NTNU
Norwegian University of
Science and Technology

Two-Axis Reaction Wheel Inverted Pendulum

Petter Brevik

Master of Science in Cybernetics and Robotics
Submission date: May 2017
Supervisor: Tor Engebret Onshus, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Task Description

Title: Two-Axis Reaction Wheel Inverted Pendulum

Task given: 19.12.2016

Deadline: 16.05.2017

Supervisor: Professor Tor Onshus, Department of Engineering Cybernetics, NTNU

Student: Petter Brevik

The inverted pendulum is a great example of an unstable system, and there are multiple ways of applying control to such a system in order to make it stable. By adding reaction wheels to an inverted pendulum it should be possible to stabilize it, actuating the reaction wheel and thus creating torque on the pendulum.

The objective is to perform modeling, control design, simulations, and physical testing for balancing an inverted pendulum in two axes, by using reaction wheels. The goal is to balance the inverted pendulum about its upright equilibrium, while the angular velocity of the reaction wheels goes to zero.

The student should:

1. Do a literary study (the theory behind, similar projects, etc.)
2. Develop a control algorithm for balancing the inverted pendulum.
3. Make a Simulink model and run simulations.
4. Build an experimental model, implement the control algorithm and run tests.
5. Evaluate results and suggest further work.

Preface

This master thesis is performed as the conclusion of my Master of Science in Engineering Cybernetics at the Norwegian University of Science and Technology.

I would like to thank my supervisor, Professor Tor Onshus, for his guidance regarding this report.

I would also like to thank my family and friends for their support and encouragement throughout my studies.

Trondheim, 2017-05-16

Petter Brevik

Abstract

This report presents the results from development and design of a two-axis reaction wheel inverted pendulum. It shows the mathematical modeling of the system using Euler-Lagrange's equations of motion, as well as MATLAB and Simulink implementation.

A strategy for balance control of the system is presented and compared through simulation, ultimately showing that the system is theoretically controllable with the given sensor measurements.

Further the report presents an experimental design of the system and implements necessary functions for measurements and balance control.

The results for both the simulations and the experimental tests are presented in this report. Although in the end the experimental model presented in this report deemed unable to stabilize about the upright equilibrium.

Sammendrag

Denne rapporten presenterer resultatene fra utvikling og design av en to-akse reaksjonshjul-basert invertert pendel. Den viser den matematiske modelleringen av systemet ved bruk av Euler-Lagrange's bevegeleseslikninger, så vel som implementering i MATLAB og Simulink.

En strategi for balansekontroll av systemet er presentert og sammenliknet gjennom simuleringer, som viser at systemet teoretisk er kontrollerbart med de gitte sensormålingene.

Videre presenterer rapporten et eksperimentelt design av systemet og implementasjon av nødvendige funksjoner for å lese sensormålinger og for balansekontroll.

Resultatene for både simuleringer og eksperimentelle tester presenteres i denne rapporten. Selv om den eksperimentelle modellen presentert i denne rapporten, til slutt viste seg å ikke klare å stabilisere seg om det oppreiste likevektspunktet.

Contents

Task Description **i**

Preface **iii**

Abstract **v**

Sammendrag **vii**

Nomenclature **xix**

1 Introduction **1**
1.1 Motivation 1
1.2 Background 1
1.2.1 Reaction Wheel 2
1.2.2 Reaction Wheel Pendulum 2
1.2.3 Similar work 3
1.3 Contribution of the Thesis 4
1.4 Outline of the Thesis 4

2 Kinematics and Dynamics **7**
2.1 Kinematics 7
2.1.1 Angular Displacement 7
2.1.2 Angular Velocity 8
2.1.3 Angular Acceleration 8
2.2 Dynamics 8
2.2.1 Moment of Inertia 9
2.2.2 Torque 9
2.2.3 Kinetic Energy 10

3 Modeling **11**
3.1 Analytical Mechanics 11
3.1.1 Approach 11
3.1.2 Lagrange's Equations of Motion 11
3.2 Derivation of Mathematical Model 12
3.2.1 Kinetic and Potential Energy 13
3.2.2 Equations of Motion 13
3.2.3 State-Space Model 14
3.3 Controller Design 15
3.3.1 PID Controller 15
3.3.2 Linear Quadratic Controller 16

4 Model Implementation and Simulation **17**
4.1 Model Implementation 17
4.1.1 Model Parameters 17
4.1.2 Simulink Model 17
4.1.3 PID Controller 18
4.1.4 PD+P controller 19
4.1.5 LQ Controller 19
4.2 Simulations 20
4.2.1 Case 1: No Control 21
4.2.2 Case 2: P Control on θ_p 21
4.2.3 Case 3: PID Control on θ_p 23
4.2.4 Case 4: LQ Control 24
4.2.5 Case 5: PID Control, Impulse Response 27
4.2.6 Case 6: LQ Control, Impulse Response 31
4.2.7 Case 7: PD Control on θ_p + P Control on $\dot{\theta}_w$ . . . 33
4.2.8 Discussion 39

5 Experimental Design **41**
5.1 Hardware 41
5.1.1 Development Board 41
5.1.2 Motors and Motor Drivers 42
5.1.3 Sensors 45
5.1.4 Reaction Wheels 46
5.2 Hardware Implementation 46
5.2.1 Build Plan and Goals 47
5.2.2 Final Design 47

5.3 Software Implementation 48
5.3.1 Sensors 48
5.3.2 Controllers 49
5.4 Discussion 49

6 Experimental Results and Discussion **51**
6.1 Experimental testing 51
6.1.1 Case 1: Functionality Tests Sensors 51
6.1.2 Case 2: Balancing 54
6.2 Discussion 57

7 Conclusion and Further Work **59**
7.1 Conclusion 59
7.2 Further Work 59

A Digital Appendix **63**

List of Figures

1.1 Inside of a reaction wheel used in spacecrafts [17] 2
1.2 The Cubli [8] 3
1.3 The Self Balancing Stick, (screed capture, YouTube) [15] . 4

4.1 Simulink setup for simulations 18
4.2 State trajectories of θ_p with no added control 21
4.3 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane 22
4.4 State trajectories of θ_p with P control 23
4.5 State trajectories of θ_p with PID control without (a) and with measurement noise (b) 24
4.6 State trajectories of τ with PID control, without (a) and with measurement noise (b) 25
4.7 State trajectories of $\dot{\theta}_w$ with PID control and measurement noise 26
4.8 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PID control and measurement noise 26
4.9 State trajectories of θ_p with LQ control without (a) and with measurement noise (b) 28
4.10 State trajectories of τ with LQ control, without (a) and with measurement noise (b) 29
4.11 State trajectories of $\dot{\theta}_w$ with LQ control and measurement noise 30
4.12 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with LQ control and measurement noise 30
4.13 State trajectories of θ_p with PID control when pendulum is pushed 31

4.14 State trajectories of $\dot{\theta}_w$ with PID control when pendulum is pushed . . . 32
4.15 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PID control when pendulum is pushed . . . 32
4.16 State trajectories of θ_p with LQ control when pendulum is pushed . . . 33
4.17 State trajectories of $\dot{\theta}_w$ with LQ control when pendulum is pushed . . . 34
4.18 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with LQ control when pendulum is pushed . . . 34
4.19 State trajectories of θ_p with LQ control (a) and PID control (b) when pendulum is pushed . . . 35
4.20 State trajectories of θ_p with PID control on θ_p and P control on $\dot{\theta}_w$ . . . 36
4.21 State trajectories of $\dot{\theta}_w$ with PID control on θ_p and P control on $\dot{\theta}_w$ . . . 36
4.22 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PID control on θ_p and P control on $\dot{\theta}_w$ . . . 37
4.23 State trajectories of θ_p with PD control on θ_p and P control on $\dot{\theta}_w$ . . . 37
4.24 State trajectories of $\dot{\theta}_w$ with PD control on θ_p and P control on $\dot{\theta}_w$ . . . 38
4.25 State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PD control on θ_p and P control on $\dot{\theta}_w$ . . . 38

5.1 The Arduino UNO development board [3] . . . 43
5.2 The Mabuchi RS-385PH-16140 [13] . . . 44
5.3 The Arduino Motor Shield [2] . . . 44
5.4 The GY-521 Breakout Board, MPU-6050 [4] . . . 46
5.5 The Reaction Wheel designed in Fusion 360 . . . 47
5.6 The final design of the two axis reaction wheel inverted pendulum . . . 48

6.1 IMU measurements of the pendulum body's angular position, when still . . . 52
6.2 IMU measurements of the pendulum body's angular position, when rotated . . . 53

6.3 Speed measurement of DC motors, and the PWM values sent to the motors 53
6.4 Speed measurement of DC motors, and the PWM values sent to the motors 54
6.5 State trajectories of θ_p with applied control, $K_p = 670, K_d = 23, K_{p,w} = 0.8$ 55
6.6 State trajectories of $\dot{\theta}_p$ with applied control, $K_p = 670, K_d = 23, K_{p,w} = 0.8$ 55
6.7 State trajectories of θ_p with applied control, $K_p = 600, K_d = 30, K_{p,w} = 2$ 56
6.8 State trajectories of θ_p with applied control, $K_p = 600, K_d = 30, K_{p,w} = 2$ 56

List of Tables

4.1 Model Parameters 18
4.2 PID Controller Gains 19

5.1 Main Hardware Components 42

Nomenclature

ARE Algebraic Riccati Equation

IDE Integrated Development Environment

IMU Inertial Measurement Unit

LQ Linear-Quadratic

LQR Linear-Quadratic Regulator/Controller

MEMS MicroElectoMechanical Systems

NTNU Norsk Teknisk- og Naturvitenskaplige Universitet
Norwegian University of Science and Technology

PID Proportional-Integral-Derivative (Controller)

PWM Power-Width Modulation

RWIP Reaction Wheel Inverted Pendulum

RWP Reaction Wheel Pendulum

Chapter 1

Introduction

1.1 Motivation

Inverted pendulums have been well researched throughout the years, and different methods of stabilization have been implemented. For example by actuating the pendulum body, or attaching the pendulum to an actuated cart. In this thesis the pendulum will be tried stabilized in two axis by using the torque from reaction wheels, such as used in spacecrafts.

Stabilizing an inverted pendulum by use of reaction wheels, makes for an interesting project that shows Newton's third law of motion applied to circular motion. Utilizing the reaction wheel to stabilize the inverted pendulum is not a new concept, but nonetheless an interesting concept that will be discussed in this report. This report will use two reaction wheels to stabilize an inverted pendulum in two axis.

1.2 Background

A pendulum is a mass suspended from a fixed point, so that it can swing freely. If the pendulum is displaced from its equilibrium point, the gravitational force applied to the pendulum mass will accelerate it back towards the equilibrium point.

There are many uses for the pendulum, extending from pendulum clocks, to measuring gravitational acceleration, to demonstration of the earths

Figure 1.1: Inside of a reaction wheel used in spacecrafts [17]

rotation, as is the case for the Foucault pendulum at the Gløshaugen campus of the Norwegian University of Technology and Science.

1.2.1 Reaction Wheel

Reaction wheels are flywheels that are used in spacecrafts to provide attitude control. By applying torque to the flywheel, the torque is transferred to the spacecraft, causing it to rotate. By using three or more reaction wheels, one can provide full three-axis attitude control and stability to a spacecraft [17]. Using reaction wheels instead of propulsion-system thrusters lowers the weight of a spacecraft by eliminating the need for fuel, as reaction wheels are electrically powered [6]. An example of a reaction wheel used in spacecrafts can be seen in figure 1.1

1.2.2 Reaction Wheel Pendulum

The reaction wheel pendulum was first introduced in 1999 by Spong et al. in the paper "Nonlinear control of the Reaction Wheel Pendulum"[18]. The reaction wheel pendulum is an underactuated system and consist of an unactuated pendulum with a rotating reaction wheel stuck to the end. The pendulum is controlled by actuating the reaction wheel and transforming the rotational acceleration of the wheel into torque working on the pendulum. The goal of the reaction wheel pendulum is to stabilize the pendulum at its upright equilibrium.

This thesis extends the reaction wheel pendulum introduced in [18] by

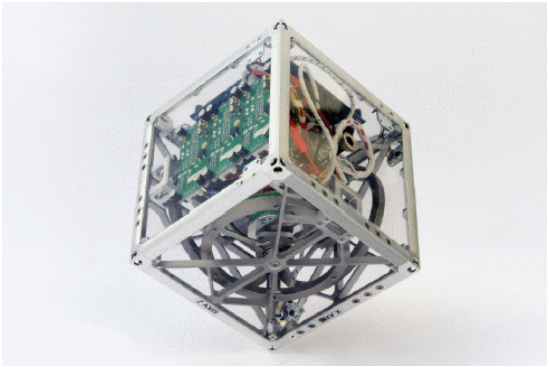


Figure 1.2: The Cubli [8]

adding another reaction wheel, and making the pendulum free standing, allowing it to move in two axes instead of one.

1.2.3 Similar work

As mentioned previously the reaction wheel pendulum for one axis control have been introduced previously. The setup uses a pendulum attached to the side of a table, a reaction wheel actuated by a permanent magnet DC motor, and high-resolution encoders for both the pendulum body and the reaction wheel. This experiment, performed at the University in Illinois aims to swing the pendulum up from its downright equilibrium, to its upright equilibrium and balance it there.[18]

Another very interesting project is the Cubli, introduced at the Swiss Federal Institute of Technology Zrich in 2012 [9]. The Cubli is a cube that utilizes three reaction wheels to be able to balance on any edge or corner, and is able to jump up from any side, perform controlled falls and thus being able to walk. [8] The Cubli can be seen in figure 1.2

Another project, that gave inspiration to this thesis, is the Self Balancing Stick, which is a dual-axis reaction wheel inverted pendulum. This project was done on a hobby basis and lacks any proper documentation [14]. The Self Balancing Stick can be seen in figure 1.3

Figure 1.3: The Self Balancing Stick, (screed capture, YouTube) [15]

1.3 Contribution of the Thesis

This thesis includes these main contributions:

- MATLAB and Simulink model of a two axis reaction wheel pendulum.
- Controller comparison for the two-axis reaction wheel inverted pendulum.
- Experimental build and test of controllers.

1.4 Outline of the Thesis

This thesis starts with some theory about kinematics and dynamics used to describe the system. A model is then designed and implemented in MATLAB and Simulink, followed by controller design and controller comparison. In the last section of this thesis the an experimental model is designed and tested with the desired controller.

The thesis is organized in the following way:

Chapter 1: Introduces the problem and contributions in this thesis.
Chapter 2: Gives some theory on the kinematics and dynamics of the system.
Chapter 3: Derives the equations of motion for the system and introduces controller design.

Chapter 4: Presents the model implementation and simulation results of the system.
Chapter 5: Presents the experimental design and implementation
Chapter 6: Presents the results and discussion obtained by experimental tests.
Chapter 7: Conclusion of the project and proposes further work

The following appendix is included:

Appendix A: Digital Appendix

Chapter 2

Kinematics and Dynamics

In this chapter the dynamics of the reaction wheel pendulum will be described using kinematics and dynamics. Section 2.1 will describe the kinematics of a rigid body rotating around a fixed axis. Section 2.2 will focus on the dynamics, or the relationship between kinematics and the moments causing motion in the system, torque. This chapter is mainly based on [10]

2.1 Kinematics

The pendulum and the reaction wheel are examples of rigid bodies rotating around fixed axes, and this section will describe the kinematics of such bodies.

2.1.1 Angular Displacement

To get the angular displacement of a rigid body around a fixed axis, we first need to find the angular position.

Lets assume a particle moving in a circle of radius r around the rotational axis. Moving an arc length s we get the angular position θ as:

$$\theta = \frac{s}{r} \tag{2.1}$$

The angular position is given in radians (rad), and one revolution around the rotational axis equals 2π rad.

The angular displacement ($\Delta\theta$) of a rigid body rotating around a fixed axis is the change of angular position. This is given by:

$$\Delta\theta = \theta_2 - \theta_1 \tag{2.2}$$

where θ_1 is the initial angular position and θ_2 is the final angular position.

2.1.2 Angular Velocity

The angular velocity is described as the angular displacement per unit time and is usually denoted ω and given in rad/s. Taking the angular displacement from equation 2.2 and the time difference between θ_1 and θ_2, Δt, we get:

$$\omega = \frac{\Delta\theta}{\Delta t} \tag{2.3}$$

This gives the instantaneous angular velocity:

$$\omega = \frac{d\theta}{dt} = \dot{\theta} \tag{2.4}$$

The tangential velocity can be related as:

$$v = \frac{ds}{dt} = r\frac{d\theta}{dt} = r\omega \tag{2.5}$$

2.1.3 Angular Acceleration

Angular acceleration, α, is described as the change in angular velocity per unit time. It is given in rad/s^2, and the instantaneous angular acceleration can be derived as:

$$\alpha = \frac{d\omega}{dt} = \frac{d^2\theta}{dt^2} \tag{2.6}$$

2.2 Dynamics

This section will describe the dynamics of rigid bodies rotating about fixed axes by using the kinematics of such systems, as described in the previous section.

2.2.1 Moment of Inertia

The moment of inertia about a fixed axis can be described as the body's resistance to change to its rotation, and is given by:

$$I = \sum_i m_i r_i^2 \tag{2.7}$$

where I denotes the moment of inertia of the object, and m_i denotes the particle mass at radius r_i about the rotational axis.

From this we can see that an increase in the mass of the body, will increase the moment of inertia. More importantly, we can see that by displacing the body's mass away from the rotational axis, we can increase the moment of inertia without increasing the mass of the body.

Parallel Axis Theorem

The parallel axis theorem or Steiner's theorem gives the relation between a body's moment of inertia about its center of mass, and its moment of inertia about an arbitrary axis, parallel to the center of mass.

The parallel axis theorem:

$$I = I_0 + mb^2 \tag{2.8}$$

where I is the moment of inertia about a parallel axis at distance b from the center of mass, m is the total mass of the body, and I_0 is the moment of inertia about the center of mass.

2.2.2 Torque

When a force F is applied to a rotating object at position r from the axis of rotation, we get the torque, τ. This can be described as:

$$\tau = rFsin(\theta) = (\sum_i m_i r_i)\frac{d^2\theta}{dt^2} \tag{2.9}$$

which can be translated to:

$$\tau = I\alpha \tag{2.10}$$

where I is the moment of inertia given by equation 2.7 and α is given by equation 2.6.

2.2.3 Kinetic Energy

The kinetic energy of a rigid body rotating about a fixed axis is the kinetic energy from the movement relative to the center of mass and the kinetic rotational energy about the center of mass. The total kinetic energy is given by:

$$K = \frac{1}{2}mv^2 + \frac{1}{2}I\omega^2 \tag{2.11}$$

Chapter 3

Modeling

This chapter will show the theoretical modeling of the reaction wheel pendulum. It will start off by a short summary of the methods used. Then this theory will be implemented and the equations of motion of the system will be derived.

3.1 Analytical Mechanics

This section summarizes the theory used to derive the equations of motion for the system, and is based on [7]

3.1.1 Approach

There are a few different approaches to deriving the equations of motion for a mechanical system. We have Hamilton's equations of motion, the Newton-Euler equations and the Lagrangian equations of motion. The Lagrangian approach is similar to that of Newton-Euler, and end up in the same equations of motion, but they use different methods to derive the equations of motion. While Newton-Euler uses vector operations, Lagrange uses algebraic operations and is a less complicated approach.

3.1.2 Lagrange's Equations of Motion

Lagrange's equations of motion for a mechanical system can be derived by selecting a set of generalized coordinates, q_i, and using generalized forces.

Lagrange's equations of motion is formulated by using the Lagrangian:

$$L(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) = T(\boldsymbol{q}, \dot{\boldsymbol{q}}, t) - U(\boldsymbol{q}) \tag{3.1}$$

where, $T(\boldsymbol{q}, \dot{\boldsymbol{q}}, t)$ is the kinetic energy of the system and $U(\boldsymbol{q})$ is the potential energy of the system. The equations of motion is then derived as:

$$\frac{d}{dt}(\frac{\delta L}{\delta \dot{q}_i}) - \frac{\delta L}{\delta q_i} = \tau_i \tag{3.2}$$

where τ_i is the generalized actuator torque.

3.2 Derivation of Mathematical Model

In this section we will use the theory discussed in chapter 2 to find the Lagrangian of the system. Then the equations of motion for the system will be derived and the state-space model of the linearized system will be derived.

Some assumptions were made in order to ease the modeling of the system.

- The control will happen in close proximity of the upright equilibrium and we will assume that the tipping angle in one axis does not affect the other axis. Hence the model can be modeled in one axis and the model reused for the other axis.
- The pendulum friction and the reaction wheel friction is neglected.
- The center of mass of the reaction wheel coincide with its axis of rotation.
- The angle of the pendulum is measured from the vertical upright equilibrium.

The reaction wheel pendulum was first introduced in 1990 and this section is based on [18].

3.2.1 Kinetic and Potential Energy

In order to derive the equations of motion, we first need to find the kinetic (T) and potential (U) energies of the system.

Let $\theta = (\theta_p, \theta_w)^T \in \Re^2$ denote the configuration vector of the system and M be the constant inertia matrix with elements:

$$m_{11} = m_p l_p^2 + m_w l_w^2 + I_1 + I_2 \tag{3.3}$$

$$m_{12} = m_{21} = m_{22} = I_2 \tag{3.4}$$

where the inertia matrix, M is derived using the parallel axis theorem given in equation 2.8.

We can then calculate the kinetic (T) energy, using equation 2.11, and potential energy (U) as:

$$T = \frac{1}{2}\dot{\theta}^T M \dot{\theta} \tag{3.5}$$

$$U = (m_p l_p + m_w l_w) g_0 \cos\theta_p = m_0 \cos\theta_p \tag{3.6}$$

where $m_0 = (m_p l_p + m_w l_w) g_0$, and g_0 is the gravitational force, and the kinetic energy is solely rotational.

The Lagrangian of the system can then be calculated by putting equations 3.5 and 3.6 into equation 3.1:

$$L(\theta, \dot{\theta}) = \frac{1}{2}\dot{\theta}^T M \dot{\theta} - m_0 \cos\theta_p \tag{3.7}$$

3.2.2 Equations of Motion

To calculate the Euler-Lagrange equations of motion, we will use equation 3.2. Since we have two sets of generalized coordinates $q_1 = \theta_p$ and $q_2 = \theta_w$, and only one actuator torque working on the reaction wheel, $\tau_1 = 0$ and $\tau_2 = \tau$, we can calculate the Euler-Lagrange equations of motion:

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\theta}_p}\right) - \frac{\delta L}{\delta \theta_p} = 0 \tag{3.8}$$

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{\theta}_w}\right) - \frac{\delta L}{\delta \theta_w} = \tau \tag{3.9}$$

By using L as given by equation 3.7, the Euler-Lagrange equations of motion can be calculated as:

$$m_{11}\ddot{\theta}_p + m_{12}\ddot{\theta}_w = \phi(\theta_p) \tag{3.10}$$

$$m_{21}\ddot{\theta}_p + m_{22}\ddot{\theta}_w = \tau \tag{3.11}$$

where $\phi(\theta_p) = m_0 \sin\theta_p$

By combining equations 3.10 and 3.11 and solving for $\ddot{\theta}_p$ and $\ddot{\theta}_w$ respectively, we get the following two equations of motion:

$$\ddot{\theta}_p = \frac{m_{22}\phi(\theta_p)}{m_{11}m_{22} - m_{12}m_{21}} - \frac{m_{12}\tau}{m_{11}m_{22} - m_{12}m_{21}} \tag{3.12}$$

$$\ddot{\theta}_w = -\frac{m_{21}\phi(\theta_p)}{m_{11}m_{22} - m_{12}m_{21}} + \frac{m_{11}\tau}{m_{11}m_{22} - m_{12}m_{21}} \tag{3.13}$$

3.2.3 State-Space Model

The system can be represented in state-space form:

$$\dot{x} = Ax + Bu \tag{3.14}$$

In order to represent the model in state space, the equations 3.12 and 3.13 has to be linearized about an equilibrium. Linearizing about the upright equilibrium, $\theta_p = 0$, choosing $x = (\theta_p, \dot{\theta_p}, \dot{\theta_w})^T$ and substituting $\tau = K_m u$, where K_m is the motor torque constant and u is the control signal, we get the controllable linear system given by equation 3.14 with matrices:

$$A = \begin{bmatrix} 0 & 1 & 0 \\ \frac{m_{22}m_0}{detM} & 0 & 0 \\ -\frac{m_{21}m_0}{detM} & 0 & 0 \end{bmatrix} \tag{3.15}$$

$$B = \begin{bmatrix} 0 \\ -\frac{m_{12}K_m}{detM} \\ \frac{m_{11}K_m}{detM} \end{bmatrix} \tag{3.16}$$

3.3 Controller Design

In this section we will look at two different approaches to controller design, the PID controller and the LQ controller respectively. Both controllers will later be implemented in MATLAB and Simulink and compared. They will also be implemented and tested on the experimental build.

3.3.1 PID Controller

The Proportional-Integral-Derivative (PID) controller is a common controller in industrial control system, and it is often used when there is no deeper understanding of the system one is trying to control.

The PID controller takes the error, $e(t)$, given by the difference between a setpoint $r(t)$ and the measured value given by the system $y(t)$, and attempts to minimize the error over time by adjusting the control variable $u(t)$. For the PID controller the control variable is calculated in continuous time as:

$$u(t) = K_p(e(t) + \frac{1}{T_i}\int_0^\infty e(\tau)d\tau + T_d\frac{de(t)}{dt}) \tag{3.17}$$

where K_p, T_i and T_d are the proportional, integral and derivative gains respectively. The proportional gain penalizes the current error, $e(t)$, while the integral and derivative gains penalize the past error and the possible future error [5].

The controller gains can be either chosen by trial and error, or they can be calculated using a tuning method such as Ziegler-Nichols [20].

Ziegler-Nichols method is describes as follows:

- Set $T_i = \infty$ and $T_d = 0$.
- Slowly increase K_p while adding small disturbances to the system, until the system output $y(t)$ has stable, consistent oscillations.
- This value of K_p is called the critical gain, K_{pk}, and the period of the oscillation is called T_k.

- The PID gains can then calculated as follows:

$$\begin{aligned} K_p &= 0.6K_{pk} \\ T_i &= 0.5T_k \\ T_d &= 0.75T_k \end{aligned} \tag{3.18}$$

3.3.2 Linear Quadratic Controller

The Linear-Quadratic (LQ/LQR) controller can be used when one has derived a linearized system model represented in state-space

Using state-space representation of the system, as shown in the previous section, and adding a state feedback control $u = -Kx$, where K is the feedback gain vector, we can stabilize the system by choosing a vector K that minimizes the quadratic cost function [1]:

$$J = \int_0^\infty (x^T Q x + u^T R u + 2x^T N u) dt \tag{3.19}$$

where $Q > 0$ and $R \geq 0$ are constant, positive-definite matrices. The Q matrix penalizes the transient state deviation, and the R matrix penalizes control effect.

The control gain vector is given by:

$$K = R^{-1}(B^T P + N^T) \tag{3.20}$$

where P is the positive definite matrix that satisfies the algebraic Riccati equation (ARE):

$$A^T P + PA - (PB + N)R^{-1}(B^T P + N^T) + Q = 0 \tag{3.21}$$

The control goal is to keep x close to 0.

Chapter 4

Model Implementation and Simulation

This chapter shows the implementation of the system and controllers in MATLAB and Simulink. Further this chapter will show simulations performed in order to tune and compare controllers.

4.1 Model Implementation

The theoretical model of the reaction wheel inverted pendulum derived in chapter 3 was implemented in MATLAB and Simulink in order to run simulations of the system and test controller designs. In this section the implementation process will be explained.

4.1.1 Model Parameters

The model was implemented with parameters as shown in table 4.1. The parameters were either measured from the physical model or taken from the data sheets of the components.

4.1.2 Simulink Model

Working off the assumptions made in section 3.2 the two-axis reaction wheel inverted pendulum was implemented in Simulink as shown in figure 4.1. The model is implemented as two one-axis reaction wheel inverted pendulums operating independently of each other. This is due to

Table 4.1: Model Parameters

Parameter	Value
l_w	0.330 [m]
l_p	0.185 [m]
m_w	0.588 [kg]
m_p	0.033 [kg]
I_w	$2.662e^{-4}$ [kg m^2]
I_p	$3.765e^{-4}$ [kg m^2]
g_0	9.81 [m/s^2]
K_m	$5.5e^{-}3$ [Nm/$\sqrt{(W)}$]
T_max	0.600 [kg cm]

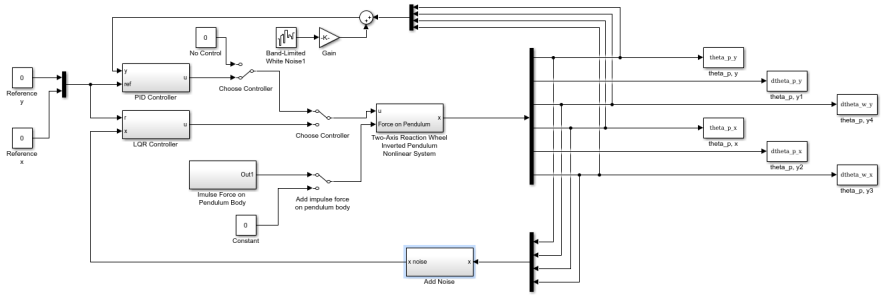


Figure 4.1: Simulink setup for simulations

that the operating area is in such close proximity to the upright equilibrium that the correlating effect of the two axes is neglected.

4.1.3 PID Controller

A PID controller was implemented in order to stabilize the reaction wheel inverted pendulum model around the upright equilibrium. The PID controller was implemented as shown in figure 4.1 taking the pendulum angular position and a reference angle as input.

In order to decide the values of the PID controller gains, the Ziegler-Nichols method was applied as discussed in section 3.3.1

Table 4.2: PID Controller Gains

Controller Gain	Value
Proportional, K_p	1200
Integral, T_i	0.5
Derivative, T_d	0.75

The final PID gain values that were found from this method are shown in table 4.2

4.1.4 PD+P controller

The PID controller will be shown to be insufficient in the simulations in the coming section. Therefore a Proportional controller was implemented on $\dot{\theta}_w$ to account for the error in the angular velocity of the motor, when an impulse external force is applied to the pendulum body.

The controller gain values are the same as for the PID, but with $T_i = \infty$ and $K_{p,w} = 0.2$.

4.1.5 LQ Controller

As one of the goals of this project was to try different controllers, a linear quadratic controller was also implemented to stabilize the system. The LQ controller was implemented as shown in figure 4.1. The LQ controller takes all the model states as input and uses weight matrices Q and R to minimize the cost function in equation 3.19.

The weight matrices were chosen as:

$$Q = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0.8 & 0 \\ 0 & 0 & 0.6 \end{bmatrix} \tag{4.1}$$

$$R = 1 \tag{4.2}$$

Using the MATLAB command:

$$lqr(A, B, Q, R)$$

where A and B are the state-space matrices given in equation 3.15 and 3.16, with the model parameters as given in table 4.1.

The resulting control gain vector is then given as:

$$K = [-2095.8 \quad -383.7 \quad -0.8] \tag{4.3}$$

where K gives the state feedback control $u = -Kx$

4.2 Simulations

Simulations were carried out in MATLAB and Simulink in order to prepare for experimental tests, and to test different controllers. The model was simulated as a two-axis reaction wheel inverted pendulum, where the axes act independently of each other, as explained in the previous section.

The IMU and reaction wheel speed measurements were added noise to make the simulations more realistic.

This section is organized in the following way:

- Section 4.2.1: Case 1: No Control
- Section 4.2.2: Case 2: P Control on θ_p
- Section 4.2.3: Case 3: PID Control on θ_p
- Section 4.2.4: Case 4: LQ Control
- Section 4.2.5: Case 5: PID Control, Impulse Response
- Section 4.2.6: Case 6: LQ Control, Impulse Response
- Section 4.2.7: Case 7: PD Control on θ_p + P Control on $\dot{\theta}_w$
- Section 4.2.8: Discussion

The setup for all parameters are as given in Section 4.1, and the initial conditions for the system was set to:

$$\theta_0 = \begin{bmatrix} \theta_{p0,x} & \theta_{p0,y} \\ \dot{\theta}_{p0,x} & \dot{\theta}_{p0,y} \\ \dot{\theta}_{w0,x} & \dot{\theta}_{p0,y} \end{bmatrix} = \begin{bmatrix} -\frac{\pi}{30} & \frac{\pi}{20} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{4.4}$$

unless otherwise specified.

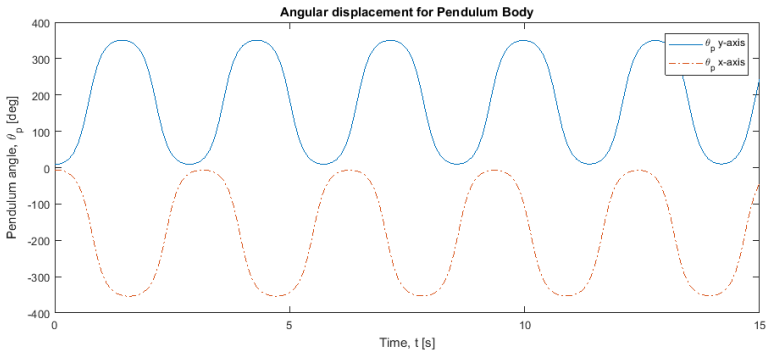


Figure 4.2: State trajectories of θ_p with no added control

4.2.1 Case 1: No Control

In the first simulation there was no applied control. This was done in order to show that, when starting with an offset from the unstable upright equilibrium, the model would swing about the stable downright equilibrium. This is of course not possible for the physical model, but it shows that the model works as expected.

Due to the simplifications made in the modeling process, the friction is set to zero, and thus the pendulum is expected to swing with constant amplitude.

Results

Figure 4.2 illustrates that the pendulum swings about the stable downright equilibrium, when the initial condition for θ_p is not equal to zero. This shows that the model behaves as expected and a controller can be implemented.

It should be noted that if the friction working on the system was also modeled, the pendulum would converge towards the stable equilibrium over time.

4.2.2 Case 2: P Control on θ_p

In the second case study, a P controller was added to θ_p. This was done in order to derive a PID controller in the next case study, using Ziegler-

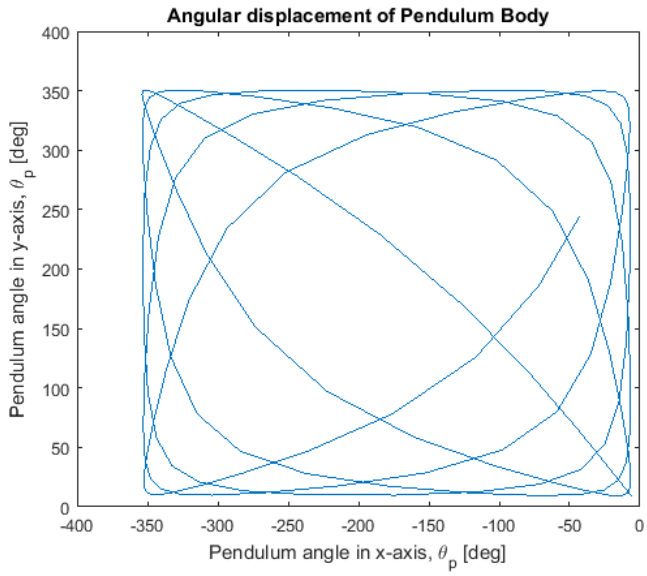


Figure 4.3: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane

Nichols method, as shown in section 3.3

Measurement noise were added to the model, T_i and T_d from equation 3.17 were set to infinity and zero respectively, and the initial condition of the pendulum angular position θ_p were set to a non-zero value. The proportional control gain K_p were then increased until the system output had stable, consistent oscillations.

Results

Figure 4.4 shows the oscillations of the system output $y(t) = \theta_p(t)$ with control value $K_p = 2000$.

The critical gain K_{pk} is then set as this value of K_p and the period of the oscillations as T_k. These values will be used in the next case study to implement a PID controller for the system.

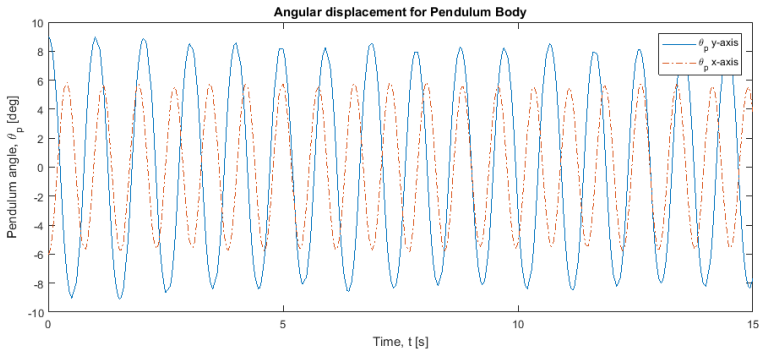


Figure 4.4: State trajectories of θ_p with P control

4.2.3 Case 3: PID Control on θ_p

Using the values for K_{pk} and T_k from the previous case study, we use the equations 3.18 to calculate the gain values for the PID controller.

We end up with the PID values shown in table 4.2.

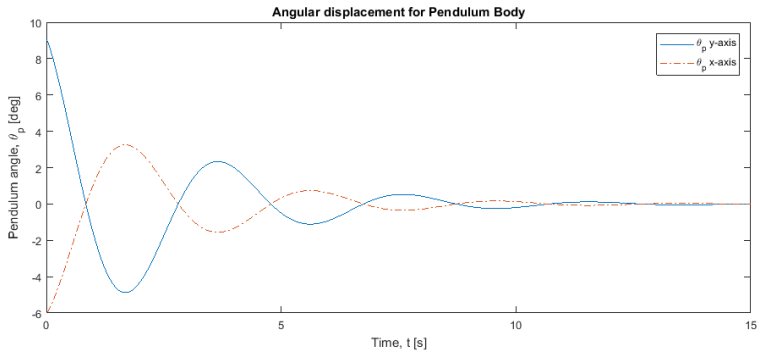
In this case study the PID controller is tested with and without measurement noise.

Results

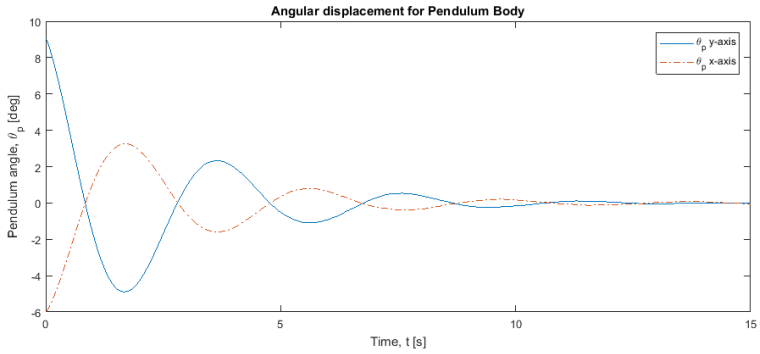
From figure 4.5 we can see that the implemented PID controller stabilizes the system about the upright equilibrium point.

From figure 4.7 we see that the reaction wheel speeds also converge to zero as the pendulum stabilizes about the equilibrium. This is the exact behavior we strive for.

We can also see from figure 4.5 that the added measurement noise has little to no effect on the state trajectories of θ_p. Although if we look at the trajectories of the applied torques τ in figure 4.6, we see that the controller makes rapid, relatively big corrections due to the measurement noise and this can cause problems on the experimental build. A filter should therefore be added to compensate for measurement noise. It should be noted however that such rapid changes in torque is not realistic.



(a) State trajectories of θ_p with PID control



(b) State trajectories of θ_p with PID control and measurement noise

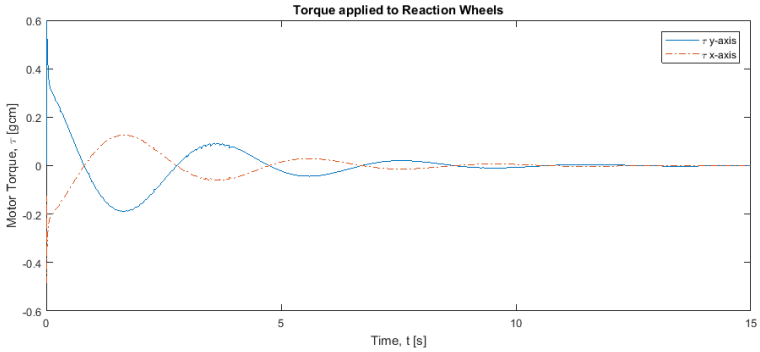
Figure 4.5: State trajectories of θ_p with PID control without (a) and with measurement noise (b)

Figure 4.8 shows the State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane and gives an indicator on how the pendulum will move in 3D space, here represented as a top down view.

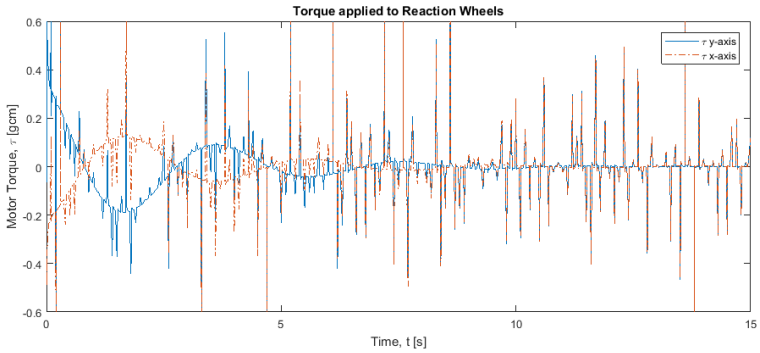
4.2.4 Case 4: LQ Control

In order to implement a linear quadratic controller, the state-space model as given by equations 3.14, 3.15 and 3.16 were used in the initialization process to calculate values for the gain vector K, given in equation 4.3. This was done as discussed in the previous section.

In this case study the linear quadratic controller was implemented both



(a) State trajectories of τ with PID control



(b) State trajectories of τ with PID control and measurement noise

Figure 4.6: State trajectories of τ with PID control, without (a) and with measurement noise (b)

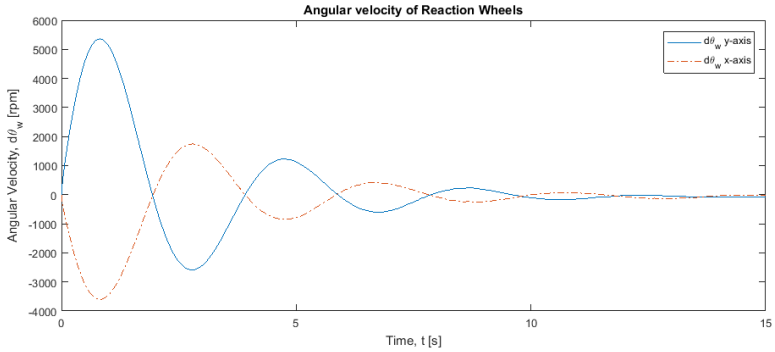


Figure 4.7: State trajectories of $\dot{\theta}_w$ with PID control and measurement noise

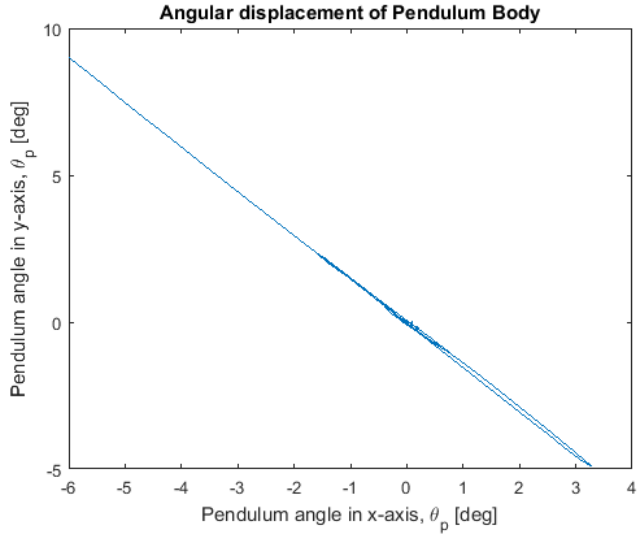


Figure 4.8: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PID control and measurement noise

with and without measurement noise on all measurements.

Results

From figure 4.9 we can see that the LQ controller is more aggressive than the PID controller in the previous case study, and that it successfully stabilizes the pendulum at the upright equilibrium point more rapidly than the PID controller and with less fluctuation. It does however seem like the LQ controller is more affected by the measurement noise, causing θ_p to deviate from the equilibrium point.

The effect of the measurement noise on the LQ controller can be seen in the trajectories of τ shown in figure 4.10.

Figure 4.12 shows the State trajectory in the $(\theta_{p,x}, \theta_{p,y})$ plane and gives an indicator on how the pendulum will move in 3D space, here represented as a top down view.

From figure 4.11 we can see that the reaction wheel speeds $\dot{\theta}_w$ converges to zero as the pendulum stabilizes about the equilibrium.

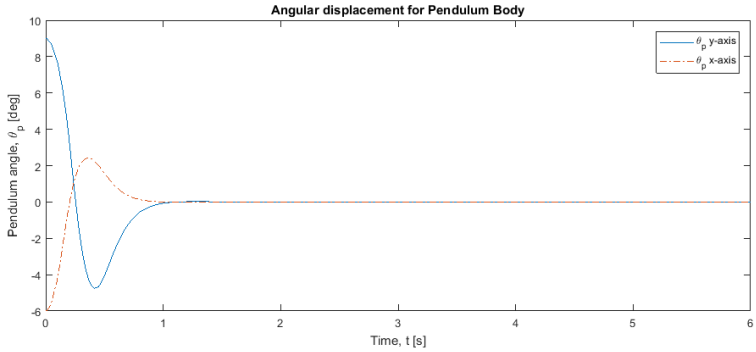
4.2.5 Case 5: PID Control, Impulse Response

In this case study a small impulse force is applied to the pendulum body, and it is to show how the PID controller handles such a disturbance. The disturbance rapidly changes $\theta_{p,x}$ and $\theta_{p,y}$.

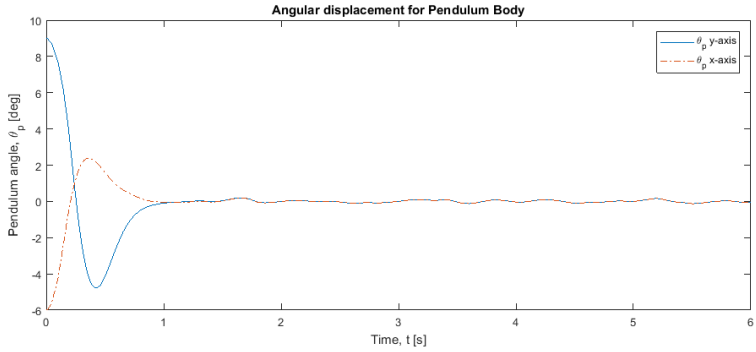
Results

As seen from figure 4.13 the PID controller handles the disturbance well, and converges back to the upright equilibrium.

Although the pendulum stabilizes after the disturbance, we can see from figure 4.14 that the angular velocity of the reaction wheels does not converge back towards zero, but instead converges to a non-zero value. This still lets the pendulum stabilize, as no torque will be applied to the system for a constant angular velocity, but this is not desirable as it increases the power draw of the system, and will prevent the system from stabilizing again if the same force is applied to the pendulum body, and the motor speed keeps drifting towards its maximum speed.

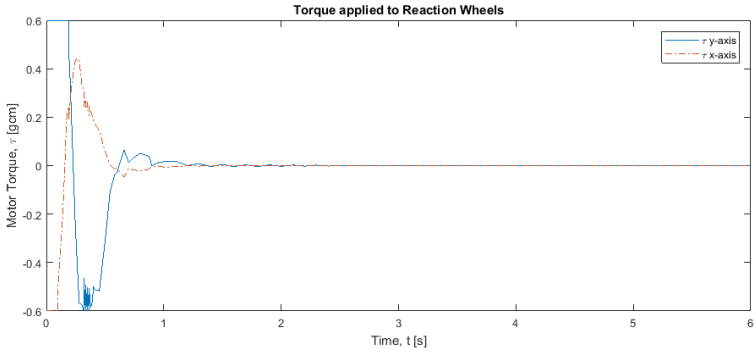


(a) State trajectories of θ_p with LQ control



(b) State trajectories of θ_p with LQ control and measurement noise

Figure 4.9: State trajectories of θ_p with LQ control without (a) and with measurement noise (b)



(a) State trajectories of τ with LQ control

(b) State trajectories of τ with LQ control and measurement noise

Figure 4.10: State trajectories of τ with LQ control, without (a) and with measurement noise (b)

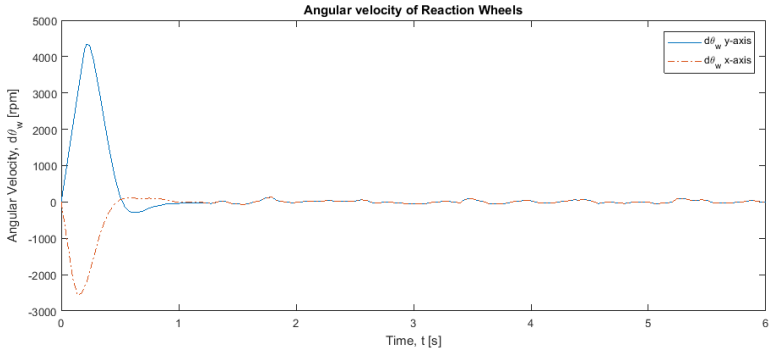


Figure 4.11: State trajectories of $\dot{\theta}_w$ with LQ control and measurement noise

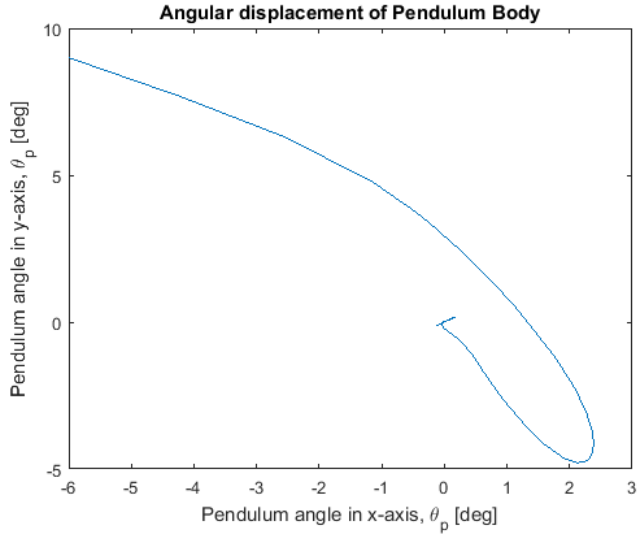


Figure 4.12: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with LQ control and measurement noise

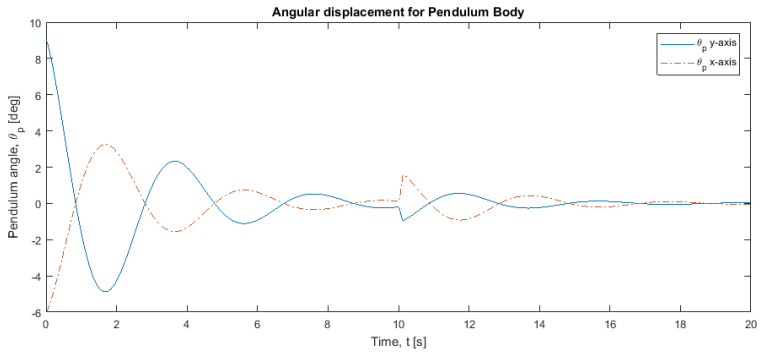


Figure 4.13: State trajectories of θ_p with PID control when pendulum is pushed

Figure 4.15 again shows a representation of a top down view of the pendulum.

4.2.6 Case 6: LQ Control, Impulse Response

In this case study the same small impulse force, as in the previous case study, is applied to the pendulum body to show how the LQ controller handles the same disturbance and to compare the two controllers.

Furthermore a larger impulse force is applied to both the PID and LQ controllers to show which of the two controllers are most vulnerable to disturbances and will not be able to stabilize.

Results

We can see from figure 4.16 that the LQ controller handles the disturbance well and shows the same rapid stabilization as with the initial response.

Figure 4.17 shows that the LQ controller, unlike the PID controller, also converges the angular momentum of the reaction wheels to zero, as is the goal of the controller. The LQ controller thus seems to handle the disturbance better than the PID controller.

However when we increase the force applied to the pendulum body, we

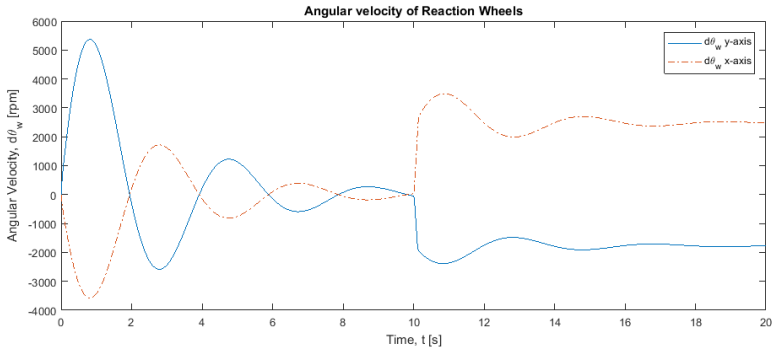


Figure 4.14: State trajectories of $\dot{\theta}_w$ with PID control when pendulum is pushed

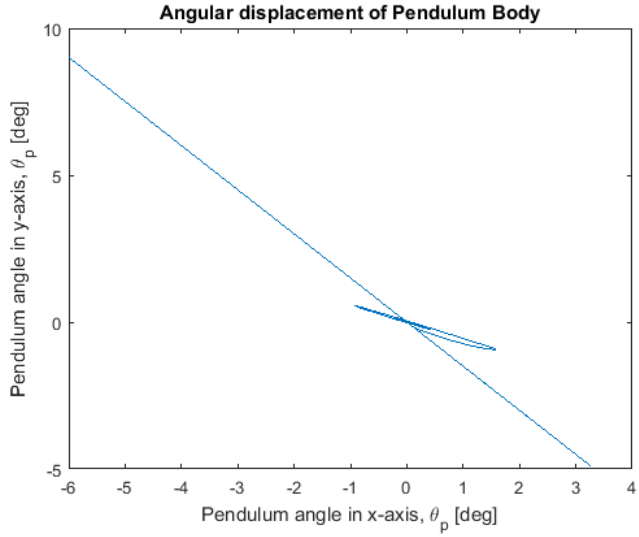


Figure 4.15: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PID control when pendulum is pushed

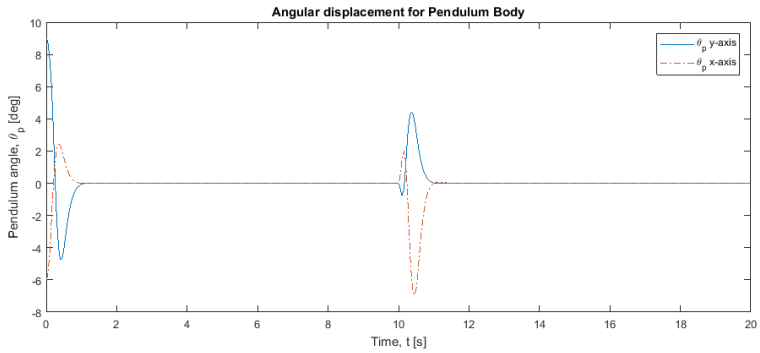


Figure 4.16: State trajectories of θ_p with LQ control when pendulum is pushed

can see from figure 4.19 that while the PID controller manages to regain stability, the LQ controller fails to do so and ends up falling.

4.2.7 Case 7: PD Control on θ_p + P Control on $\dot{\theta}_w$

The PID controller was shown to be more robust against impulse forces applied to the pendulum body, but failed to converge $\dot{\theta}_w$ to zero after the applied force. In this section a P controller will be added to $\dot{\theta}_w$ to account for this. But as will be shown in the results, this is not enough and the Integral term is removed from the PID controller, thus leaving us with a PD controller on θ_p and a P controller on $\dot{\theta}_w$.

Results

When the P controller on $\dot{\theta}_w$ was implemented alongside the PID controller on θ_p, the system still managed to stabilize around the upright equilibrium, but with more oscillations as shown in figure 4.20. Along this undesirable behavior the controller still failed to converge $\dot{\theta}_w$ to zero. This can be seen in figure 4.21.

By removing the Integral term in the PID controller on θ_p we end up with a controller that can handle disturbances and converges both θ_p and $\dot{\theta}_w$ to zero, even with applied impulse force to the pendulum body. This can be seen in figure 4.23 and 4.25 respectively.

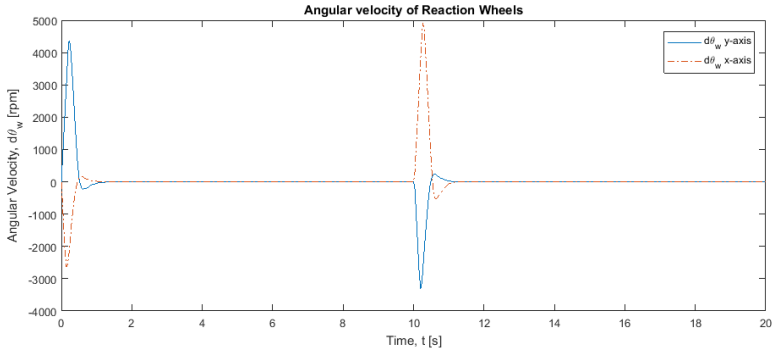


Figure 4.17: State trajectories of $\dot{\theta}_w$ with LQ control when pendulum is pushed

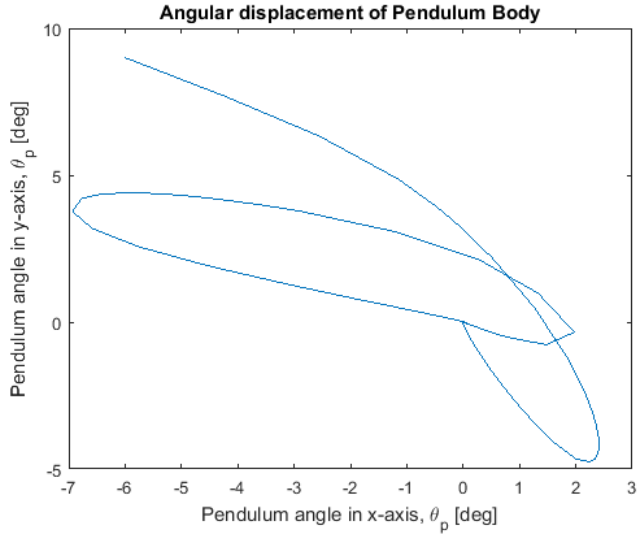
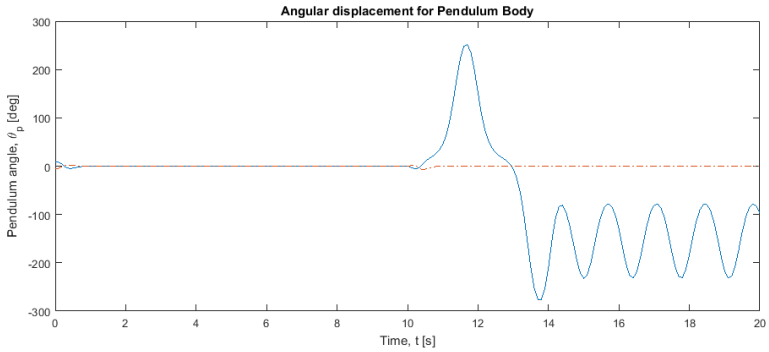
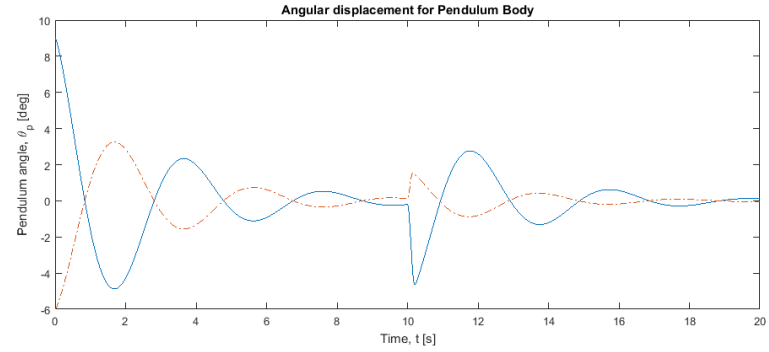


Figure 4.18: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with LQ control when pendulum is pushed



(a) State trajectories of θ_p with LQ control when pendulum is pushed



(b) State trajectories of θ_p with PID control when pendulum is pushed

Figure 4.19: State trajectories of θ_p with LQ control (a) and PID control (b) when pendulum is pushed

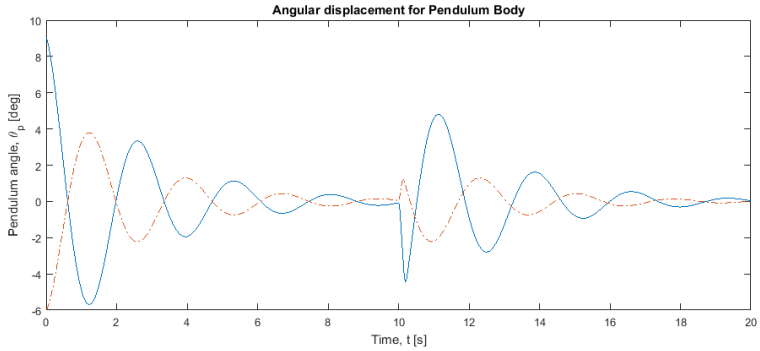


Figure 4.20: State trajectories of θ_p with PID control on θ_p and P control on $\dot{\theta}_w$

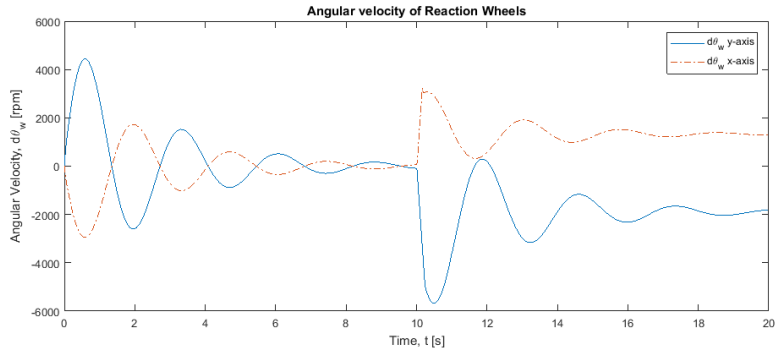


Figure 4.21: State trajectories of $\dot{\theta}_w$ with PID control on θ_p and P control on $\dot{\theta}_w$

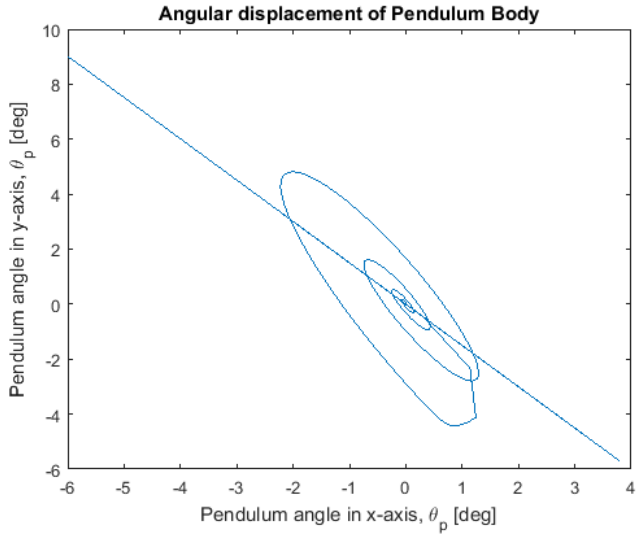


Figure 4.22: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PID control on θ_p and P control on $\dot{\theta}_w$

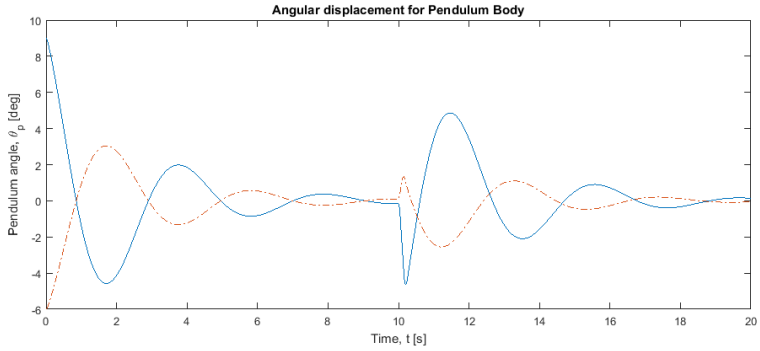


Figure 4.23: State trajectories of θ_p with PD control on θ_p and P control on $\dot{\theta}_w$

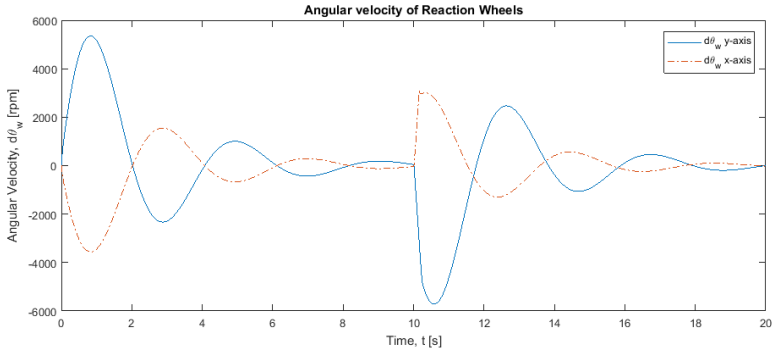


Figure 4.24: State trajectories of $\dot{\theta}_w$ with PD control on θ_p and P control on $\dot{\theta}_w$

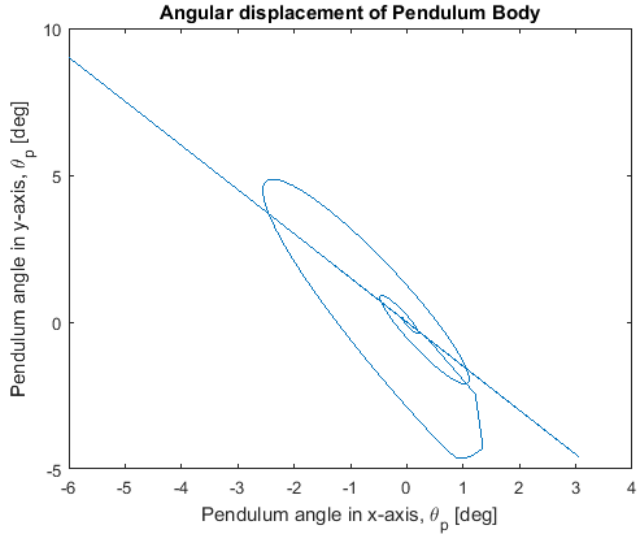


Figure 4.25: State trajectory in $(\theta_{p,x}, \theta_{p,y})$ plane with PD control on θ_p and P control on $\dot{\theta}_w$

4.2.8 Discussion

From the case studies performed in this section it has been shown that a PID controller on θ_p is not enough to attain the goal of this project, which is to stabilize the system about the upright equilibrium while also converging the motor speeds, $\dot{\theta}_w$, to zero.

It was then shown that a LQ controller is able to attain that goal, but the LQ controller used here, with the chosen Q and R matrices, is not as robust against outer disturbances as the PID controller. This can most likely be fixed by doing multiple tests and changing the Q and R matrices, using the trial and error approach.

By introducing a Proportional controller to $\dot{\theta}_w$ and removing the Integral term in the PID controller to θ_p, it was shown that the system can be stabilized and the motor speeds converged to zero, while at the same time being more robust than the LQ controller. The resulting PD+P controller thus satisfies the control goal better than the LQ controller, although being slower.

Chapter 5

Experimental Design

This chapter describes the experimental design of the two-axis reaction wheel pendulum.

The first section of this chapter will present the main hardware component, along with some theory and reasoning behind the choice. In the second section the hardware implementation and build will be presented. And finally the software implementation will be presented i the third section.

5.1 Hardware

In this section the main components of the two-axis reaction wheel pendulum will be presented. The reasoning behind the selection and some theory on the components will also be presented. The total cost of the project played a big role in the hardware chosen, due to a low personal budget.

All the main hardware components are listed in table 5.1

5.1.1 Development Board

The development board used in this project is the Arduino UNO shown in figure 5.1. It is an easy to use and widely available development board based on the ATmega328P microchip, produced by Atmel Cooperation. The Arduino UNO has 14 digital input/output pins, 6 of which can be

Table 5.1: Main Hardware Components

Component	Details	Quantity
Development Board:	Arduino Uno	1
Motor:	Mabuchi RS-385PH-16140	2
Motor Driver:	Arduino Motor Shield L298 Dual Full-Bridge Drive	1
IMU:	MPU-6050 (GY-521 Breakout Board)	1
Encoders:	Single Channel incremental encoder	2

used as Power-Width Modulation (PWM) outputs. The digital input pins will be used for DC motor control, and interrupts for the Inertial Measurement Unit (IMU) and single-channel incremental encoders. The board also has 6 analog inputs pins, two of which will be used for reading IMU values. The board has 32KB flash memory for storing code, and has a clock speed of 16MHz, which will be sufficient for this project. [3]

The board was initially meant to be attached to the pendulum, but due to weight issues was decided to be left external, as discussed in the next section.

5.1.2 Motors and Motor Drivers

The motors used in this project are the Mabuchi RS-385PH-16140 shown in figure 5.2. It is a carbon-brush DC motor with a maximum speed of 8700 revolutions per minute (RPM). The operating voltage of the motor 12-30V and has a nominal voltage of 24V.[12]

The motors were chosen for their low weight of 82g, and the fact that they came with single-channel incremental encoders attached.

The motor torque can be modeled as:

$$\tau = K_T I_A \tag{5.1}$$

where I_A is the armature current and K_T is the motor torque constant.

To drive the motors the Arduino Motor Shield shown in figure 5.3 was

Figure 5.1: The Arduino UNO development board [3]

chosen. It is based on the L298P dual full-bridge driver and can run two DC motors at up to 2A each.[19] The Arduino Motor Shield has two channels and takes three inputs on each channel, PWM (Pulse-Width Modulation) to control voltage supplied to the motors, DIR (direction) to control the direction of the motor, and Brake, which will effectively brake the motors. There is also one output per channel for current sensing, which lets you measure the current going through the DC motor.[2]

The Arduino Motor Shield was chosen due to its ease of use, as it fits on top of the Arduino UNO board, its low cost, and because it can drive two DC motors, which is what this project uses.

A mistake were made when choosing the Arduino Motor Shield. The choice was made based on the L298P dual full-bridge motor driver's data sheet. The driver is rated for up to 48 volts, and were thought to be a good choice, but it was later discovered that the Arduino Motor Shield is only rated for a maximum of 18 volts. Therefore the motors will not perform optimally.

Figure 5.2: The Mabuchi RS-385PH-16140 [13]

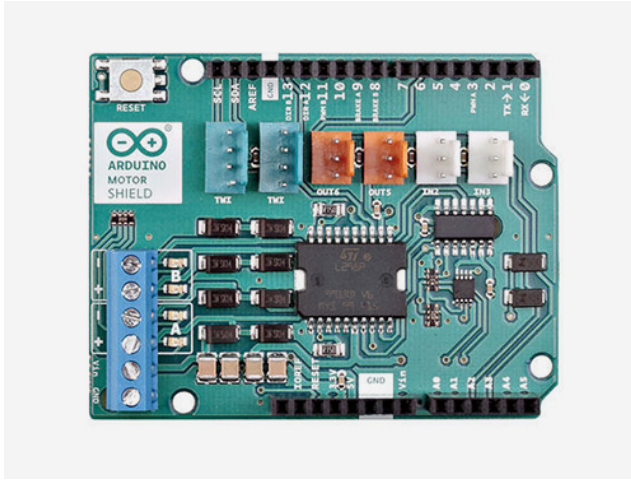


Figure 5.3: The Arduino Motor Shield [2]

5.1.3 Sensors

In order to control the two-axis reaction wheel inverted pendulum, we need to measure some state variables. The goal of the thesis is to stabilize the pendulum in its upright equilibrium point while the speed of the reaction wheel goes to zero, thus the angle of the pendulum in two directions needs to be measured, as well as the speed of the reaction wheels.

The DC motors came with single-channel incremental encoders attached, making it possible to calculate the angular velocity of the motors.

As for the pendulum body an IMU with 6 degrees of freedom is chosen.

Encoders

The DC motors chosen for this project came with single-channel incremental encoders attached. The encoders have a code track which have a set number of track positions that are sensed by the encoder. For each track position, the encoder sends a pulse to the Arduino. The Arduino counts the pulses and calculates the motor speed from the time it takes to reach a full revolution.

Some setbacks of the single-channel incremental encoder is the possibility of pulses not being registered due to interference or misreading, and that it does not provide the direction of movement. The direction of the reaction wheels must therefore be handled in code.

IMU

The velocity and acceleration of the system are measured by an Inertial Measurement Unit, or IMU for short. In this project the GY-521 breakout board, shown in figure 5.4, which is based on the InvenSense MPU-6050 sensor [11], was chosen. The MPU-6050 allows measurements in 6 degrees of freedom (DOF). It is a widely available sensor and has a very low price. The MPU-6050 contains a triple-axis MicroElectroMechanical (MEMS) accelerometer and a triple-axis MEMS gyroscope. It is very accurate , due to its 16-bits analog to digital conversion hardware for each channel. The sensor uses Inter-Integrated Circuit (I^2C) protocol to communicate with the development board.[4]

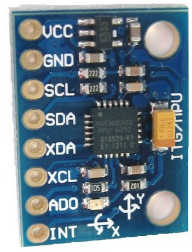


Figure 5.4: The GY-521 Breakout Board, MPU-6050 [4]

5.1.4 Reaction Wheels

The reaction wheels were designed in Fusion 360 and 3D printed to keep the main body weight down, and designed with the option to evenly add weight to the outermost edge of the wheels. The design of the reaction wheel can be seen in figure 5.5

The moment of inertia of a rigid body about an axis is defined by:

$$I = \sum_i m_i r_i^2 \tag{5.2}$$

where m_i is the point mass at a distance l_i from the rotational axis.

By moving all the weight to the edge of the reaction wheel, and simplifying, the moment of inertia can be approximated to:

$$I \approx mr^2 \tag{5.3}$$

where m is the total mass of the reaction wheel and r is the radius. This allows for minimizing the mass of the reaction wheel, while increasing the moment of inertia.

5.2 Hardware Implementation

In this section the hardware implementation of the two-axis reaction wheel inverted pendulum (RWIP) will be discussed. First we will look at the original plan and goals of the system, then we will go through the changes that were made and why.

Figure 5.5: The Reaction Wheel designed in Fusion 360

5.2.1 Build Plan and Goals

The goal of the build was to create a standalone system, that was not attached to anything and could easily be moved and taken around. The motors and reaction wheels were to be mounted on top of a rod along with the IMU unit, using 3D printed parts designed in Fusion 360.

The development board and motor shield would be mounted on a 3D printed standoff, with a battery pack for driving the system. This would be mounted in such a way as to move the center of mass to the optimal position. The motor mounts would also be adjustable for this reason.

Due to the low personal budget of the project, the motors chosen would not allow for the system to be built as planned, because of the total weight.

5.2.2 Final Design

With the components chosen, the build plan had to be modified. The overall weight of the system as originally planned, was too high. Due to this, the development board, motor driver shield and power supply was moved off the system, and wires run between the RWIP and the development board. The final build can be seen in figure 5.6.

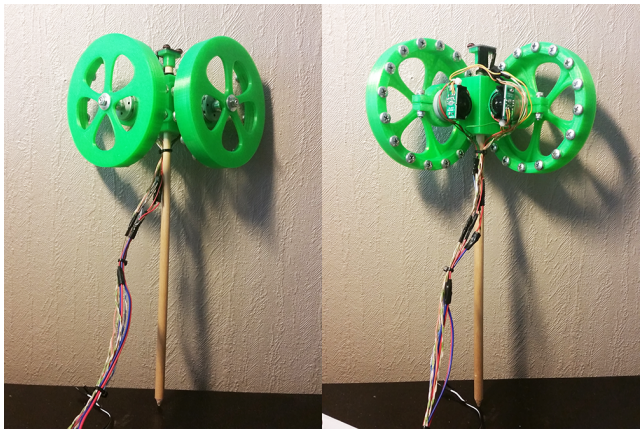


Figure 5.6: The final design of the two axis reaction wheel inverted pendulum

The wires hanging from the pendulum creates a disturbance on the system and makes it more difficult to control, as it shifts the center of mass and adds drag on the system that changes with the movement of the pendulum.

The final design draws inspiration from the design of Rouleau [14].

5.3 Software Implementation

This section will discuss the software implementation of the system.

The two-axis RWIP was programmed using Arduinos own Integrated Development Environment (IDE), which supports C/C++ programming language.

5.3.1 Sensors

The encoders were implemented in the system by using interrupt functions. Each time a change of signal on the interrupt pin is registered, a function is called. This function counts pulses until it reaches one full revolution, saves the time and time difference since last revolution, and resets the counter. When the speed measurement is needed, a function is called that calculates the motor speed, $\dot{\theta}_w$ based on the current time

of one revolution.

In order to decide the direction of the motor, the direction set by the controller is compared to the rate of change in angular velocity. If the Angular velocity is decreasing, and has become sufficiently small yet, the reaction wheel has not changed direction.

The IMU measurements are read using the MPU-6050 library written by Jeff Rowberg [16] and uses code from an example file in that library to read values. The IMU measurements are then filtered to get rid of some of the noise, and give the current values of θ_p and $\dot{\theta}_p$. These values are then fed to the controllers, to calculate the desired motor output.

5.3.2 Controllers

The PD+P controller is first implemented with the gain values from the simulations, and then tuned through several iterations.

The controllers are programmed using the reference value and the measured state variables, and the controller gain values. The output value of the controller is constrained to map the limits of the PWM signal boundaries, and this final value is then sent to the DC motor that is to be controlled.

Moving Reference

In order to compensate for center of mass not being at $\theta_p = 0$, a moving zero reference were implemented. For each iteration of the system calculations, the zero reference is updated with 0.5% of the current θ_p value.

5.4 Discussion

The experimental model was built as shown in the final design and software was implemented as described. Fusion 360 had to be learned in order to 3D model the parts needed for this project and took good amount of time to master, and the parts were successfully designed and 3D printed after multiple iterations.

A few setbacks were encountered during the build. Because of the limited

personal budget, most of the components were ordered from eBay and had a long delivery time. During the project one of the motors failed and a new one had to be ordered, this new motor did not arrive until a week before the due date of this report, giving limited time to properly test the system and to get it to stabilize.
Another setback was the choice of motor shield, which ultimately limited the performance of the DC motors. Had this mistake been discovered earlier, a different motor driver would have been chosen.

Chapter 6

Experimental Results and Discussion

In this chapter we will go through the experimental testing of the two-axis reaction wheel inverted pendulum described in the previous chapter.

First we will look at the outline of the tests to be performed. Thereafter we will look at the data from the experiments and show the results. And lastly we will discuss the result and see how the system performed according to expectations, pointing out what did not work and what did, and why.

6.1 Experimental testing

This section is organized in the following way:

- Section 6.1.1: Case 1: Functionality Tests
- Section 6.1.2: Case 2: Balancing

6.1.1 Case 1: Functionality Tests Sensors

In this case study we will look at the sensors of the system and run tests to ensure that they work properly and as expected. This is to make sure we get proper feedback from the system, so that we can apply control.

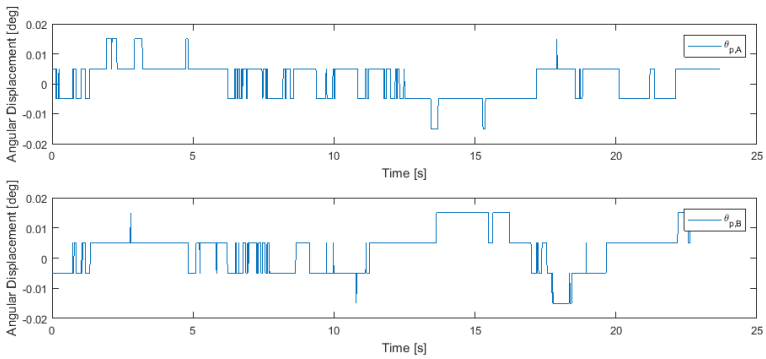


Figure 6.1: IMU measurements of the pendulum body's angular position, when still

The Inertial Measurement Unit (IMU) should provide accurate measurements of the angular position of the pendulum body, and with a limited amount of measurement noise. The θ_p values are filtered and calculated from the raw data provided by the IMU.

To test the encoders, the motors were driven with different PWM values to show that the encoders would be able to calculate speed. Rapid changes both in PWM and direction were added to show that the encoders were coded properly and would display the correct direction.

Results

The IMU were tested by holding the IMU still, shown in figure 6.1, and by tilting the pendulum body about both axes as shown in figure 6.2. We can here see that the IMU will accurately determine the angular position of the pendulum with a measurement noise of $\pm 0.015 deg$. The measurement noise shown in figure 6.1 is small enough as to not affect the control of the system too much.

We can see from figure 6.3 and 6.4, by comparing the plots for angular velocity and PWM setpoints, that the encoders are able to successfully determine speed and direction of the motors. The speed values and change in speed values provided by the encoders performs sufficiently well to be used as measurement feedback.

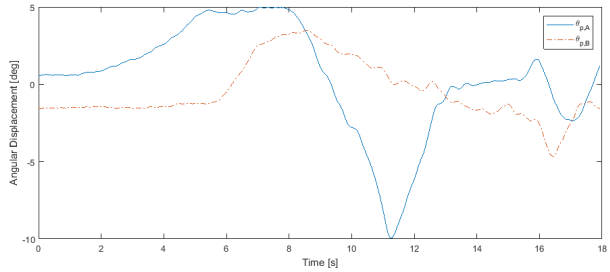


Figure 6.2: IMU measurements of the pendulum body's angular position, when rotated

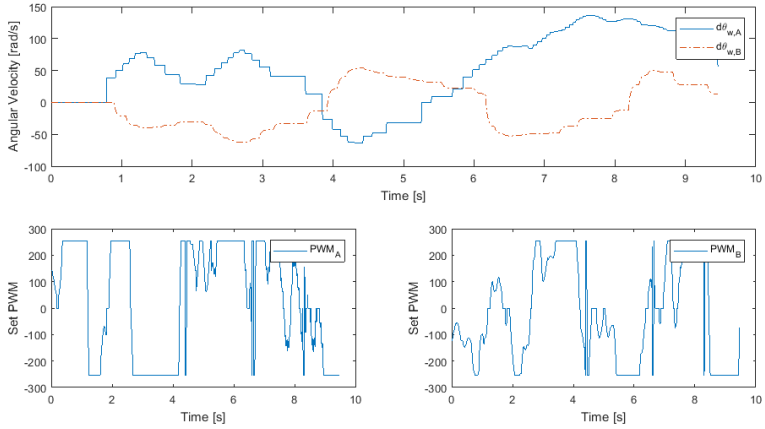


Figure 6.3: Speed measurement of DC motors, and the PWM values sent to the motors

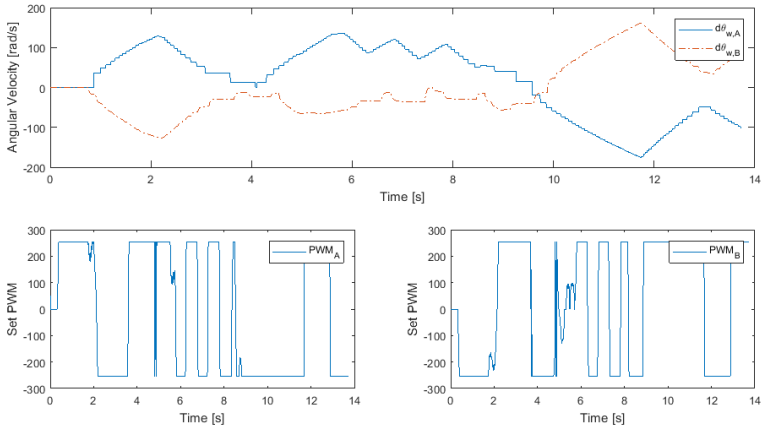


Figure 6.4: Speed measurement of DC motors, and the PWM values sent to the motors

6.1.2 Case 2: Balancing

In this case study we will implement controllers to the system and try to balancing the two-axis reaction wheel inverted pendulum with no added disturbance. First the values used in simulations will be used, and then the values will be tuned from there through multiple iterations.

Unfortunately the system were never stabilized and no final, working values for the PD+P controller were found.

Results

The initial controller gain values did not manage to balance the system and provided no usable plots, as it fell straight over.

The controller values were attempted tuned as best as could be, but as seen from figures 6.5 and 6.6, the pendulum oscillates out of control after a short amount of time. In the end the system were not able to stabilize. Figures 6.7 and 6.8 show the best settings that were found.

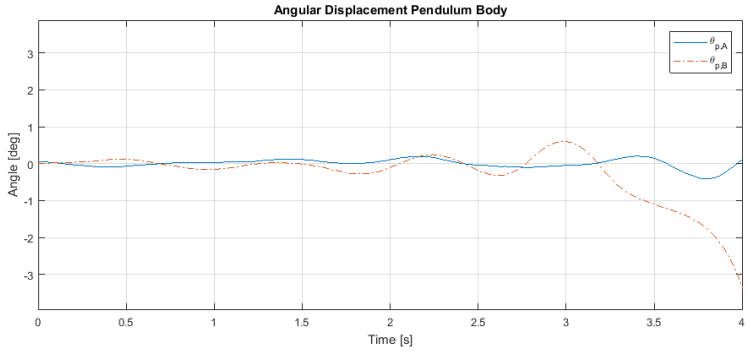


Figure 6.5: State trajectories of θ_p with applied control, $K_p = 670, K_d = 23, K_{p,w} = 0.8$

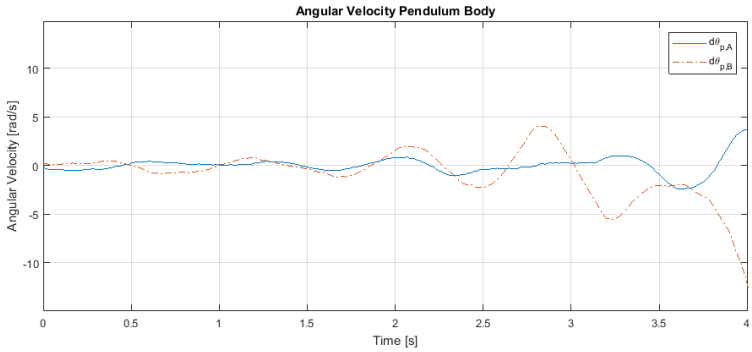


Figure 6.6: State trajectories of $\dot{\theta}_p$ with applied control, $K_p = 670, K_d = 23, K_{p,w} = 0.8$

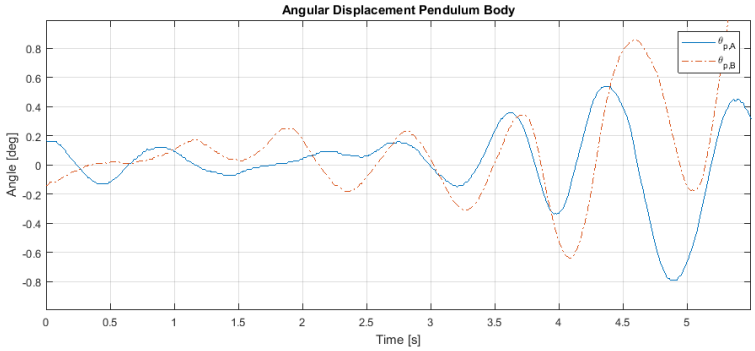


Figure 6.7: State trajectories of θ_p with applied control, $K_p = 600, K_d = 30, K_{p,w} = 2$

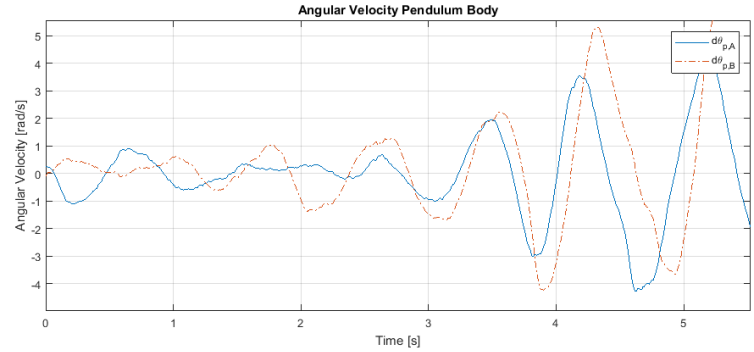


Figure 6.8: State trajectories of θ_p with applied control, $K_p = 600, K_d = 30, K_{p,w} = 2$

6.2 Discussion

In this chapter the system was tested, doing experiments with feedback measurements and balancing control.

First we looked at the feedback measurements of the IMU which showed that the MPU-6050 sensor provides sufficiently accurate values to use in feedback control. The filtered IMU values, shows low measurement noise, and will not affect the performance of the controller.

Secondly the feedback measurements of the encoders were tested. Here we saw that the encoders were able to determine both speed and direction of the motors, and that they perform sufficiently well for the use in feedback control.

Lastly the completed experimental system was tested with different controller gain values in order to make it balance at its upright equilibrium. Unfortunately, through countless tests and different gain values, the system were never stabilized. This is partly due to running out of time, but mostly due to the wrong choice in hardware components as discussed in the previous chapter.

Due to the voltage limitation on the motor shield, the motors were under-powered and is the main cause of the system not being able to balance. In order to provide enough torque to move the pendulum body, the voltage sent to the motors almost had to be maxed out. This gave little room for control.

Chapter 7

Conclusion and Further Work

7.1 Conclusion

In this master thesis a model for the reaction wheel inverted pendulum was derived using Euler-Lagrange's equations of motion. The model was implemented in MATLAB and Simulink to show the behavior of a two-axis reaction wheel inverted pendulum. Different controllers were implemented in order to stabilize the system about its upright equilibrium, and to compare their performances.

An experimental model was developed and built, including software and hardware design. Sensors for measuring the system states were implemented and tested, and proven to be sufficient. The experimental model was tested with a Proportional-Derivative controller on the pendulum's angular position and a P controller on the reaction wheel's angular velocity. Unfortunately the experimental model failed to stabilize about the upright equilibrium.

7.2 Further Work

It has been shown that the experimental model, as is, does not stabilize at the upright equilibrium, and thus the most obvious remaining work is to change out some of the components, to attain this goal, mainly the

motor shield. The motor shield should be able to handle the nominal voltage of the DC motors.

Further the DC motors chosen should work if the motor shield was to be replaced, but it would be favorable to exchange the motors for ones with a better torque to weight ratio. The encoders could also be switched out for two-channel encoders that can also detect the direction of rotation. The reaction wheels could also be improved by being redesigned and milled from a metal to increase precision and moment of inertia.

A short term goal would then be to get the experimental model to balance about the upright equilibrium. Once this goal is attained, tests can be performed adding external impulse forces on the system, and on performance when changing the center of mass.

Bibliography

[1] Brian D. O. Anderson and John B. Moore. *Optimal Control - Linear Quadratis Methods*. 1989.

[2] Arduino. *Arduino Motor Shield*. URL: `https://www.arduino.cc/en/Main/ArduinoMotorShieldR3`.

[3] Arduino. *Arduino UNO & Genuino UNO*. URL: `https://www.arduino.cc/en/main/arduinoBoardUno`.

[4] Arduino. *MPU-6050 Accelerometer + Gyro*. URL: `http://playground.arduino.cc/Main/MPU-6050`.

[5] Jens G. Balchen, Trond Andresen, and Bjarne A. Foss. *Reguleringsteknikk*. 2003.

[6] Bill Dunford. *Basics of Space Flight, Attitude and Articulation Control Subsystems (AACS)*. Ed. by Phillips Davis. URL: `https://solarsystem.nasa.gov/basics/chapter11-2`.

[7] Olav Egeland and Jan Tommy Gravdahl. *Modeling and Simulation for Automatic Control*. 2002.

[8] M. Gajamohan, M. Muehlenbach, and R: D'Andrea. "Nonlinear Analysis and Control of a Reaction Wheel-based 3D Inverted Pendulum". In: *52nd IEEE Conference on Decision and Control*. 2013.

[9] M. Gajamohan et al. "The Cubli: A cube that can jump up and balance". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2012.

[10] E.H. Hauge and J.A. Stvneng. *Grunnleggende fysikk -klassisk mekanikk og varmelre*. 2009.

[11] InvenSense. *MPU-6000 and MPU-6050 Product Sepecification Revision 3.4*. 2013. URL: https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf.

[12] MabuchiMotor. *RS-385PH*. URL: https://www.jameco.com/Jameco/Products/ProdDS/2120461.pdf.

[13] MabuchiMotors. *RS-385PH*. URL: https://product.mabuchi-motor.com/detail.html?id=101.

[14] Mike Rouleau. *Self Balancing Stick - [Dual Axis Reaction Wheel Inverted Pendulum]*. 2015. URL: https://www.reddit.com/r/arduino/comments/3n0g8p/self_balancing_stick_dual_axis_reaction_wheel/.

[15] Mike Rouleau. *Self Balancing Stick - Dual Axis Reaction Wheel Inverted Pendulum*. 2015. URL: https://www.youtube.com/watch?v=woCdjbsjbPg.

[16] Jeff Rowberg. *I2C Devise Library - MPU-6050*. 2012. URL: https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050.

[17] NASA SPINOFF. *Reaction/Momentum Wheel*. URL: https://spinoff.nasa.gov/spinoff1997/t3.html.

[18] M. W. Spong, P. Corke, and R. Lozano. "Nonlinear control of the Inertia Wheel Pendulum". In: *Automatica* (1999).

[19] STMicrooelectronics. *L298 - Dual Full-Bridge Driver, Data Sheet*. 2000. URL: http://www.st.com/content/ccc/resource/technical/document/datasheet/82/cc/3f/39/0a/29/4d/f0/CD00000240.pdf/files/CD00000240.pdf/jcr:content/translations/en.CD00000240.pdf.

[20] J. G. Ziegler and N. B. Nichols. "Optimum Settings for Automatic Controllers". In: *Transactions of the ASME* (1942).

Appendix A

Digital Appendix

The digital appendix contains the following files:

- Arduino Code for the Two-Axis Reaction Wheel Pendulum.
- Design Files for Parts Designed in Fusion360.
- MATLAB and Simulink Files, Including Model of the Two-Axis RWIP.