction models in a continuously differentiable manner, allowing the numerical based solving methods of the parameter estimation process to function. Static friction is captured by the a_1 and a_2 terms, whilst the Stribeck and Coulomb effects are captured by the $\tanh(B_1\omega) - \tanh(B_2\omega)$ and $a_2 \tanh(B_3\omega)$ terms, respectively. The $a_3\omega$ term captures the viscous friction. Figure 33 shows the component friction profiles alongside the total friction profile of (75) with $a_1 = 0.25, a_2 = 0.5, a_3 = 0.5, B_1 = 100, B_2 = 1$ and $B_3 = 100$ [17].

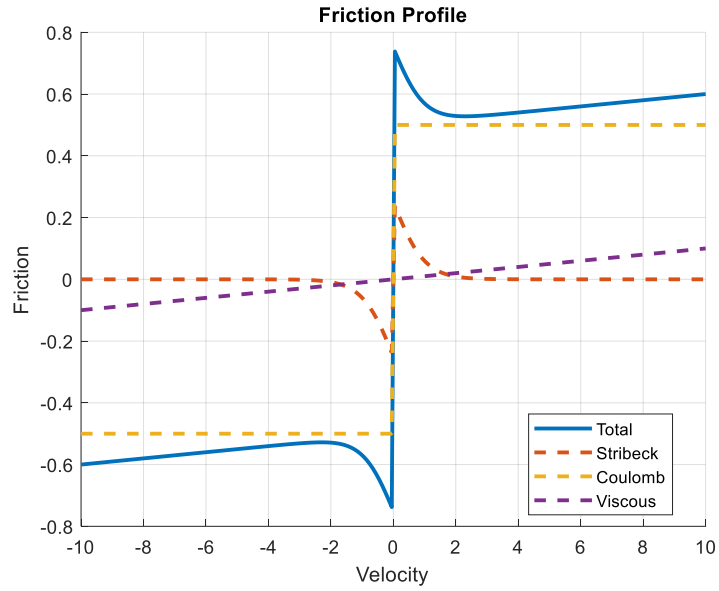


Figure 33. Friction profiles of component models and total model.

Using this new model, the parameter estimation process is again performed with the six friction coefficients as the parameters in focus. The result, shown in Figure 34, is far more promising than any previous result. The simulated and measured velocity are very similar, with most of the deadzone behaviour being captured. Including multiple types of friction models clearly improves the predictive ability of the model despite the large amount of nonlinear behaviour near zero velocity.

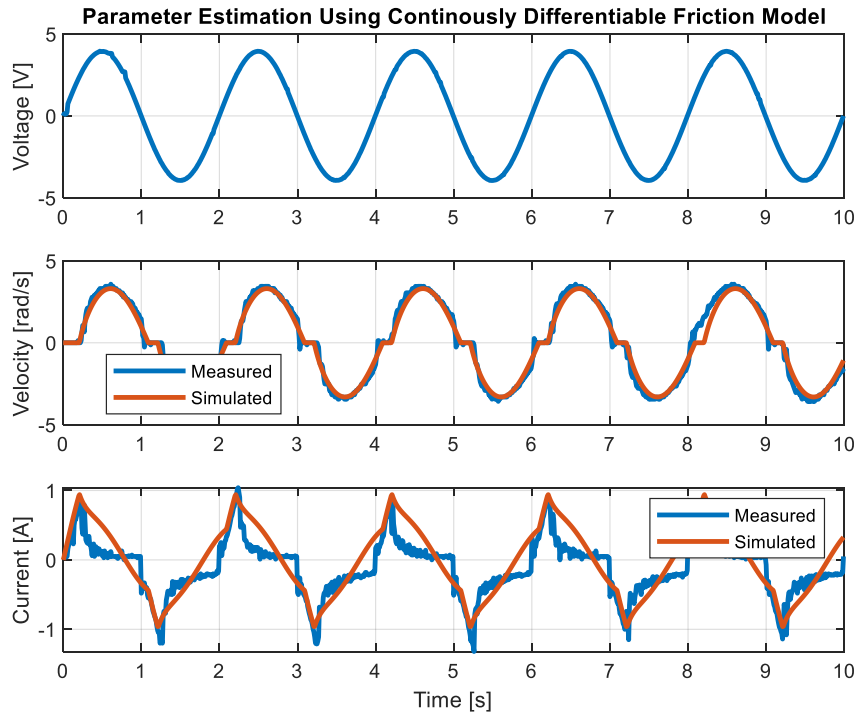


Figure 34. Parameter estimation using continuously differentiable friction model.

A drawback of permanent magnet DC motors is parameter variation dependent on the direction of excitation. When rotating in the positive direction (positive armature voltage), the motor appears to be more efficient than when rotating in the negative direction. This is most notable in the current subplot of Figure 34, where the motor current following the breakaway point is approximately 0.1A in the forward direction, whilst it is approximately 0.3A in the reverse direction. Asymmetric behaviour such as this indicates asymmetric friction models may be suitable.

Asymmetric models are typically implemented utilising a Heaviside activation function, which effectively turns on or off the corresponding model dependent on the value of a variable. In this case, the variable in question is the shaft angular velocity. The Heaviside function is given by

$$H(x) = \begin{cases} 0, & \text{for } x < 0 \\ 1, & \text{for } x \geq 0 \end{cases} \quad (76)$$

which is non-smooth. A smooth approximation of the utilises the logistic function

$$H_{smooth}(x) = \frac{1}{1 + e^{-2kx}} \quad (77)$$

where larger values of k correspond to a better approximation of the Heaviside function. A visualisation is shown in Figure 35.

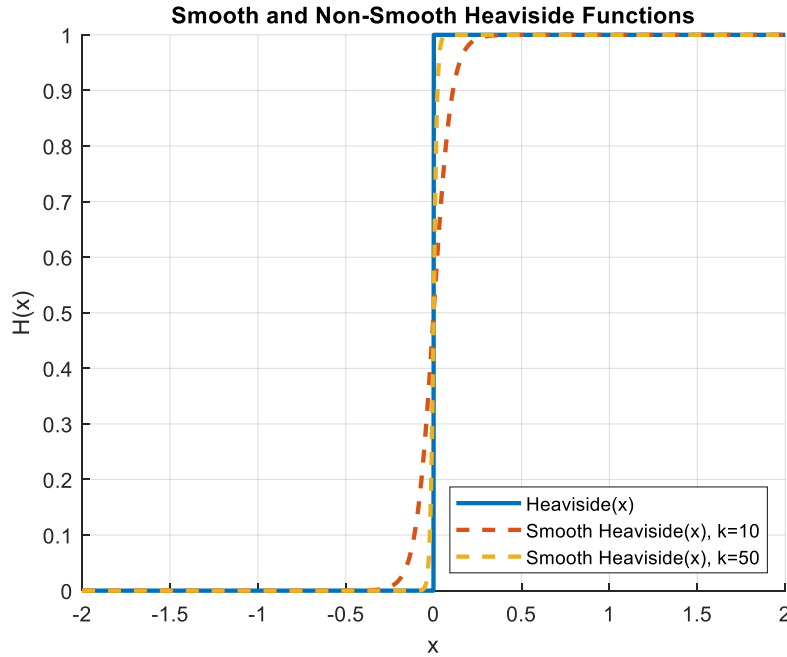


Figure 35. Smooth and non-smooth Heaviside functions.

Utilising this smooth approximation of the Heaviside function, an asymmetrical continuously differentiable friction model is developed

$$F_{total} = (a_1(\tanh(B_1\omega) - \tanh(B_2\omega)) + a_2 \tanh(B_3\omega) + a_3\omega) * \frac{1}{1 + e^{-2k\omega}} + (a_4(\tanh(B_4\omega) - \tanh(B_5\omega)) + a_5 \tanh(B_6\omega) + a_6\omega) * \frac{1}{1 + e^{2k\omega}} \quad (78)$$

Using (78), the parameter estimation is performed, with results shown in Figure 36.

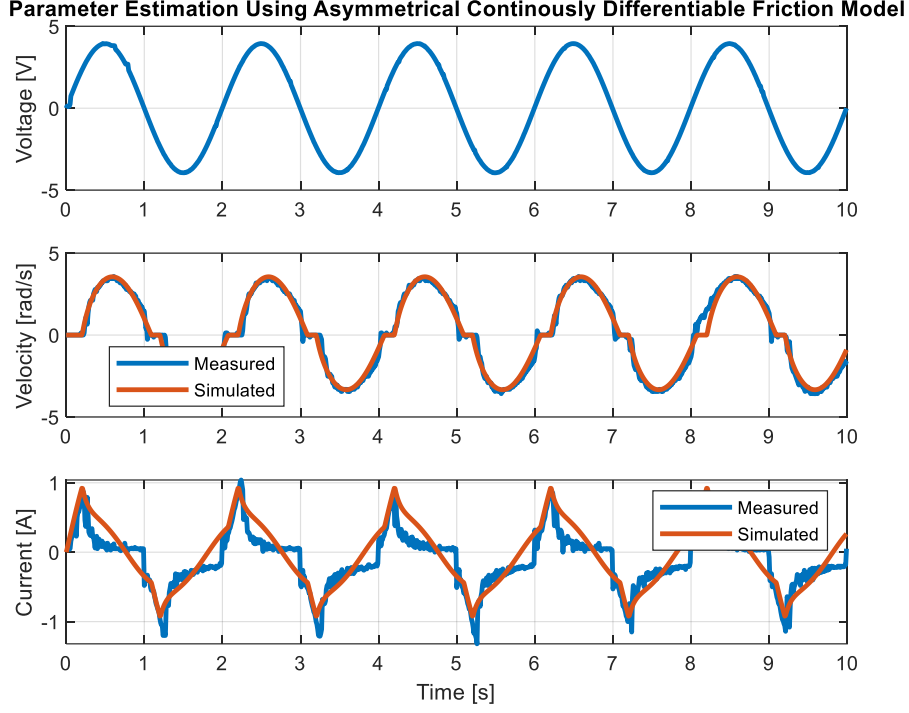


Figure 36. Parameter estimation using asymmetrical continuously differentiable friction model.

The asymmetrical model offers slight improvement over the symmetrical model, particularly in the agreement between the simulated and measured velocities in the positive region. This is likely due to the increased friction in the reverse direction, and the ability of the solver to isolate the forward and reverse direction friction due to the asymmetric model.

Despite the sophistication of the asymmetrical model, the simulated and measured armature current plots show only very general agreement. Experiments conducted in parallel with this project by the project supervisor using identical current sourcing hardware determined the measurements acquired may be inherently inaccurate due to the measurement method, and a decision is made that the results shown in Figure 36 are suitable for the purpose of control allocation. Future works are recommended to utilise purpose-built current sensing hardware as sensing methods found onboard the motor driver are likely to be of low quality and highly inaccurate.

6.2 Control Allocation

As the model developed in section 3.2 has a torque input, the natural input to the DC motor is desired armature current due to the current-torque relationship developed in section 6.1.3. As the current sense measurements are assumed to be inaccurate, the control allocation is extended to calculate a desired armature voltage to be applied to the motor.

6.2.1 Torque Control

Using the equations developed when modelling the system, and the parameters determined during the system identification process, a control allocation process with a desired torque input and voltage output to be commanded can be developed.

The control allocation process is most effective when run at significantly higher frequencies than the controller frequency, ensuring the commanded torque has been achieved prior to the next control action. Algorithm 3 shows the torque control allocation process, which is implemented at 200Hz on the STM32.

Algorithm 3 – Torque Control Allocation

Given $\hat{\tau}$

while(controller is running)

1. Get measurement of velocity, ω

2. Calculate friction, F_t

$$F_{total} = (a_1(\tanh(B_1\omega) - \tanh(B_2\omega)) + a_2 \tanh(B_3\omega) + a_3\omega) * H(x) \\ + (a_4(\tanh(B_4\omega) - \tanh(B_5\omega)) + a_5 \tanh(B_6\omega) + a_6\omega) * (-H(x))$$

3. Calculate desired motor torque, τ_m

$$\tau_m = \hat{\tau} - F_t$$

4. Calculate desired armature current, I_a

$$I_a = p_1\tau_m^3 + p_2\tau_m^2 + p_3\tau_m + p_4\tau_m$$

5. Calculate desired armature voltage, V_d

$$V_d = K_\omega\omega + R_a I_a$$

end

Unfortunately, torque-based control allocation is extremely ineffective when implemented on the DC motor. Much of the friction model, used to calculate the desired current, relies on the use of the

Heaviside function, which is equal to zero for a zero input. Additionally, the hyperbolic tan function is also equally zero for a zero velocity, as is the back electromotive force term $K_{\omega}\omega$. Since the velocity of the motor shaft is exactly zero until the static friction effects have been overcome, the control allocation process is unable to function until the velocity is non-zero.

A possible solution to this issue is to determine the intended direction of rotation and evaluate the friction equation for a very small velocity. Implementation of this solution results in a significant level of chatter and very poor performance due to the rapid switching between model-based and manually determined friction.

To highlight the complications involved with high ratio gearboxes, a deadzone test is performed. Figure 37 shows the result of incrementally increasing and decreasing the armature voltage of the motor, and the resulting velocity and current draw. The effects of static friction are highlighted by the significant difference in current draw at $\omega = 0$ and at $\omega \neq 0$. Due to the issues highlighted above, and the noise present in current measurements, it is determined that angular velocity-based control may be more suitable.

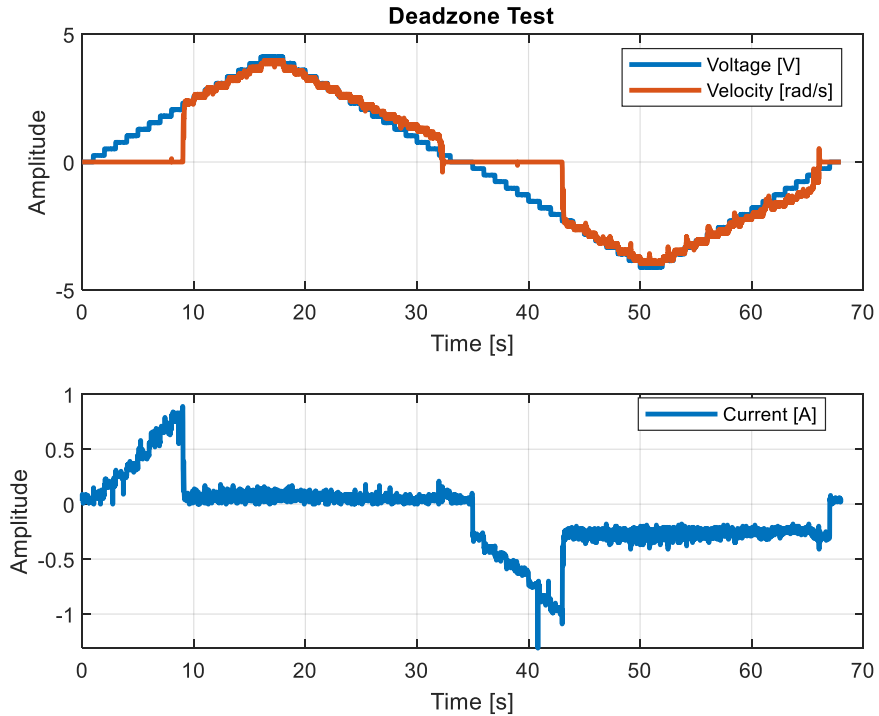


Figure 37. Results of deadzone test showing large initial friction in mechanical subsystem.

6.2.2 Velocity Control

With the alternate plant, controller and observer model derived in section 4.1.2, the system is suitable for velocity-based control allocation. As the results of the parameter estimation process were not promising, proportional-integral control (PI) is used for velocity control. PI control uses both an instantaneous and cumulative error between the desired and actual output to determine the adjustment to be made to the input voltage, as follows

$$V_{d_{k+1}} = c_1(\omega_{d_k} - \omega_k) + c_2 \sum_{i=k-1}^n \omega_{d_i} - \omega_i \quad (79)$$

With $c_1 = 3.5$ and $c_2 = 0.7$, the control allocation is very effective in delivering the demanded output velocity, as shown in Figure 38.

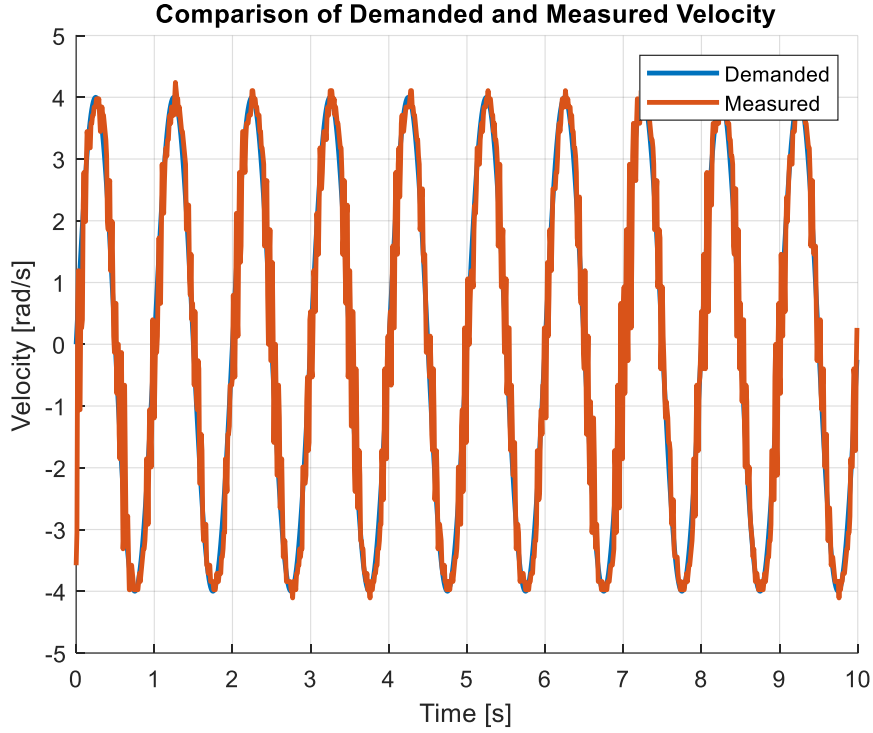


Figure 38. Comparison of demanded and measured velocity for 1hz sine wave input.

The 1Hz sine wave signal used for the input is representative of typical control actions of the system. For comparison, a plot of measured velocity for a similar 1Hz sine wave demanded torque using torque control is shown in Figure 39. The expected shape of the velocity for a sine wave torque input is a sine wave of equal frequency with scaled magnitude, which is clearly not achieved. A major cause of the poor performance is the presence of backlash in the motor gearbox, visible as sharp troughs directly following sharp changes in velocity, which is a drawback of many high-ratio planetary gearboxes. As the motor changes direction, the inertia spins freely as the gears swap contact points. During this

time, the motor rapidly accelerates as the load is greatly reduced, causing an impact and subsequent over acceleration when the gears mesh. This in turn causes the inertia to overtake the motor and causes the erratic behaviour shown below.

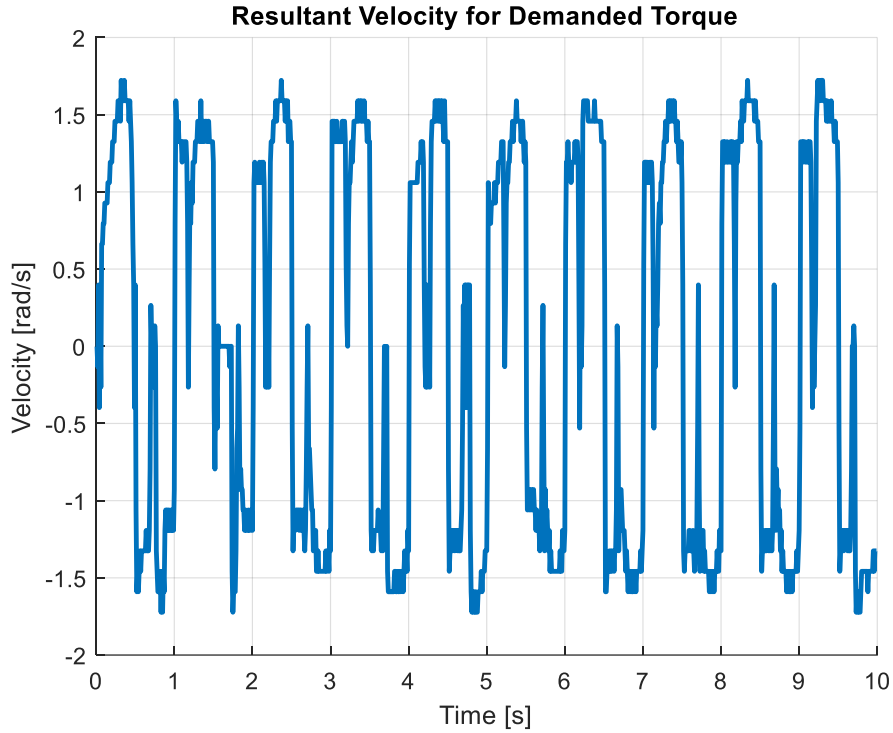


Figure 39. Resultant velocity for demanded torque, showing erratic behaviour of system under torque control schemes.

The velocity-based control allocation offers two main advantages over the torque-based control. Firstly, there is no requirement for system identification, modelling, or parameter estimation. The system identification process is extremely time consuming, with many stages to the process, some with considerable computation time. Additionally, a significant level of research was required to determine the source of non-linear behaviour, electrical or mechanical, and to determine a suitable augmentation of the model to accurately capture this behaviour. Even when the model is determined and parameters estimated, there is still an assumption of accuracy for the current measurement equipment, which is unlikely to be valid. Secondly, the performance of the PI method is far superior. Despite being a rather rudimentary method, the PI control of velocity produces extremely useful results without needing any knowledge of the system and is far more readily implemented with only minimal tuning required.

6.3 Implementation of Computer Vision Video Stream

In section 4.2.1 the process of determining an angle displacement between the hand and object using computer vision process for a single image is outlined. For implementation, this process must be extended to a video stream in near real-time.

6.3.1 Microsoft Webcam

Initially, the camera used to capture the video stream is a Microsoft Life-Cam VX3000 shown in Figure 40, allegedly capable of capturing 30 frames per second (fps) at a resolution of 640x480 pixels. In modern terms, this would be considered a low-quality image, but is perfectly suitable for the application considering the lack of detail required.



Figure 40. Microsoft Life-Cam VX3000 [18].

There are several methods for capturing images from a video stream in MATLAB. The most rudimentary is to create a webcam object, initiate a video stream and retrieve images as they become available. Using this method, the camera is able to achieve a framerate of 10fps, with a 400ms delay between an event occurring and a frame showing that event being available. By adjusting the exposure time to be as low as possible and increasing gamma and brightness, this is improved to 30fps and a 130ms delay. Modern webcams and cameras attempt to calculate the ideal settings based on previous frames during the image post-processing sequence, which consumes a small amount of time. By manually specifying these settings, the computation time is reduced, and frames are processed at a higher frequency.

Achieving a frame rate of 30fps with a delay of 130ms indicates some form of buffer is implemented in the private MATLAB functions used for image capture, or at a software level by the webcam driver. To achieve a frame rate of 30fps, the maximum exposure time of the camera is 33ms. Without any form of threaded image processing and retrieval, it is assumed that the longest period of time required for an image to be available is two times the exposure time. The first period is used to capture the image, and the second is used to post-process the image. As the delay is four times larger than the exposure time, the images are likely processed and stored in a memory buffer prior to being made

available. As the software drivers and image acquisition functions are provided by external parties, it is impossible to determine the cause of this delay.

To account for this delay, the Kalman filter can be run at a frequency of 8Hz, causing the timestep of the predictions to align with the delay period. At this frequency, the system is unstable, making this camera setup unsuitable without significant increase in operating frequency.

MATLAB is an interpreted language which uses just-in-time compilation of code. Interpreted languages are often significantly slower than compiled languages, such as C, but offer improved readability and debugging functionality. Compiled languages such as C can be used in MATLAB through a process called 'MEX', for MATLAB Executable. MEX compilation is the process of compiling C code for execution via MATLAB, taking advantage of significant speed increases. [19] saw an increase in speed of 85% when using MEX files as opposed to MATLAB code. It is assumed that a using a MEX file for image acquisition will increase the speed of capture.

OpenCV is an open source computer vision library, available in many languages, one of which is C code ready for MEX compilation in MATLAB. Additionally, as OpenCV is open source it is clear how the functions operate and whether a memory buffer is being used. The standard image capture functions do not appear to implement a memory buffer, and simply returns the most recently captured and processed image possible.

Using the MEX OpenCV library, image capture is slightly improved. The frame rate remains at 30fps which is hardware restricted, but the delay reduces to 90ms. Unfortunately, this improvement is not significant enough to overcome to reach a stable control frequency.

It is important to note that a controller frequency that exhibits stability in simulation is theoretical reference for the minimum allowable control rate, and so far only image capture frequencies have been discussed. In order to achieve stability in implementation the time to complete the computer vision process must be accounted for, and any non-ideal behaviour in sensors and actuators must be accounted for. It is far more likely that an image acquisition frequency two to three times greater than the minimum controller frequency, and a frame availability delay equivalently shorter, must be achieved in order to realise stability in the experiment. In pursuit of this, a new camera setup is sourced.

6.3.2 Raspberry Pi Camera

A Raspberry Pi (RPi) is a small single-board computer capable of using many peripherals, such as cameras. The camera board made for the RPi, shown in Figure 41, features a high-speed shutter and hardware drivers offer a large amount of control over exposure settings. Without modification of any software or hardware, the sensor is capable of 640x480 video at 90fps, provided good lighting conditions. Additionally, despite the native resolution being 2592x1944 pixels, the reduced resolution utilises ‘binning’ as opposed to cropping, allowing full field of view at low resolutions.

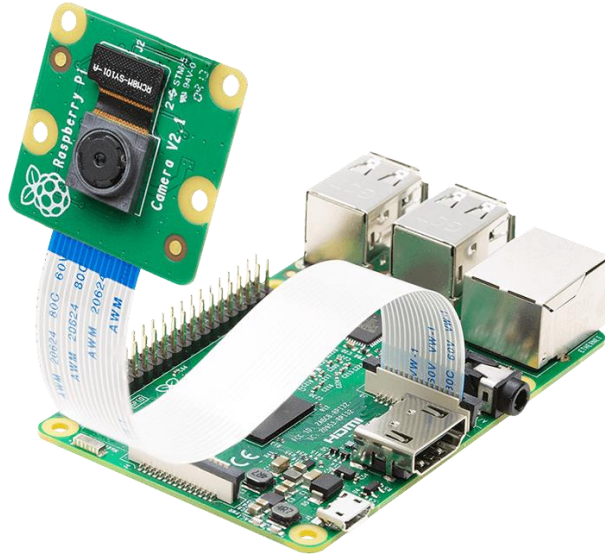


Figure 41. Raspberry Pi Camera Board [20].

The RPi zero-w is the smallest of the RPi series but offers a significant advantage when utilising the board for image acquisition: the use of USB on-the-go (OTG). USB OTG enables the RPi and connected camera board to present and connect to a computer as a USB webcam. Without this capability, connection must be made using ethernet with the camera function as an internet protocol (IP) camera.

The RPi camera is set up using ‘uvc-gadget’, an open source enhancement to the standard UVC driver for Linux. Using this enhancement, the video stream resolution and frame rate are able to be set, and the camera is set up as a UVC gadget with a direct pipe between processed frames and the USB port. Direct connection between the camera and USB port is ideal in reducing frame availability latency. Unfortunately, despite the increased framerate capability of the sensor, the achieved frame rate is only 30fps, with an average image acquisition delay of 80ms.

As the source of the input latency is assumed to be the use of memory buffers in private toolbox functions, the image acquisition process is moved onto the RPi directly and executed using Python.

Using the OpenCV library, which can be built on the RPi from source files, a Python script can be written to determine achievable image processing rates. Using the OpenCV library and task threading, the image acquisition task and image processing task can be carried out simultaneously, ensuring there is minimal latency for images to be provided. Unfortunately, a drawback of the RPi zero is the significantly reduced computational power when compared to other models of RPi, causing the maximum framerate to be 40fps. As the initial test is conducted without any feature detection and the achieved frame rate is not significantly greater than the required rate, it is unlikely there is sufficient computational overhead to allow feature recognition to occur without a significant reduction in frequency. The desktop computer using a 3.2GHz 6-core CPU takes 2ms to undistort an image, leading the assumption that a significantly greater amount of time would be required on the single core 1GHz CPU of the RPi. While 2ms is not a long period of time, an optimistic increase to 5ms to undistort an image decreases the frame rate to 30fps, prior to any feature recognition.

Further testing confirmed the ability of the sensor to capture video at 90fps, with frame timestamps occurring 11ms apart. This test was run locally on the RPi unit, without any requirement to stream or pipe data to the desktop computer. The results of this test indicate that despite the capability of the sensor and the widespread support of the RPi system, this camera configuration is not the ideal choice due to the additional layer of complexity in obtaining images from the sensor to be used in the image acquisition stream.

6.3.3 Measuring Image Acquisition Latency

Throughout section 6.3 the image acquisition latency of each camera is referred to, and measurements of latency have been provided. Methods for measuring image acquisition latency is an area of ongoing research at this time [21]. Without specialised equipment or purpose-built test setups, it can be extremely difficult to reliably quantify the amount of latency due to the camera in isolation of other effects. The method used in this report is based around measuring the latency in terms of number of frames, as opposed to other methods which measure the latency in periods of time, typically milliseconds.

To measure the latency of each camera, a test is performed in which a script is run in MATLAB with the camera pointed at the monitor, capturing the output. The script is designed which follows the form of Algorithm 4.

Algorithm 4 – Image Acquisition Latency Determination

for $k = 1, 2 \dots 10$

- Store current UNIX time
- Print k
- Capture image of screen and store

End

The rate of execution of this process in MATLAB appears to be limited to 30Hz regardless of the capabilities of the image acquisition sensor in use. Additionally, there are errors introduced in two main areas.

Firstly, the monitor displaying the current frame number has its own inbuilt latency, both in the refresh rate of the monitor and the response time. The monitor used has a refresh rate of 60Hz which introduces a worst-case additional latency of $1/60$ or 16.67ms. This value varies depending on where the output is display on the monitor physically, with the refresh action not being simultaneous for the entire display areas [21]. The response time is 5ms, giving a total possible additional input latency of approximately 22ms. This is a significant amount of time which is unable to be accurately measured in this test setup.

Secondly, the process of printing a value to the screen during script operation is well known to be a time-consuming operation. The amount of latency introduced due to this operation is currently unknown.

Despite these known sources of error, the test is run in an attempt to quantify the latency of each camera. By inspecting the saved images and determine the first frame to show the counter variable, k , the input latency in terms of number of frames can be determined. The latency time is then calculated using the stored UNIX time at each frame. Figure 42 below shows the result of one test, with the first three frames not showing any output from the script, despite the frame capture being requested after the output was to be displayed. In this instance the input latency would be 4 frames, or 120ms, as the output is the script is first visible in the fourth frame.

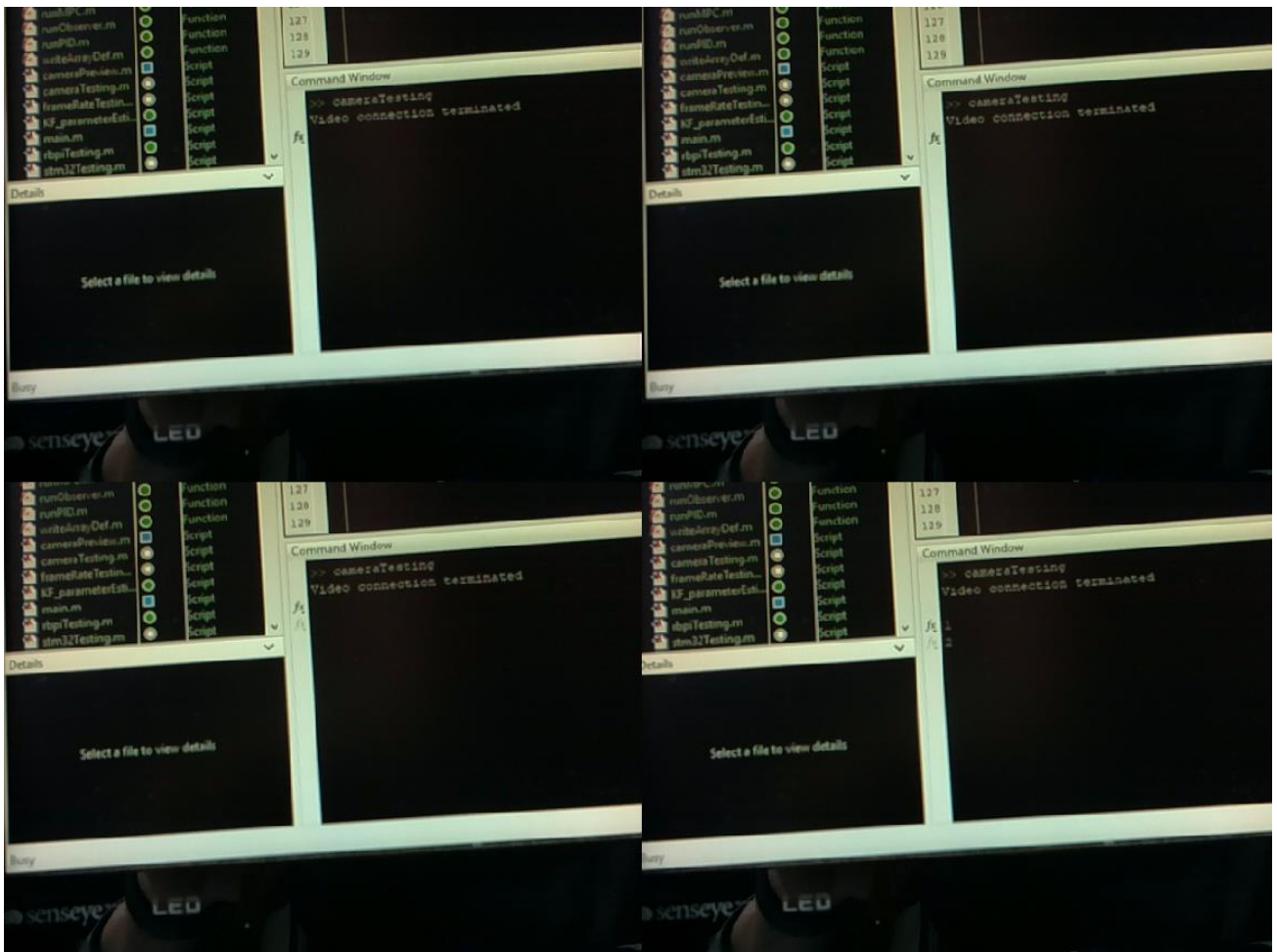


Figure 42. Image acquisition latency test. Frame 1 (top left), frame 2 top right), frame 3 (bottom left) and frame 4 (bottom right).

7 Results

Despite the control frequency being limited to frequencies that begin to exhibit instability in simulation, the experiment is conducted. After inclusion of the computer vision process to obtain position measurements of the object, the control and measurement frequencies are limited to 10Hz to ensure the prediction timestep aligns with the image acquisition delay. A simulation of the system with these parameters shows behaviour consistent with the boundary of stability, as shown in Figure 43. The object angle does stabilise to the balancing position, but undergoes significant overshoot from each control action, and takes an extremely long time to reach stability. Typically, systems are not designed to operate so close to the limit of simulated stability, as the accumulation of non-ideal behaviour of the components in physical systems causes instability in implementation.

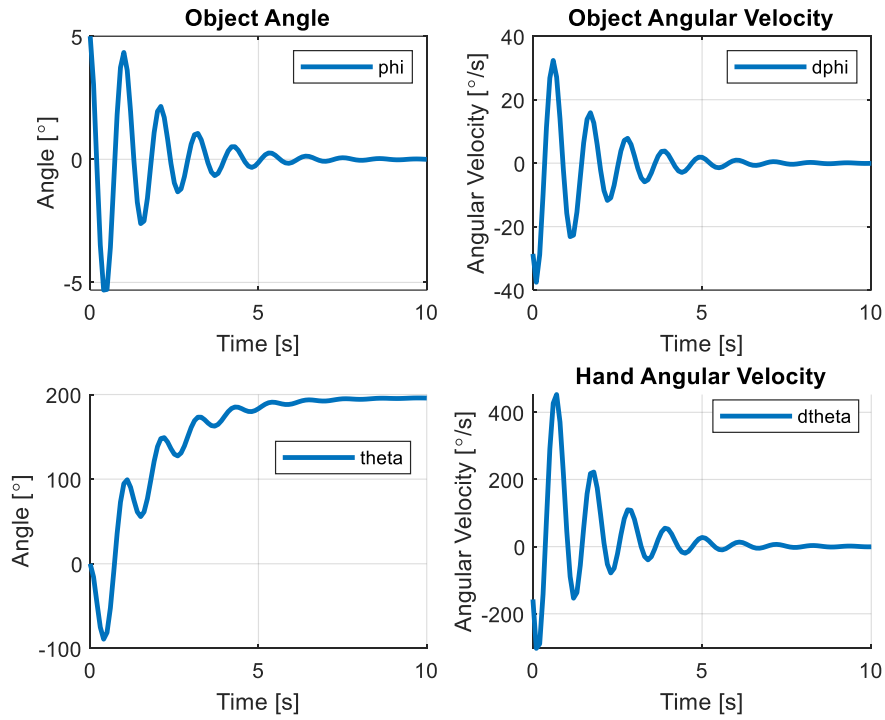


Figure 43. Results of simulation at 10Hz, showing signs of the limit of stability.

The results of implementation are shown in Figure 44. The system is clearly unstable, with the object oscillating between the physical bumpers at ± 9 degrees. This is a result of the low control and measurement frequency, as the object has often passed beyond the balancing point between two control actions. This behaviour is not unexpected, as the implemented control frequency is very close to the theoretical lower limit.

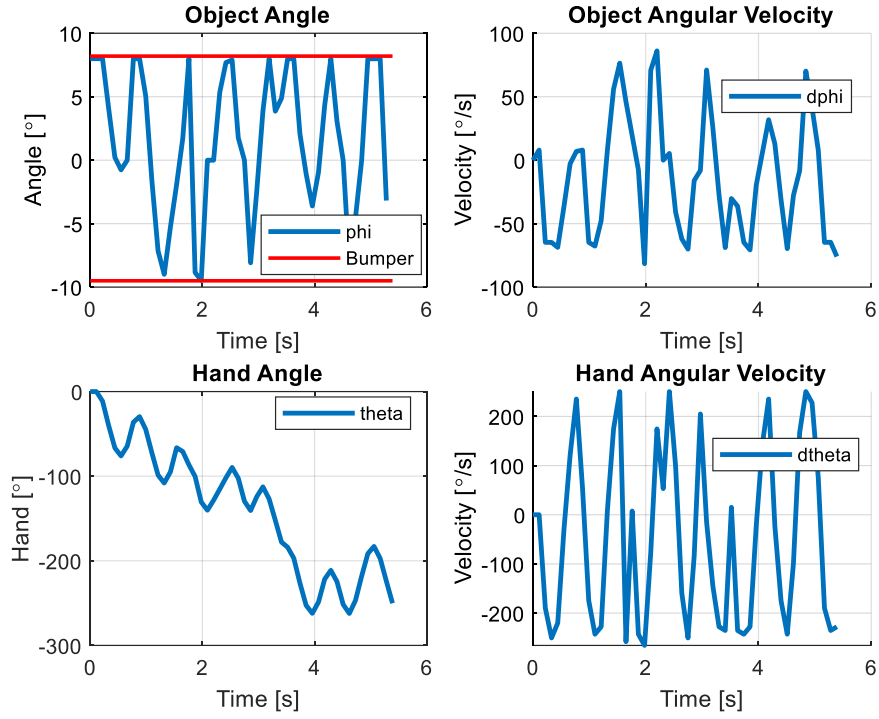


Figure 44. Results of experiment at 10Hz control and measurement frequency.

For comparison, Figure 45 and Figure 46 show the results of a simulation at 9Hz and 20Hz respectively. Despite only a 10% decrease in control frequency compared to the 10Hz simulation, the system has become unstable despite a relatively small initial deviation from the balancing position.

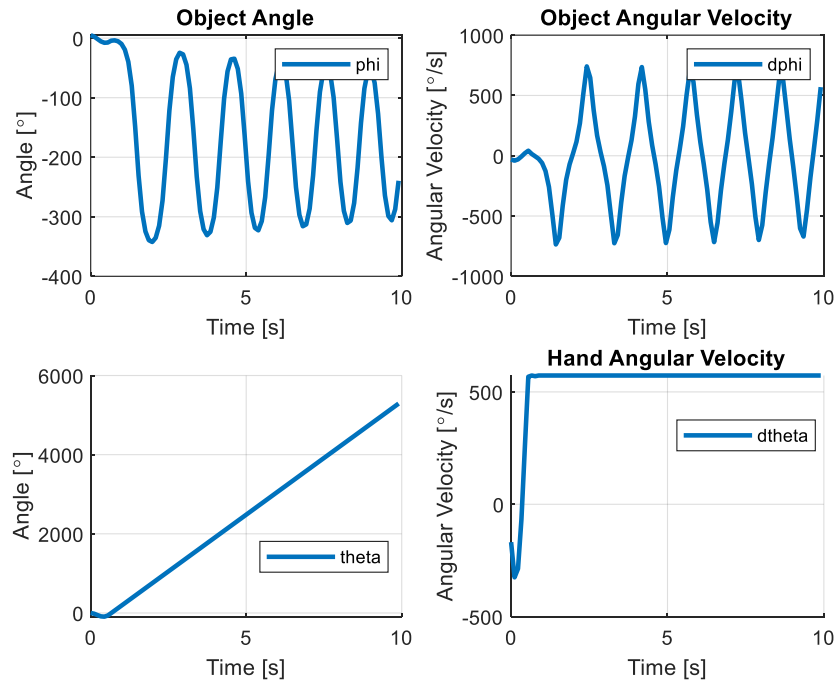


Figure 45. Results of simulation at 9Hz, showing instability.

At 20Hz, the system exhibits far stronger stability. This frequency is theoretically achievable with a camera framerate of 30fps but has not been achieved in implementation.

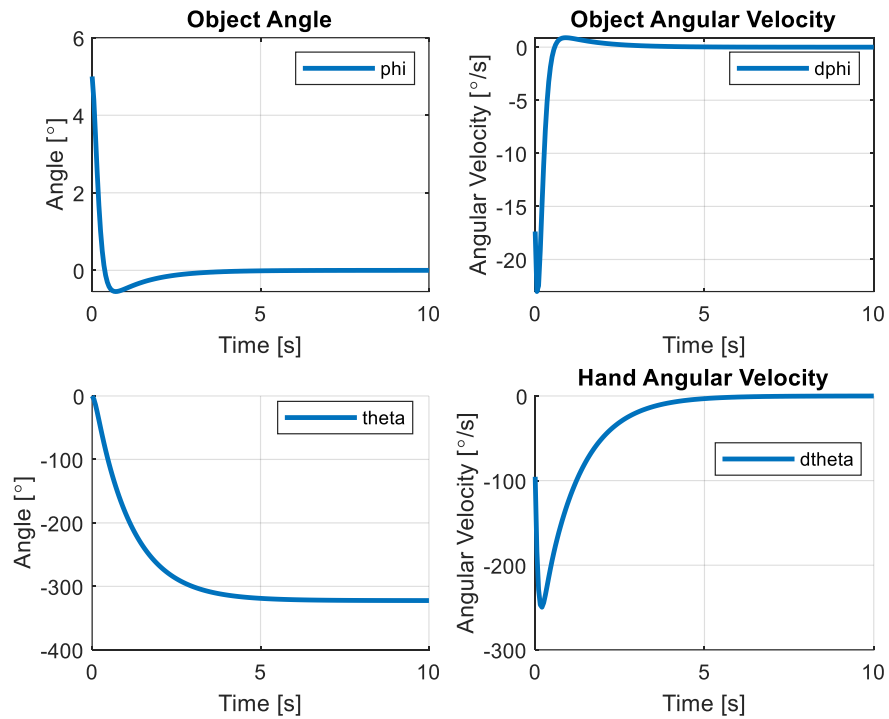


Figure 46. Results of simulation at 20Hz.

8 Conclusion and Recommendations

This report has developed two models of the disk-on-disk system, suitable for use with a Linear-Quadratic Regulator or Model Predictive Control alongside state estimation using the Kalman Filter. The Lagrangian model is designed for a torque input to the plant, whilst the Hamiltonian model is designed for a velocity input. Both models presented promising results in simulation.

A process for feature recognition in video streams was implemented, with promising results for extension to near real-time applications.

The design and construction of the experiment apparatus was successful and suitable for the application. No considerable shortcomings in the design process were highlighted during testing of the equipment.

The implementation process determined that velocity control achieved far superior performance compared to torque control when tested against reference inputs. This finding is particularly significant, as the method implemented negated the requirement for any modelling of the motor and subsequent parameter estimation using system identification to be performed. The poor performance of torque control is highly likely due to the use of a high ratio gearbox, with significant friction effects becoming apparent during system identification.

Implementation also revealed a significant and unavoidable image acquisition latency which rendered achieving stability impossible due to the achievable control and measurement frequency falling outside the range of stable frequencies determined in simulation. Image acquisition latency issues are widespread in communities using consumer level image capturing devices, with many successful applications negating the issue by using hardware capable of capturing image streams at rates several times higher than the devices used in this report.

The recommendation for completion of this project are:

- Obtain a high frequency camera for image acquisition
- Subsequently increase measurement and control frequency to be significantly faster than the minimum stable frequency of approximately 10Hz

9 References

- [1] R. Online, "The Emergence of Smart, Collaborative Robot Grippers," vol. 594x594, C. R. G. The Emergence of Smart, Ed., ed, 2018, p. A robotic gripper.
- [2] D. Serra, *Robot Control for Nonprehensile Dynamic Manipulation Tasks*. 2016.
- [3] A. Donaire, F. Ruggiero, L. R. Buonocore, V. Lippiello, and B. Siciliano, "Passivity-Based Control for a Rolling-Balancing System: The Nonprehensile Disk-on-Disk," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 6, pp. 2135-2142, 2017, doi: 10.1109/TCST.2016.2637719.
- [4] J. Ryu, F. Ruggiero, and K. M. Lynch, "Control of Nonprehensile Rolling Manipulation: Balancing a Disk on a Disk," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1152-1161, 2013, doi: 10.1109/TRO.2013.2262775.
- [5] A. Fairclough, "Lab 2: Euler-Lagrange modelling," in *MCHA3500 Lab Notes*, ed: The University of Newcastle, 2019.
- [6] T. Perez, *Engineering System Dynamics (Modelling, Analysis, and Simulation)*, T. Perez, ed.: The University of Newcastle, AUSTRALIA, 2013.
- [7] P. R. Tedrake, "Fully Actuated vs. Underactuated Systems," *Underactuated Robotics*: Massachusetts Institute of Technology: MIT OpenCourseWare, 2009, pp. 1-8. [Online]. Available: <https://ocw.mit.edu/>
- [8] C. Renton, "MCHA3500 – Review of LQG," *Mechatronics Design 1 Course Notes*: The University of Newcastle, Australia, 2019.
- [9] A. Wills, *Course Notes: Advanced Estimation*, The University of Newcastle, Australia., 2020.
- [10] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital control of dynamic systems*. Addison-wesley Menlo Park, CA, 1998.
- [11] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence, Pattern Analysis and Machine Intelligence, IEEE Transactions on, IEEE Trans. Pattern Anal. Mach. Intell.*, Periodical vol. 22, no. 11, pp. 1330-1334, 11/01/ 2000, doi: 10.1109/34.888718.

- [12] J. Heikkila and O. Silven, "A four-step camera calibration procedure with implicit image correction," ed, 1997, pp. 1106-1112.
- [13] MathWorks. "estimateCameraParameters." MathWorks. (accessed 26/05, 2020).
- [14] A. Smith, *Color Gamut Transform Pairs*. 1978, pp. 12-19.
- [15] T. Fitzermann. "DC motor control simulation." (accessed 09/10, 2020).
- [16] C. Makkar, W. E. Dixon, W. Sawyer, and G. Hu, "A new continuously differentiable friction model for control systems design," *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics.*, pp. 600-605, 2005.
- [17] Q. C. Jing Na, Xuemei Ren, *Adaptive Identification and Control of Uncertain Systems with Non-Smooth Dynamics*. Marr Conner, 2018, p. 314.
- [18] "Microsoft LifeCam VX-3000," ed, 2014.
- [19] S. L. M. A. Tanner, "Image Processing for Multiple-Target Tracking on a Graphics Processing Unit," Degree of Master of Science in Computer Engineering, Department of Electrical and Computer Engineering, AIR FORCE INSTITUTE OF TECHNOLOGY, Wright-Patterson Air Force Base, Ohio, AIR FORCE INSTITUTE OF TECHNOLOGY, Wright-Patterson Air Force Base, Ohio, 2009.
- [20] ArduCam, "Raspberry Pi Camera," ed: ArduCam.
- [21] S. Ubik and J. Pospíšilík, "Video Camera Latency Analysis and Measurement," *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 1-1, 2020, doi: 10.1109/TCSVT.2020.2978057.

Appendix A – Time Log Graphic

Total time spent: 457 hours.

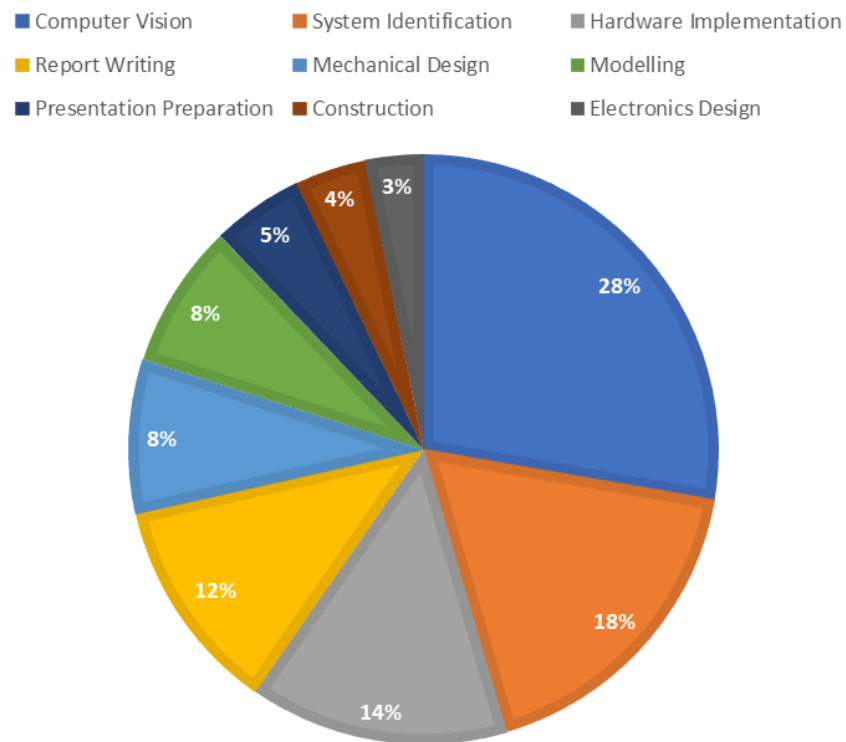


Figure 47. Breakdown of time spent.

Appendix B – Code, Drawings and Documents Generated

All code, drawings, documents, images and items generated or used in this project can be found at the GitHub page https://github.com/TimDunn-tds/FYP_DoD.