|  |  |
| --- | --- |
| Design Document & Report for MCHA3000's Project | University of Newcastle  University Drive  Callaghan NSW 2308  Australia |

|  |  |
| --- | --- |
| **Report Number:** |  |
| **Revision:** | . 1 |

|  |  |
| --- | --- |
| **Course** | MCHA3000 |
| **Project:** | Project |
| **Robot Type:** | Type I |

|  |  |  |  |
| --- | --- | --- | --- |
|  | *Name* | *Student ID* | *Contact* |
| **Prepared by:** | Christopher Caelli | C3206246 | C3206246@UON.EDU.AU |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *Name* | *Role* | *Signature* | *Date* |
| **Approved:** |  |  |  |  |
| **Authorised:** |  |  |  |  |

|  |  |
| --- | --- |
| **FILENAME:** | **DesignDoc\_MCHA3000** |

Change History

|  |  |
| --- | --- |
| Rev No. | Change Details |
| 1.0 | Initial Issue |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Executive Summary

* Defines the intention of the report.
* Places the report in context so the reader knows why it is important to read it.
* Why was the reported work undertaken?
* Why is it important?
* What problem is addressed?
* Briefly states the results.
* Briefly presents the implications and recommendations.

Table of Contents

[1. Introduction 5](#_Toc498347925)

[1.1. Identification & Objective 5](#_Toc498347926)

[1.2. System Overview 5](#_Toc498347927)

[2. Applicable Documents 6](#_Toc498347928)

[2.1. Project Documents 6](#_Toc498347929)

[2.2. Other Documents 6](#_Toc498347930)

[3. Abbreviations 6](#_Toc498347931)

[4. Abbreviations 6](#_Toc498347932)

[5. Project Work 7](#_Toc498347933)

[5.1. Prelude: Model Based Design 7](#_Toc498347934)

[ Show how the mathematical model was derived and used, with the design constraints, to design your mechanical components, choose sensors and actuators and design the controller. 7](#_Toc498347935)

[ Mathematical models and discussion (backed by results) about validity and limitations 7](#_Toc498347936)

[ Divide and conquer – how the problem was broken down into smaller sub-problems and how they were solved. For example, separation of controller and control allocation 7](#_Toc498347937)

[5.2. Hardware (0-100%) 8](#_Toc498347938)

[5.3. Actuator 10](#_Toc498347939)

[5.4. Sensors 11](#_Toc498347940)

[5.5. Power Electronics & Regulation 11](#_Toc498347941)

[5.6. Software (0-100%) 12](#_Toc498347942)

[5.7. System Identification (0-100%) 13](#_Toc498347943)

[5.8. Simulation Model (0-100%) 15](#_Toc498347944)

[5.9. Control (0-100%) 16](#_Toc498347945)

[6. Conclusion 18](#_Toc498347946)

[Summarise the results 18](#_Toc498347947)

[reflect on the work presented 18](#_Toc498347948)

[make recommendations 18](#_Toc498347949)

[suggest future work or improvements. 18](#_Toc498347950)

[6.1. Reflect on the work done 18](#_Toc498347951)

[6.2. What would you do differently in terms of design if you could start again? 18](#_Toc498347952)

[7. Appendixes. 19](#_Toc498347953)

[7.1. MCHA3000 Requirement vs Design Matrix 20](#_Toc498347954)

[7.2. Project Inherited Requirements 21](#_Toc498347955)

[7.3. State Space Calculations 22](#_Toc498347956)

List of Figures

[Figure 1 Work-Breakdown-Structure 5](#_Toc498347957)

[Figure 2 State Space Model Implimented in Simulink® 7](#_Toc498347958)

[Figure 3 Breakdown of Type-I Design Decisions 8](#_Toc498347959)

[Figure 4 Creo Drawings of Type I Robot 9](#_Toc498347960)

[Figure 5 Final Revision of Type-I Robot 9](#_Toc498347961)

List of Tables

**No table of figures entries found.**

1. Introduction
   1. Identification & Objective

This report is written to document the design decisions, progress and achievements of the MCHA3000 Project. This document covers the all aspects of the work done throughout JUNE17-NOV17. It will focus on the detailed design of the Type-I cart robot named and analyse the results of the project.

* 1. System Overview

The MCHA3000 Project & prerequisite assessments seek to

“introduces the process of mechatronic system

design. It is a project-based course where a mechatronic system

for an electromechanical component is designed and built. The

course integrates tools and skills related to computer and

software, electronics, control, modelling and simulation. It also

develops the concepts of experimental modelling and

implementation of computer control systems. The course

provides a real-life experience related to the practice of

mechatronics engineering.”.

It achieves this through a personal project requiring the listed skills to achieve the end task: Produce a robot that can balance a pendulum. As such,



Figure 1 Work-Breakdown-Structure

[TODO: Add work breakdown structure here]

*[This paragraph shall briefly state the purpose of the system and the software to which the document applies. It shall describe the general nature of the system and software; summarise the history of system development, operation, and maintenance; identify the project sponsor, acquirer, user, developer, and support agencies; identify current and planned operating sites; list other relevant documents.]*

***V-Diagram here***

1. Applicable Documents

The following documents at the latest issue unless otherwise specified form a part of this document to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this document, the contents of this document shall be considered a superseding requirement.

* 1. Project Documents

| Project Identifier | Title | Rev. | Date |
| --- | --- | --- | --- |
| 001 | Report | 1.0 | 17/11/17 |
| 002 | Code | 1.0 | 17/11/17 |
| 003 | Robot | 1.0 | 17/11/17 |
| 004 | Simulink | 1.0 | 17/11/17 |
| 005 | Test Logs | 1.0 | 17/11/17 |

* 1. Other Documents

| Document Identifier | Title | Rev. | Date |
| --- | --- | --- | --- |
| 2503Q–AVR–02/11 | ATmega32/L Datasheet | Q | 02/2011 |
|  |  |  |  |

1. Abbreviations

|  |  |
| --- | --- |
| Abbreviation | Expansion |
| NEMA | National Electrical Manufacturers Association |
| TI | Texas Instruments |
| MCU | Micro Controller Unit |
|  |  |

1. Abbreviations

Overview of Required Work

The requirements of the project can be broken down into 3 sections: Report Requirements, Project Requirements, and Course Requirements. They can be found in Appendix 12.2 Project Inherited Requirements.

Critical

1. Project Work
   1. Prelude: Model Based Design
      1. Brief

To do the MCHA3000 Project, I utilised a Model-Based-Design using traditional state space modelling to produce a state space we could simulate. Creating our model first helped us explore any emerging properties created in the Type I Cart based robot.

The rationale was to use this simulation model to derive minimum component specifications, such as motor torque, cart & pendulum mass, required sensor fidelity etc.

* Show how the mathematical model was derived and used, with the design constraints, to design your mechanical components, choose sensors and actuators and design the controller.
* Mathematical models and discussion (backed by results) about validity and limitations
* Divide and conquer – how the problem was broken down into smaller sub-problems and how they were solved. For example, separation of controller and control allocation
  + 1. Approach

By defining our states as , we found the lagrangian, the hamiltonian and finally a linearised model about the operating point,

The final model is listed below, but please see *Appendix 12.3 State Space Calculations* for the full procedure.

Equation (1) is the State Space Model for the Type I cart.

This was implimented in MATLAB® Simulink® using integrator chain methodology from UoN prerequisite course MCHA2000.

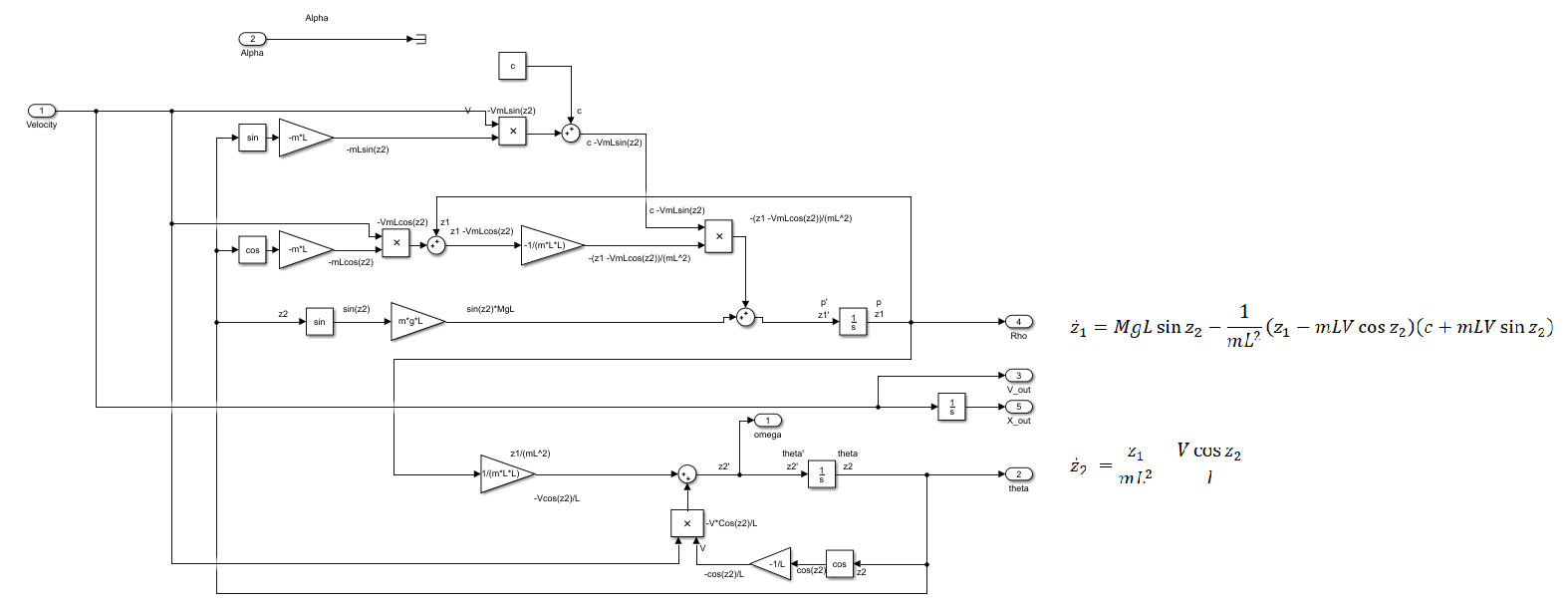


Figure 2 State Space Model Implimented in Simulink®

* + 1. Results – HIGHLIGHT THE SUBMODULES WE’RE INTERESTED IN



* 1. Hardware (0-100%)
     1. Chassis
        1. Brief

The rolling stock design evaluation was kept minimal due to highly supportive feedback from workshop staff of an Aluminium box design, with workshop provided 606ZZ mountings, 6mm Axle & Sensor couplings.

Figure 3 Breakdown of Type-I Design Decisions

* + - 1. Approach

A 2-day sprint was carried out over a Friday-Saturday timeframe to evaluate using a 10mm hollow aluminium tube, with a 200gram threaded rod insert to provide weight at the top. Self-review and evaluation of the prototype proved sufficient and was carried through to the final design.

Additionally, 90mm Polulu wheels were chosen pre-project on advice of previous successful students, and no evaluation was under taken for these components.

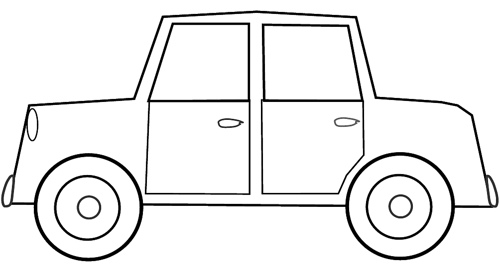


Figure 4 Creo Drawings of Type I Robot

* + - 1. Results

The Robot proved robust and remained cost effective. Total cost was:

|  |  |
| --- | --- |
| Item | Cost |
| Alum. | $0 |
| Rivets | $0 |
| Total: | $0 |

Testing the dynamics of the plant showed that even though there was a low weight of the pendulum, the dynamics were slow and reliable. This would aid our control design at a limited 100Hz control loop rate.

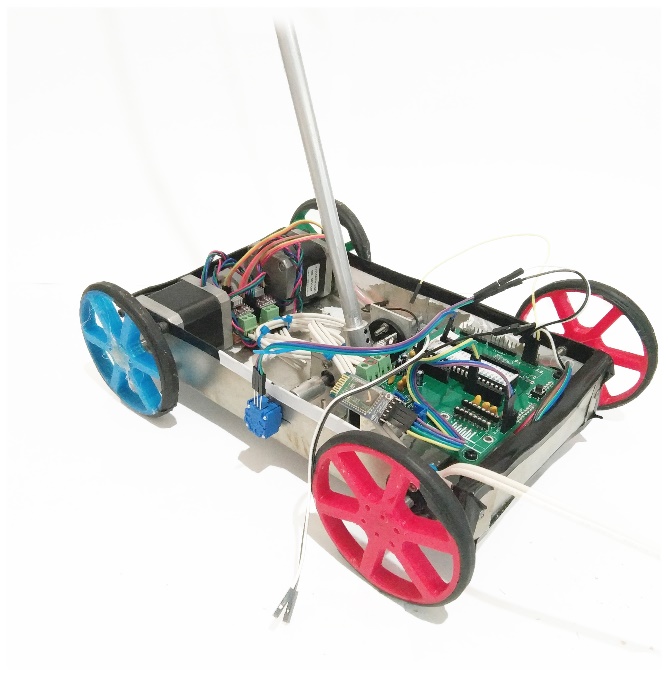


Figure 5 Final Revision of Type-I Robot

* 1. Actuator
     + 1. Brief
       2. Approach

A Pugh Matrix was undertaken considering 3 options that fit the Simulink Model’s criteria (Using Lab 0 Effort-based actuator & (1) Flow-based actuator).

Table 1 Pugh Matrix: DC Motors vs Steppers



Steppers we selected as the most suitable candidate due to previous student opinion of stepper motors being more reliable, as well as having suitable steppers available for use. This significantly reduced the cost and lead-time of the project.

* + - 1. Results

The specific steppers are the 17H185-04A, and were deemed adequate at providing the required velocity. Analysing the data available online about the 17H185-04A, the following calculations were made.

Table 2 Stepper Calculations



As our plant is able to be controlled with an input velocity saturation of 0.5m/s, the motors are suitable.

Figure of Ramping Velocity till failure/slippage

* 1. Sensors
     + 1. Brief

The Sensor choice was heavily biased by the early requirements of Labs 1 – 5, requiring use and consequently informal evaluation of the Sparkfun Rotary Encoder - Illuminated (RGB)[[1]](#footnote-1) & the BOURNS 91 Series Rotary Potentiometer[[2]](#footnote-2). Early access to these sensors, and previous student’s feedback about the MPU-series IMUs drove the evaluation of the Pugh Matrix.

* + - 1. Approach

A Pugh Matrix was used to evaluate sensor options

Table 3 Pugh Matrix: Sensor Suite



The Potentiometer’s low-setup costs and high-quality output came out on top. The IMU + Potentiometer has been highlighted as potential alternative, due to its higher fidelity output through data infusion, as well as being able to read the Type-I Cart’s inclination. Ultimately, the lead-time in setting up the IMU, along with data infusion proved too risky.

* + - 1. Results

Sensor Raw Data

* 1. Power Electronics & Regulation
     + 1. Brief

Power options for the Type-I Robot were limited to offboard/tethered & Battery. The robot was shown to use 1.5A @ 12V Peak, and 0.05A Average

[Matlab Figure of Current over time, vs input]

* + - 1. Approach

Both options were utilised throughout the prototyping phase, which allowed review of power regulation by the MC 7805CT under load, and the stepper current/vref under load.

* + - 1. Results

[Photo of PSU, Lead, Battery, Buck Converter]

[Matlab Figure of Load & Photo of PSU under load]

* 1. Software (0-100%)
     1. Maintainability, Code Organisation & Style
        1. Brief

The final code base was over 2,200 lines long, so a focus was on how to ensure the code was maintainable and readable, when features needed to be added or changed.

* + - 1. Approach

Three main best-practice paradigms were implemented to improve the maintainability of the code.

A tokenised command interrupter was implemented to ensure all commands were handled separately, and could be readily expanded.

|  |
| --- |
| /\* Command table code Adapted from: http://fundamental-code.com/ on 14/10/17, under MIT License use\*/ |
| *typedef* *struct* { |
| const *char* \*nameOfFunction; |
| *void* (\*func)(*char*\*); |
| const *char* \*helpText; |
| } commands\_t; |
| *int* MaxCommandLength = 20; |
| commands\_t commandTable[] = {{"", \_cmd\_empty, ""}, |
| {"help", \_cmd\_help, "Gives all commands"}, |
| ... |
| {"get", \_cmd\_get, "get [theta|pho]"}, |
| {"log", \_cmd\_log, "log [enc] <samples>"}}; |
| *void* cmd\_parse(const *char* \* *cmd*) |
| { |
| if (cmd == NULL)  ... |
| else |
| { |
| *uint8\_t* lengthOfCommandTable = sizeof commandTable / sizeof commandTable[0]; |
| for (*uint8\_t* i = 0; i < lengthOfCommandTable; i++) |
| { |
| //Check where the space is, so we can have different length commands |
| *int* spaceFound = MaxCommandLength; |
| if (strchr(cmd,' ')) |
|  |
| ... |
| if(!strncmp(cmd,commandTable[i].nameOfFunction,spaceFound)) |
| { |
| if(strcpy(&arg, cmd+spaceFound+1)) |
| { |
| printf\_P(*PSTR*("Arg: %s\n"), arg); |
| commandTable[i].func(arg); |
| } |
| ... |
| } |

Figure 6 Simplified Code Snippet for Serial-Interrupter

Encapsulation was used in parallel with a N2 Diagram to maintain a natural structure to code segments. This improved code organisation as well as maintenance

Table 4 Software Encapsulation



The Static Analysis Tool cppcheck in combination with a Sublime linter was used to enforce a code structure style, to ensure a high maintainability in the future. This tool also ensured a consistent code formatting/naming schema.

* + - 1. Results

NEEDED

* + 1. Testing
       1. Brief

Embedded program proved a difficult platform for unit tests, but was critical to ensuring each model was performing correctly and robustly.

* + - 1. Approach

A Test-Driven approach for software development was selected. The Unity framework was chosen to compile our code in gcc to run a combination of unit tests for each module. This involved mocking certain avrgcc, avr functionality as well as standard assertions.

|  |
| --- |
| #include "unity\_fixture.h" |
| #include <stdint.h> |
| #include "avr/io.h" |
| #include "avr/mock\_sfr.h" |
| #include "stepper.h" |
|  |
| TEST\_GROUP(Stepper); |
|  |
| TEST\_SETUP(Stepper) |
| { |
| stepper\_init(); |
| } |
|  |
| ... |
| TEST(Stepper, setDir) //Tests if our Direction method is working |
| { |
| PIND = 0x00; |
| mock\_portd = 0x00; |
| set\_dir("backwards"); |
| TEST\_ASSERT\_BITS\_HIGH\_MESSAGE(0x20, mock\_portd, "Expected 00100000"); |
| set\_dir("forwards"); |
| TEST\_ASSERT\_BITS\_HIGH\_MESSAGE(0x04, mock\_portd, "Expected 00000100"); |
| set\_dir("left"); |
| TEST\_ASSERT\_BITS\_HIGH\_MESSAGE(0x00, mock\_portd, "Expected 00000000"); |
| set\_dir("right"); |
| TEST\_ASSERT\_BITS\_HIGH\_MESSAGE(0x24, mock\_portd,"Expected 00100100"); |
| }  ... |
|  |
| TEST\_GROUP\_RUNNER(Stepper) |
| { |
| RUN\_TEST\_CASE(Stepper, setDir); |
| } |

Figure 7 Simplified Code Snippet for Stepper setDir Test

* + - 1. Results

Unity can then run this test, and report whether it was successful or not.

|  |
| --- |
| $ cat Unity\_20171113\_234843\_Log.txt  Unity test run 1 of 1  TEST(Encoder, DefaultCount) PASS  ...  TEST(Startup, Test2) PASS  TEST(Stepper, setDir) PASS  -----------------------  8 Tests 0 Failures 0 Ignored  OK  Completed on 20171113\_234843 |

* + 1. Documentation
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Safety Features
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* 1. System Identification (0-100%)
     1. Log Data to MCU
        1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Use Data to guide Model Structure
       1. Pendulum Friction
          1. Characterisation:

A combination of items effected the Pendulum Friction.

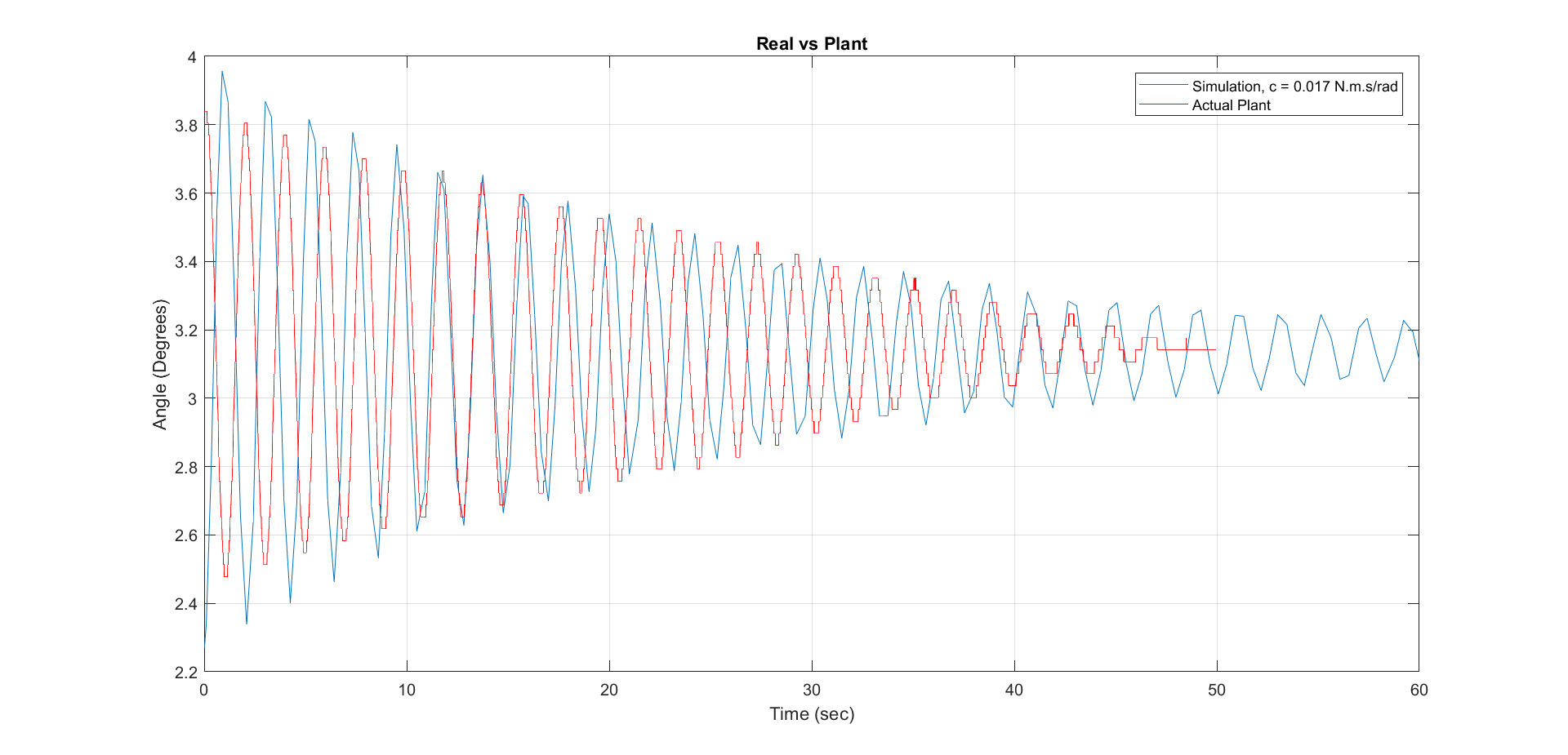
The 606-ZZ bearings, L-Bracket alignment, M6 Rod surface finish & tolerance, and potentiometer/encoder friction all contribute to the total friction moment occurring at the rotational axis of the pendulum.

* + - * 1. Experiment Design:

I decided to implement a single friction coefficient, and find a plant value experimentally. I logged real plant data and plotted this against an unpowered plant simulation. Using a binary search method, I was able to identify and estimate this value. A final value of 0.017 N.m.s/rads was chosen as best fit.

* + - * 1. Model calibration:

By experimentally fitting a value of friction from real plant data, crude calibration was already achieved. The friction was sufficiently small to not warrant looking at the specific datasheet for the supplied bearings, and no further action was taken.



* + - 1. Stepper Motors
         1. Characterisation:

The principle used to model and control the stepper motors was to find an operating envelope where the stepper motors did not slip, and add saturation limits to my input velocity, to ensure the steppers would not slip. As a result, the steppers are modelled as pure flow actuators, delivering the demanded velocity with no losses.

* + - * 1. Experiment Design:

The Data below helped me decide of a maximum of +-0.5Meters a second.  
Configuration:  
2\* 17H185H-04A

2 \* drv8825 (Vref @ 1.2V (Texas Instruments, 2017), ¼ microstepping)

2\* Pololu Wheel 80×10mm



**running torque: unknown**

**Holding torque of** 43.8 N·cm

* + - * 1. Model calibration:

The data was taken from the robot with full weight, and no input disturbances. Whist the final design was unable to balance, stress-testing during balancing would help further calibrate these saturation limits.

Testing with larger input disturbances would help identify the saturation points required. Further calibration could be achieved by characterising a maximum acceleration slew rate, which could help us further expand the stepper’s operating envelope.

* + - 1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Fit Actuator Model to Recorded Data
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Validate new Models
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Automation between MATLAB and MCU
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* 1. Simulation Model (0-100%)
     1. Sensor Sampling and Quantisation
        1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Actuator Model and Control Allocation
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Actuator Limits
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Model Effect on Inclined Ground Plane and Transition
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* 1. Control (0-100%)
     1. Model-Based Control Design
        1. Brief

Knowledge from ELEC4400, consultation with students undertaking ELEC4410, and tutor assistance helped generate the capability required design the controller.



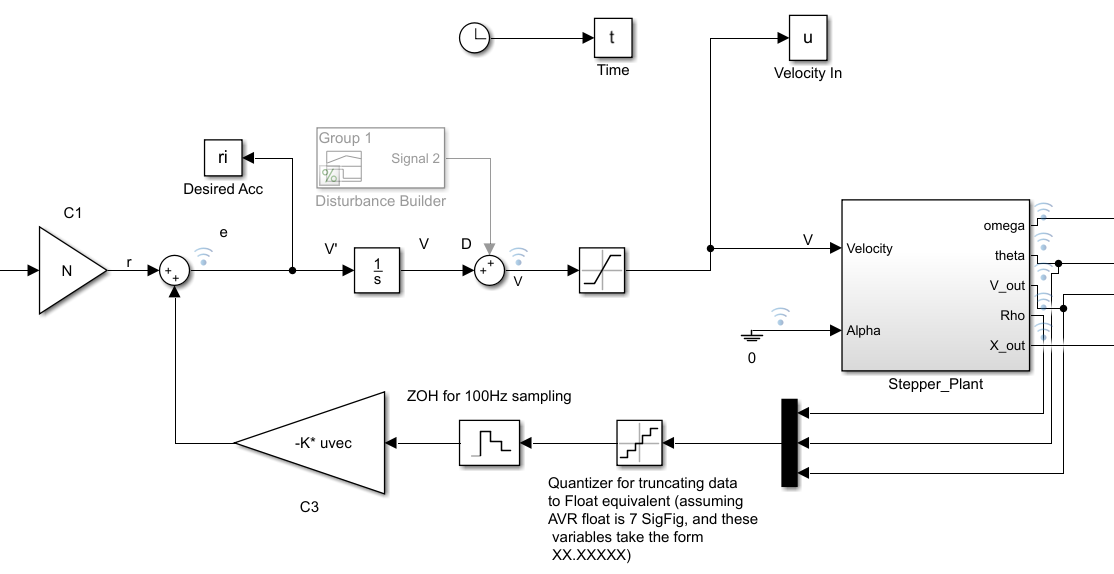
Three (3) options were considered: State Feedback, State Feedback with integral action, and a PI controller. As with Sensors, it was decided that the State Feedback option was best, with the capacity to extend with Integral feedback before project delivery.

This was achieved by taking equation (1), and linearising it about the operating point,

.

Equations (2.1) & (2.2) show the State-space and output respectively. This was then implimented in Simulink, and verified against the Plant model derived in section 6.2.

The following Matrices were used to impliment the controller in simulink.

Lastly, the model had to have a Zero-Order Hold to replicate the 100Hz sampling that would occur on the real controller, and a quantizer to emulate the floating-point resolution.

* + - 1. Results
    1. Disturbance Rejection
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Actuator Limits
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. Plant Uncertainty & Robustness
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A MonteCalo simulation was used

* + 1. Controller Dynamics at 100Hz
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

* + 1. HIL Test
       1. Brief

A

* + - 1. Approach

A

* + - 1. Results

A

1. Conclusion

Summarise the results

reflect on the work presented

make recommendations

suggest future work or improvements.

* 1. Reflect on the work done
  2. What would you do differently in terms of design if you could start again?

1. Appendixes.
   1. MCHA3000 Requirement vs Design Matrix



* 1. Project Inherited Requirements
     1. Project Report (Assessment 7)[[3]](#footnote-3)

**Type** Report

**Description** The project report meets the course objectives of knowledge acquisition and demonstrates assimilation of data, upon reflection and analysis, to produce articulate and document the design conducted in the project, which conveys evidence-based understanding of the concepts and topics.

* Abstract indicates objective, scope and results (10%)
* Report organisation (25%)
* Clarity of writing (20%)
* Clarity of figures and tables (15%)
* Analysis and discussion of results (30%)
  + 1. Project Demonstration Assessment (Assessment 6)[[4]](#footnote-4)

**Type**Project

**Description**The purpose of demonstration is to assess whether the project meets the required goals and specifications, and give students the opportunity to discuss orally the degree to which they attained the required goals

Project

Hardware (0-100%)

* Robust construction
* Cost effectiveness
* Plant dynamics suitable for control loop rate
* Suitable power electronics for selected actuators
* Adequate power regulation under load

Software (0-100%)

* Maintainability (e.g., serial interpreter uses command tables rather than if-else chains)
* Code organisation (e.g., encapsulation, orthogonal functionality separated into C modules (.c/.h), use data interfaces instead of global variables)
* Testing (e.g., coverage of unit tests, design for testing, TDD)
* Style (e.g., consistent indenting, function/variable naming convention)
* Documentation (e.g., appropriate use of code comments, online help via serial interface)
* Safety features (e.g., watchdog, comms keepalive/timeout, unrecoverable actuator limits)

System identification

* Log experimental data using MCU
* Use data to guide model structure selection
* Fit actuator/plant models from gathered data (training set)
* Validate model fits against new data (validation set)
* Automation between MATLAB and MCU

Simulation model

* Sensor sampling and quantisation (e.g., encoder ISR, ADC)
* Actuator model and control allocation
* Actuator limits (e.g., saturation and slew rate)
* Model effect of inclined ground plane and transition

Control

* Model-based control design
* Robustness to disturbances (e.g., integral action and/or disturbance estimation + feedforward cancellation)
* Design consideration for actuator limits (e.g., anti-windup scheme for integral states, closed-loop bandwidth)
* Account for uncertainty in plant structure/parameters and validate robustness via MonteCarlo simulation
* Appropriate controller dynamics for implementation at control loop rate
* PIL/HIL testing of control system (controller + control allocation) implementation
  + 1. Course Outline

On successful completion of this course, students will be able to:

1. Formulate specifications for adopting/designing different components of a mechatronic system (mechanical, electrical, sensors, actuators).

2. Conduct a mechatronic design using a structured formal approach. Make decisions about component choice taking into account its effects on the choice of other components and the performance of a mechatronic system.

3. Design and implement software for a computer control system with sensor and actuator interfaces.

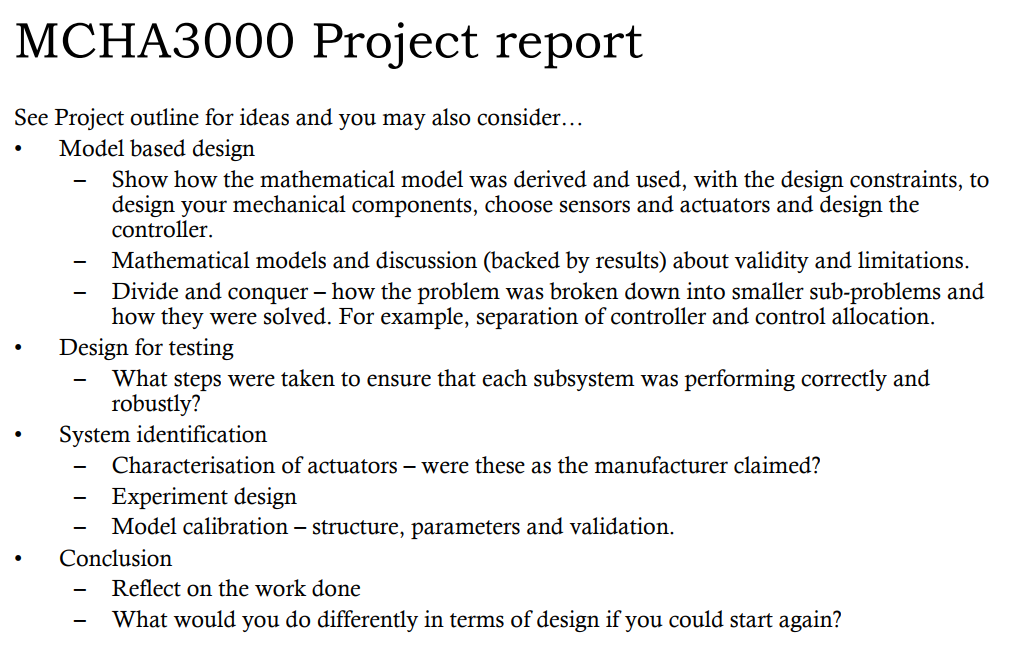
4. Design and implement communication interface with a computer control system for tuning.

5. Design and implement printed board circuit (electronic hardware) for a computer control system.

6. Implement a software-hardware verification using hardware-in-the-loop testing.

7. Conduct experimental modelling to assist in the design and tuning of control systems.

* + 1. Week 11 - MCHA3000\_Project\_report.pdf



* 1. State Space Calculations

1. https://www.sparkfun.com/products/10982 [↑](#footnote-ref-1)
2. http://au.element14.com/bourns/91a1a-b28-b15l/potentiometer-10k/dp/9357769 [↑](#footnote-ref-2)
3. MCHA3000 Week 11 Lecture Slides, Dr Chris Renton [↑](#footnote-ref-3)
4. Project outline, Dr Chris Renton [↑](#footnote-ref-4)