Design of a Linear Quadratic Gaussian Control System for a Thrust Vector Controlled Rocket

a project presented to
The Faculty of the Department of Aerospace Engineering
San José State University

In Fulfillment
Of the Requirements for the Degree
Master of Science in Aerospace Engineering

by

Alex Ganbold
May 2023

approved by

Dr. Jeanine Hunter
Faculty Advisor

Abstract

Design of a Linear Quadratic Gaussian Control System for a Thrust Vector Controlled Rocket

Alex Ganbold

There are several different types of control methods that can be used for linear and non-linear systems. These control methods require simple to complex controllers. In this project, the pitch stability of a finless rocket is analyzed by obtaining the state space model and examining the open and the closed loop response of different control methods. Also, the response of a simple, yet robust Proportional, Integral, Derivative (PID) controller is evaluated against the response of a Linear Quadratic Regulator (LQR). Due to the limitation of real-life applications and cases, a Kalman Filter (optimal estimator), was developed to fully observe and obtain the necessary state variables. Ultimately, the LQG and Kalman Filter results and the gains will be combined to obtain the Linear Quadratic Gaussian (LQG) controller response. Each section will define, derive, and implement the necessary functions into MATLAB and Simulink for optimal responses.

Acknowledgements

I would like to thank Professors Hunter and Professor Lu for their guidance throughout the course of this project. The courses on controls and navigation taught by both professors became the foundation of my interest in GNC and the beginning of my career path. I really appreciate the patience and the knowledge that was imparted to me and the project. I would also like to thank my family and my friends for their support.

Table of Contents

List of Figures

Symbols

Symbols	Definition	Units (SI)
I	Moment of inertia matrix	$kg * m^2$
F	Linear force	N
ω	Rotational velocity	rad/s
v	Linear velocity	m/s
a	Linear acceleration	m/s^2
m	Mass	g
A	State matrix gain	$---$
B	Input matrix gain	$---$
C	Output Matrix gain	$---$
D	Feed-forward matrix gain	$---$
x	State matrix	$---$
$\hat{x}$	Estimated state matrix	$---$
u	Input matrix	$---$
w	Process noise	$---$
v	Measurement noise	$---$
P	Covariance matrix	$---$
K	Gain matrix	$---$
n	System rank	$---$
L	Estimator gain matrix	$---$
e	Error	$depends$
i	Symbol mainly used with x axis	$---$
T	Torque	$N * m$
r	Distance from the center of mass to applied torque (when used in Torque equation)	m
J	Cost function	$---$
M	Mach number	$---$
h	Altitude	m
H	Angular Momentum	$kg * m^2/s$
Greek Symbols		
θ	Pitch angle	rad
$\dot{\theta}$	Pitch angle rate	rad/s
$\dot{Z}$	Lateral drift	m/s
α	Angle of attack	rad
δ	Angle of flap	rad
φ	Euler angle rotation	$---$
ϵ	Euler angle rotation	$---$
Subscripts		
$\#^*$	Equilibrium points, # can be any variable	$---$

$()_e$	Parameter in inertial frame	$---$
$()_b$	Parameter in body frame	$---$
$()_\alpha$	Parameter that is affected by angle of attack	$---$
$()_\delta$	Parameter that is affected by elevator pitch	$---$
$()_{c_g}$	Parameter that is connected with center of gravity	$---$
$()_{c_p}$	Parameter that is connected with center of pressure	$---$
Acronyms		
GNC	Guidance, Navigation and Control	$---$
TVC	Thrust Vector Control	$---$
NASA	National Aeronautics and Space Administration	$---$
VTOL	Vertical Take-Off and Landing	$---$
PID	Proportional Integral Derivative	$---$
LQR	Linear Quadratic Regulator	$---$
LQG	Linear Quadratic Gaussian	$---$
KF	Kalman Filter	$---$
ECI	Earth Centered Inertial	$---$
DCM	Direct Cosine Matrix	$---$
ODE	Ordinary Differential Equation	$---$

Chapter 1: Introduction

1.1 Motivation

Guidance Navigation and Control (GNC) is a crucial subsystem for any aerial vehicle to operate as desired. Reliable guidance and control for a rocket or a missile is a necessity for its maximum potential. For many years, aerodynamic control surfaces were used as the medium to actuate the vehicle. Surfaces such as fins or canards did an excellent job of controlling the attitude of a rocket and reaching its destination. For many of the benefits of control surfaces, there are also several shortcomings that need to be addressed. Some of the shortcomings include the inability of control surfaces to effectively function at altitudes with little to no atmosphere and requiring different surfaces for different pitch angle corrections. To overcome these shortcomings, thrust vector control (TVC) was invented and optimized for both rocket and aircraft use. With less dependency on aerodynamic properties, a vehicle with a TVC system can maneuver without the absence of an atmosphere and is able to provide control moments at high angles of attack. With the first usage of exhaust vanes on V2 missiles and the first testing of NASA's X-31 in 1990, several different types of TVC systems were invented, including vertical take or landing (VTOL) capable aircraft.

For most of the aerospace industry, classical control theory has been implemented for decades as it provides simple and working control system to fulfill the necessary objectives. A control theory such as PID is simple and great to use, even on non-linear dynamic models. However, there are several modern control theories that also optimize the best gains and compensate for any possible inadequacies that either the software or the hardware might have.

1.2 Literature Review

Ever since Robert Goddard started the age of space with his revolutionary works on propulsion and rocketry, humans have developed various types of propulsion systems to accomplish the diverse needs of spaceflight. From the use of solid propellant rockets, dating back to ancient China, to liquid bi-propellant rockets and to even electric and jet engines, there are applications of propulsion in every aspect of human society. With the increased need of complex propulsive vehicles, there is also an increased need to control them. For rocketry, there exists few physical control systems including fins, canards, wings and/or attitude control using thrust vectoring. There are benefits and drawbacks to each configuration, thus they are often grouped together to optimize the control system. One of the most unconventional yet very crucial systems is thrust vectoring. The most important advantage of a thrust vector control is its use in space where there is no atmosphere for aerodynamic control surfaces. Even at low atmospheric pressures, aerodynamic surfaces lose their efficacy at an alarming rate; so, it is increasingly crucial to implement TVC systems in most rockets or missiles.

There are several physical ways to employ thrust vectoring, including: engine gimbaling, reactive fluid injection, vernier thrusters, and exhaust vanes or jet vanes.

The first uses of TVC were implemented during WWII on V-2 rockets by Wernher Von Braun. V-2 rockets utilized four static fins for stability and graphite jet vanes for TVC. Since then, there have been numerous implementations of TVC design in every aeronautical and astronautical application. There are also several control system methods that can be implemented to actuate the necessary physical hardware.

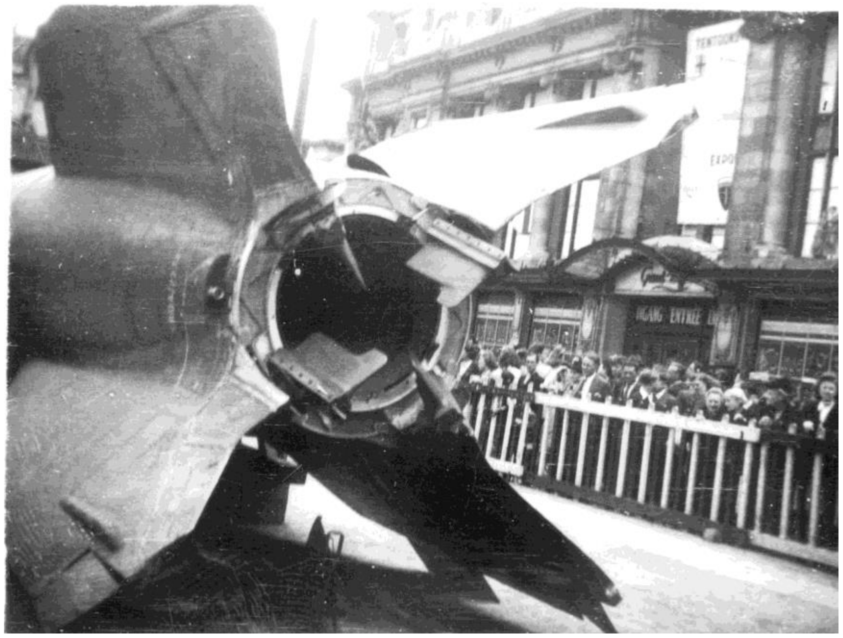


Figure 1: V-2 Jet vanes [4]

Although there were no defensive measures against the V-2 at the time, due to the inaccuracy of the missiles, the program proved to be very inefficient due to its cost of production.
The most common method, for rocketry and aircraft alike, is to gimbal the whole engine or the nozzles. By creating a torque on the center of gravity of the rocket, either yaw or pitch of the rocket can be manipulated through controlled inputs. Thrust gimbaling is illustrated in Figure 2.

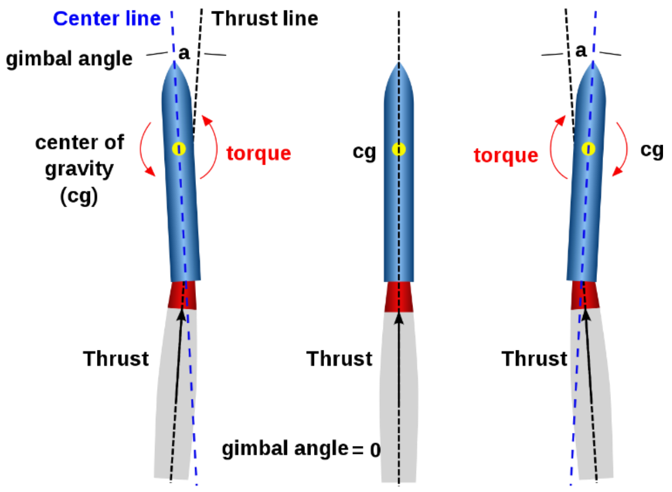


Figure 2: Engine Gimbaling [5]

1.3 Proposal

In this paper, a simple rocket with thrust vector control system will be analyzed for its attitude due to disturbance, thrust input and/or sensor noise using different types of optimal control theories. For its proven efforts in industry and its robust behavior, PID control will be first implemented for a given thrust angle. PID is a classical control theory that is simple, efficient, and very robust to use. With numerous modern control theories being designed and developed, PID control systems are still overwhelmingly preferred by the majority of engineers in the industry. The results from the classical theory will be compared to modern techniques including Linear Quadratic Regulator (LQR) and Linear Quadratic Gaussian (LQG). Linear Quadratic Regulator is an optimal control system method that finds the optimal gain and influences the input signal to correct error signal. With LQG, LQR relates to Kalman Filter theory and obtains the optimal controller gain with observer gain to fully estimate the optimal state of the model and the output.

1.4 Methodology

Using the equations of motion for a thrust vector-controlled rocket, a linear and continuous state space model will be created. Using aerodynamic characteristics of a known rocket, the necessary matrices for a linear rocket are designed. With fully built state space, various types of control system theories can be implemented to observe the attitude and stability changes. With classical control theory, the PID method will be implemented on the pitch angle disturbance. Since the rocket is inherently unstable with no static stability fins, it is necessary to tune the P, I, and D gains to efficiently reach a desired attitude. Next, LQR will be designed and implemented using MATLAB and Simulink to find the necessary Q and R gains for the cost function and obtain the optimal gain. Since the plant is fully controllable and fully observable, a Kalman Filter will be designed to estimate the optimal rocket state from its output signals. Using the separation principle, LQR and Kalman Filter can be combined to find the optimal response.

Chapter 2: Rocket Model

2.1 Frame of Reference

For any object in space, there exist two frames of references to describe it. There are earth and body reference frames, with each reference frame fixed to earth itself and the body of the object. The earth centered inertial (ECI) frame has the origin at the middle with each directional axes perpendicular to each other, with Z locating at the geographical north pole. The ECI frame can be seen in Figure 3:

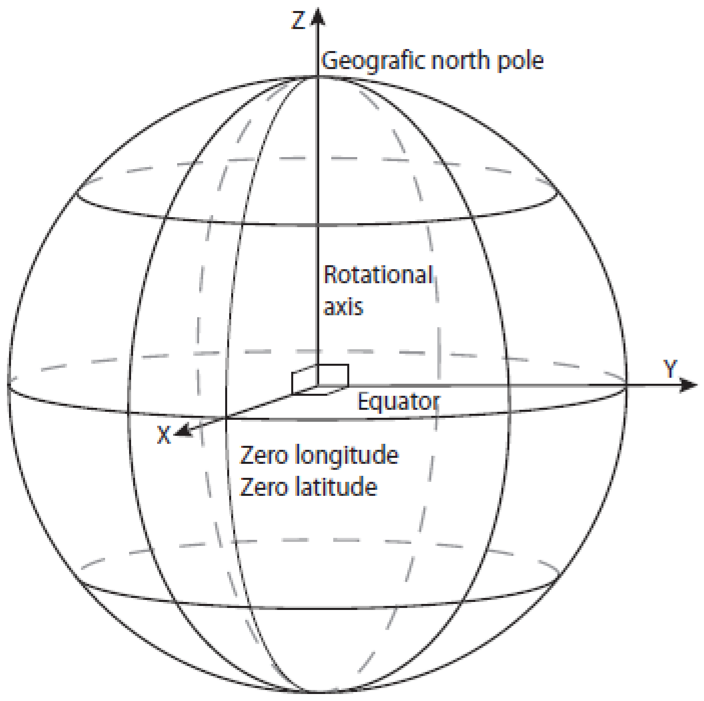


Figure 3: Earth centered inertial reference frame [1]

ECI directional vectors are often described as $\hat{\imath}_e$, $\hat{y}_e$, $\hat{z}_e$ and can be used to describe a position of an object compared to the earth. The body refence frame, however, emphasizes the directions from the perspective of the object with t = 0 as seen from the object itself. Similar to ECI, body frames are described as $\hat{\imath}_b$, $\hat{y}_b$, $\hat{z}_b$ and is illustrated in Figure 4.

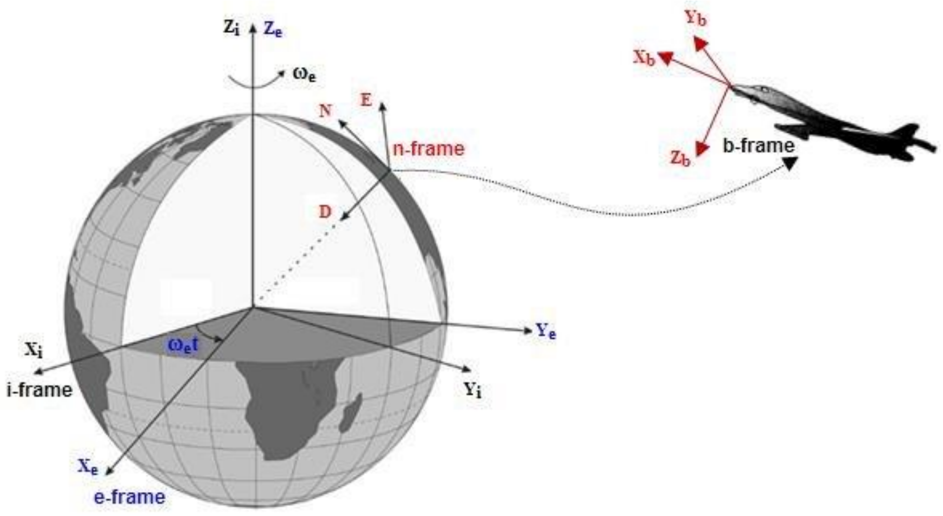


Figure 4: ECI and body frame [2]

2.2 Equations of Motion

In any control design process, whether it is aerial or not, the modeling of the plant is a necessity. Any dynamic object can be defined by a set of mathematical equations that will predict its response to any disturbance or inputs. Equations of motion of a system describe the full non-linear characteristics of the plant in time domain. For any rigid body in space, there are three ways to describe its motion: translational, rotational and with Euler angles.
Translational motion, as the name states, looks at the physical location and the movements of a plant from ECI frame and can be used to predict the location at given time.

Table 1: Coefficients symbols

	Roll	Pitch	Yaw
Angular Rates	p	q	r
Velocity	u	v	w
Aerodynamic Force	F_{A_x}	F_{A_y}	F_{A_z}
Aerodynamic Force Coefficients	C_D	C_Y	C_L
Aerodynamic Moment Coefficients	C_L	C_M	C_N
Angular Rates	ω_x	ω_y	ω_z

Since rocket body is symmetric along XY and XZ planes, the products of inertial: Ixy, Ixz and Iyz, become zero.
For simplicity and since the rocket will be well within the atmosphere, the aerodynamic coefficients will be constant, and the air will be incompressible. Rocket dynamics can be divided into six degrees of freedom: three translational and three rotational degrees of freedom with additional Euler angles for angular representation.

Translational Motion

Considering Newton's second law of motion:

$$\sum F = ma \tag{2.1}$$

$$\sum T = \frac{d}{dt}(r \text{ x } mV) \tag{2.2}$$

with the net force, *F*, on the rocket and net torque, *T*, as foundational equations for the translational movement, the final equations can be derived:

$$\begin{aligned}
\dot{u} &= \frac{F_{A_x} + F_{P_x} + F_{g_x}}{m} + rv - qw \\
\dot{v} &= \frac{F_{A_y} + F_{P_y} + F_{g_y}}{m} + pw - ru \\
\dot{w} &= \frac{F_{A_z} + F_{P_z} + F_{g_z}}{m} + qu - pv \\
\dot{p} &= \frac{L_A + F_P - qr(I_z - I_y)}{I_x}, rad/s^2 \\
\dot{q} &= \frac{M_A + M_P - rp(I_x - I_z)}{I_y}, rad/s^2 \\
\dot{r} &= \frac{L_A + F_P + qr(I_z - I_y)}{I_x}, rad/s^2
\end{aligned} \tag{2.3}$$

(The full derivation of the EOM is found in Appendix 1)

For dynamic systems, it is necessary to describe the motion using the correct frame of reference. The Euler angles, $\theta, \varphi, and\ \epsilon$ are used to convert the frames of reference through series of rotations. As shown in Figure 5, there can be multiple different types of rotations to reach the desired reference.

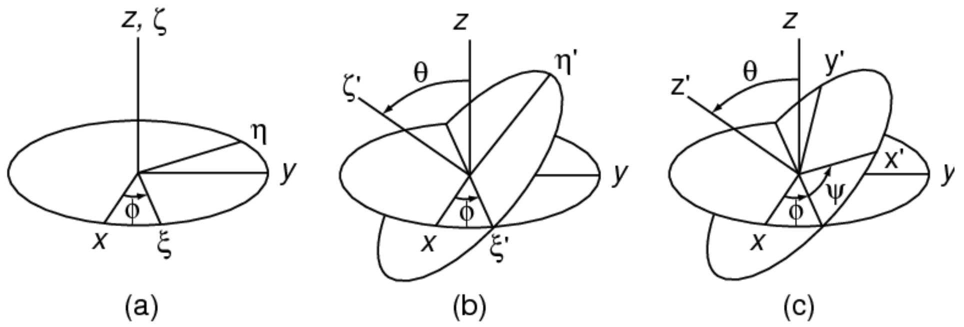


Figure 5: Euler angle rotations [16]

The full rotation follows the sequence of,

$$a \rightarrow a' \rightarrow a'' \rightarrow b \tag{2.4}$$

and there exists a matrix, called the Direct Cosine Matrix (DCM), that represents the relation between axes and used as a conversion matrix between vectors. The DCM follows the equation:

$$c_{ij} = b_j \cdot a_j \tag{2.5}$$

with each element representing different rotation:

$$c^{b/a} = \begin{bmatrix} b_1 \cdot a_1 & b_1 \cdot a_2 & b_1 \cdot a_3 \\ b_2 \cdot a_1 & b_2 \cdot a_2 & b_2 \cdot a_3 \\ b_3 \cdot a_1 & b_3 \cdot a_2 & b_3 \cdot a_3 \end{bmatrix} \tag{2.6}$$

The DCM matrix rotations can be simplified by the notation of $c^{b/a}$, $a \rightarrow b$, or $c^{a/b}$, $b \rightarrow a$.

2.3 Linearization

Although equations of motion describe the full dynamics of a system, the DCM describes non-linear dynamics that restrict the ability to implement most of the classical or modern control theories. A linear time-invariant system is often the pre-requisite for any control design as most methods were developed for a linearized model.
To linearize any model, the equations of motion must be rewritten in the form of state-variable, as shown in Eq. 2.7.

$$\begin{gathered} \dot{x_1} = f_1(x_1, \dots, x_n, u) \\ \dot{x_2} = f_2(x_2, \dots, x_n, u) \\ \vdots \\ \dot{x_n} = f_n(x_1, \dots, x_n, u \\ y = h(x_1, \dots, x_n, u) \end{gathered} \tag{2.7}$$

Eq. 2.7 shows the necessary state variables, input scalars and the system output in a clear and concise manner. The equations take the form of:

$$f(x, u) = \begin{bmatrix} f_1(x_1, \dots, x_n, u) \\ f_2(x_1, \dots, x_n, u) \\ \vdots \\ f_n(x_1, \dots, x_n, u) \end{bmatrix} \tag{2.8}$$

Since the end goal of the linearization is to turn the non-linear EOMs into linear state space form, the system must be trimmer or reach an equilibrium point. The equilibrium points, $x^*[x_1{}^*, \dots, x_n{}^*]^T$, are the parameters when the system or the rocket is in their designated equilibrium state (i.e. flying straight and level in the longitudinal plane). With assumed equilibrium points and denoting $\Delta x = x = x^*$, $\Delta u = u - u^*$ and $\Delta y = y - h(x^*, u^*)$, the state space can be formed using the new variations of $\Delta x, \Delta u\ and\ \Delta y$ from x, u and y.

By taking the partial derivatives of the function by each state variable and the inputs, the state space matrices are formed.

$$A = \left[\frac{\partial f}{\partial u}\right]_{x^*,u^*} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}({x_1}^*, \dots, {x_n}^*, u^*) & \cdots & \frac{\partial f_1}{\partial x_n}({x_1}^*, \dots, {x_n}^*, u^*) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}({x_1}^*, \dots, {x_n}^*, u^*) & \cdots & \frac{\partial f_n}{\partial x_n}({x_1}^*, \dots, {x_n}^*, u^*) \end{bmatrix}$$

$$B = \left[\frac{\partial f}{\partial u}\right]_{x^*,u^*} = \begin{bmatrix} \frac{\partial f_1}{\partial u}({x_1}^*, \dots, {x_n}^*, u^*) \\ \vdots \\ \frac{\partial f_n}{\partial u}({x_1}^*, \dots, {x_n}^*, u^*) \end{bmatrix} \tag{2.9}$$

$$C = \left[\frac{\partial h}{\partial u}\right]_{x^*,u^*} = \begin{bmatrix} \frac{\partial h}{\partial x_1}({x_1}^*, \dots, {x_n}^*, u^*) & \dots & \frac{\partial h}{\partial x_n}({x_1}^*, \dots, {x_n}^*, u^*) \end{bmatrix}$$

$$D = \left[\frac{\partial h}{\partial u}\right]_{x^*,u^*}$$

For the C matrix, a gain, often a gain of 1, is placed on the respective dimension in the matrix to output the response. In this case, if the pitch angle is the desired output response, then the C matrix would become [1 0 0]. The feedforward matrix, D, is often zero as the system and the control method is a feedback response method.

2.4 State Space Modeling and System Stability

State space modeling of any dynamic system provides a simple, matrix-based representation of the plant. With known aerodynamic characteristics of the vehicle, state space representation can become a linear, time-invariant model of the specific system that can fully represent the motion with any input. This paper will be using known aerodynamic characteristics of a rocket or use values from another paper for proof of concept.

$$\begin{aligned} \dot{\vec{x}} &= A\vec{x} + B\vec{u} \\ \vec{y} &= C\vec{x} + D\vec{u} \end{aligned} \tag{2.10}$$

State space model consists of state vector, $\vec{x}$, and matrix, **A**, input vector, $\vec{u}$, and matrix, **B**, the output vector, $\vec{y}$, and matrix **C** and the feedforward matrix, **D**. A simple block diagram of a state space is shown in Figure 6 below with all matrices and logical paths represented.

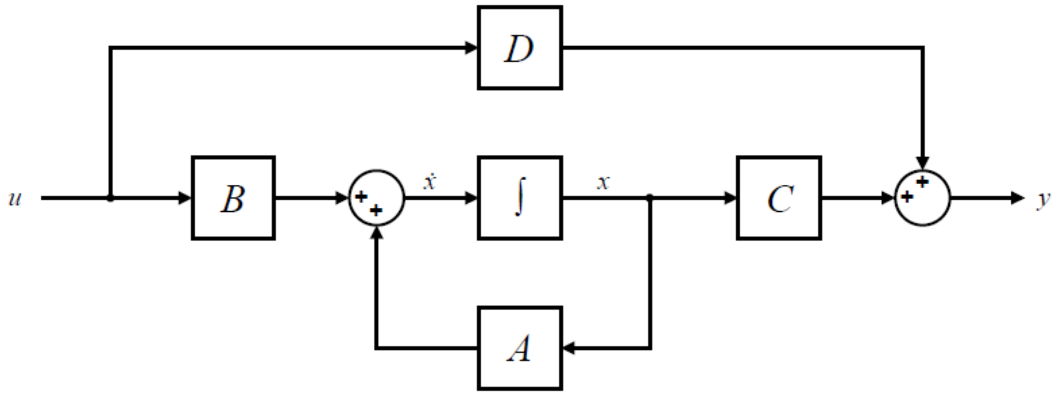


Figure 6: State space block diagram [20]

All the necessary states, such as position, rotation, or the rates of each variable, of a system are represented by the state vector and is manipulated by the input states. Although the state and input matrices are constant for the system, the C matrix is dependent on the desired output of the designer.

Using the aerodynamic characteristics of a known rocket (15), the state space model of the system is given in Equation 2.11.

$$\frac{d}{dt}\begin{bmatrix}\theta\\ \dot{\theta}\\ \dot{Z}\end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ M_\alpha & 0 & \frac{M_\alpha}{V} \\ \frac{(D-F-N_\alpha)}{m} & 0 & -\frac{N_\alpha}{mV}\end{bmatrix}\begin{bmatrix}\theta\\ \dot{\theta}\\ \dot{Z}\end{bmatrix} + \begin{bmatrix} 0 & 0 \\ M_\delta & M_\alpha \\ \frac{T_\delta}{m} & -\frac{N_\alpha}{m}\end{bmatrix}\begin{bmatrix}\delta \\ \alpha_w\end{bmatrix} \tag{2.11}$$

The reference parameters of the vehicle at 60 second time point are given in Table 2.

Table 2: Vehicle parameters [15]

Data	Value	Unit
m	567718	Kg
I_y	$296.4*10^6$	Kgm^2
T_o	10506450	N
T_δ	10506450	N
V	410.56	m/s
N_α	3056.34	kN/rad
M_α	0.3807	s^{-2}
M_δ	0.5726	s^{-2}
x_{cg}	16.21	m
x_{cp}	39.94	m
M	1.4	--
h	10.0	Km
D	903.3	kN

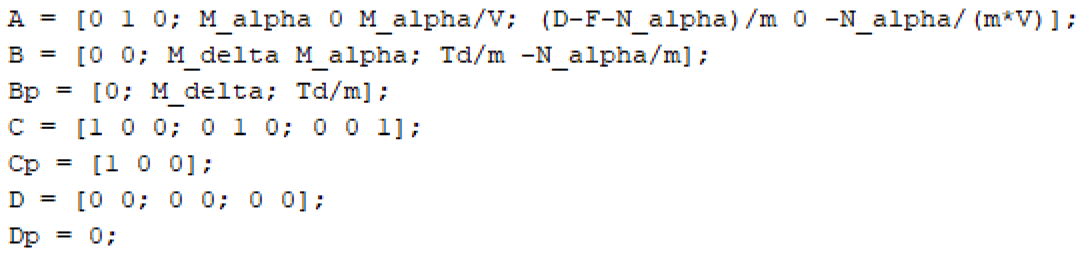
The state space model is for the whole rocket with all matrices provided. This paper will be focusing on the pitch attitude due to pitch angle input. The pitch state space equation is given in Eq. 2.12.

$$\frac{d}{dt}\begin{bmatrix}\theta \\ \dot{\theta} \\ \dot{Z}\end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ M_\alpha & 0 & \frac{M_\alpha}{V} \\ \frac{(D - F - N_\alpha)}{m} & 0 & -\frac{N_\alpha}{mV}\end{bmatrix}\begin{bmatrix}\theta \\ \dot{\theta} \\ \dot{Z}\end{bmatrix} + \begin{bmatrix} 0 \\ M_\delta \\ \frac{T_\delta}{m}\end{bmatrix}\delta \tag{2.12}$$

For the C matrix, since the pitch angle of the rocket is considered:

$$C = \begin{bmatrix}1 & 0 & 0\end{bmatrix} \tag{2.13}$$

For MATLAB and Simulink implementation:



```
A = [0 1 0; M_alpha 0 M_alpha/V; (D-F-N_alpha)/m 0 -N_alpha/(m*V)];
B = [0 0; M_delta M_alpha; Td/m -N_alpha/m];
Bp = [0; M_delta; Td/m];
C = [1 0 0; 0 1 0; 0 0 1];
Cp = [1 0 0];
D = [0 0; 0 0; 0 0];
Dp = 0;
```

Figure 7: MATLAB state space setup

For any system, it is necessary for the system to be controllable before any type of control system is designed for it. Similarly, it is also crucial for the system to be observable, meaning that the states of a system can be estimated from a simulated output behavior. If the ranks of the controllability and the observability matrices are the rank of N, then, it is said that the system is controllable and observable.

Controllability

$$rank\begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1}\end{bmatrix} = n \tag{2.14}$$

Observability

$$rank\begin{bmatrix}B & AB & A^2B & \cdots & A^{n-1}B\end{bmatrix} = n \tag{2.15}$$

In MATLAB, both matrices and ranks can be easily checked with simple code.

$$rank\left(ctrb\left(A, B_p\right)\right) \tag{2.16}$$
$$rank\left(obsv\left(A, C_p\right)\right) \tag{2.17}$$

Using the given code, the ranks of both matrices were found to be 3. This proves that the given system is both controllable and observable.
Since the rocket does not possess any control surfaces that provide inherent static stability such as fins or wings, it is expected that the attitude of the rocket would be extremely unstable. To check for the inherent behavior of the rocket without any active control system, the Open Loop response can be simulated.

In Simulink environment, an open loop model is a simple feed forward path with external input attached.

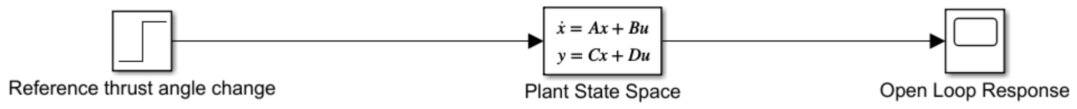


Figure 8: Open loop model in simulink

The open loop response of the system after the simulation is shown in figure 9:

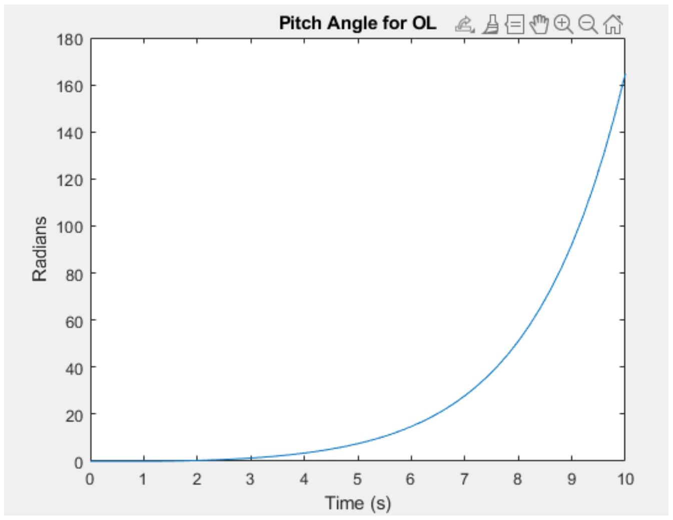


Figure 9: Open loop response

The response is according to the prediction as it is very unstable and diverges quickly to infinity. This response can be calculated quantitatively shown through the eigenvalues of the system with MATLAB.
Using pre-programmed code to create a state space model in MATLAB:

$$\mathrm{system} \;=\; \mathrm{ss}(\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{D}) \tag{2.18}$$

and use:

$$eig(system) \tag{2.19}$$

the eigenvalue of the system is shown in Table 3:

Table 3: Eigenvalues of the open loop

Eigenvalues
-0.6579
0.5657
0.0922

Chapter 3: PID

3.1 Introduction to Classical Control

Classical control has been in use since the early 1900s with different ways to manipulate and control the desired outcome of a vehicle. There are several methods that were developed to analyze, and design closed control loops including Laplace transform, Bode plots, Nyquist stability criterion, root locus, etc... Using Laplace transform to change ordinary differential equations (ODE) into a simple function that represents the output for each possible input and using Bode plot to find the gain and phase margins of a system, classical control methods are proven to be valuable in a successful design. One of the most important and still relevant control system methods for a closed loop system is a Proportional, Integral and Derivative (PID) controller. PID was first developed in 1939 and, due to its simplicity and ease of understanding, it has been in use for the majority of control systems in the technological world.
PID acts as a compensator for the whole system by manipulating the error signal e into a new input signal and obtaining the desired response. Figure 10 shows a basic PID block diagram with each gain:

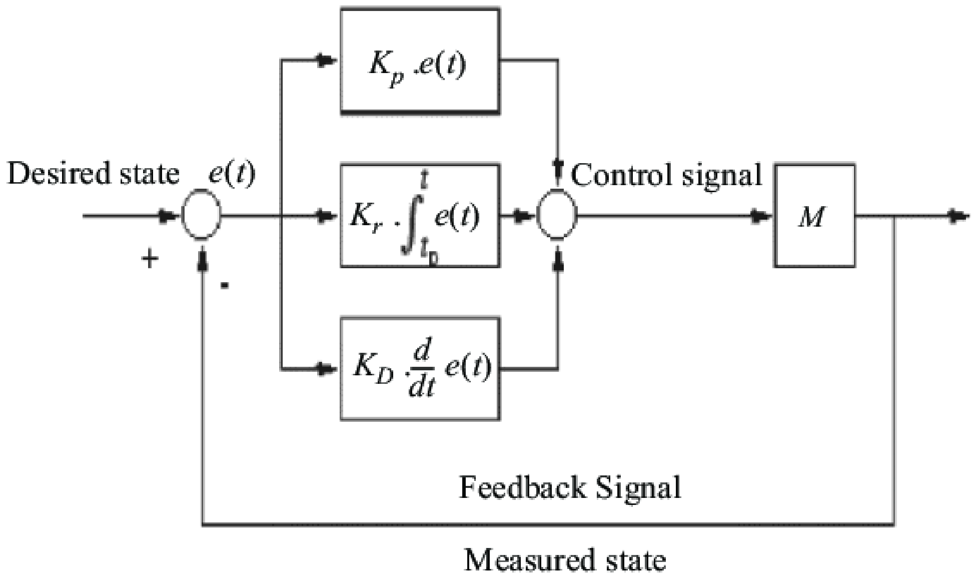


Figure 10: PID block diagram [6]

The error is defined as:

$$e(t) = r - y \tag{3.1}$$

Where r and y are the reference and output signals, respectively.

The error signal is fed into the controller where it is subjected to a proportional, integral, and derivative functions with specific gains, which can be optimized intuitively and easily, and the desired response signal is acquired.

3.2 PID Usage and Gains

The transfer function of a PID can be written in the Laplace domain as:

$$G(s) = K_P + \frac{K_I}{s} + K_D s \tag{3.2}$$

With time domain being:

$$u(t) = K_P e(t) + K_I \int e(t)dt + K_D \frac{de(t)}{dt} \tag{3.3}$$

Essentially, the error signal is proportionally multiplied, integrated, and derived to obtain a desired, new input signal.

To obtain the optimal response for an input or disturbance, the gains for each control mechanism must be tuned and tested. There are several ways to tune the PID gains but first, it is necessary to understand each gain and its effects on the response.
Table 4 shows the relationship between response parameters and stability with increasing different PID gains.

Table 4: PID gains effects [3]

	Rise time	Overshoot	Settling time	Steady-State Error	Stability
Increase K_P	Decrease	Increase	Small Increase	Decrease	Degrade
Increase K_I	Small Decrease	Increase	Increase	Large Decrease	Degrade
Increase K_D	Small Decrease	Decrease	Decrease	Minor Change	Improve

For this PID controller, Simulink auto-tuner that is programmed into the PID block will be used to optimize the gains.

3.3 Implementation of PID and Tuning

In MATLAB and Simulink, it is very simple to create a full closed loop response system with pre-programmed PID and state space blocks. Since the controller is affecting the whole system compared to a controller such as a servo motor, the PID block will be connected to a continuous, linear state space block. Figure 11 shows the full closed loop control diagram.

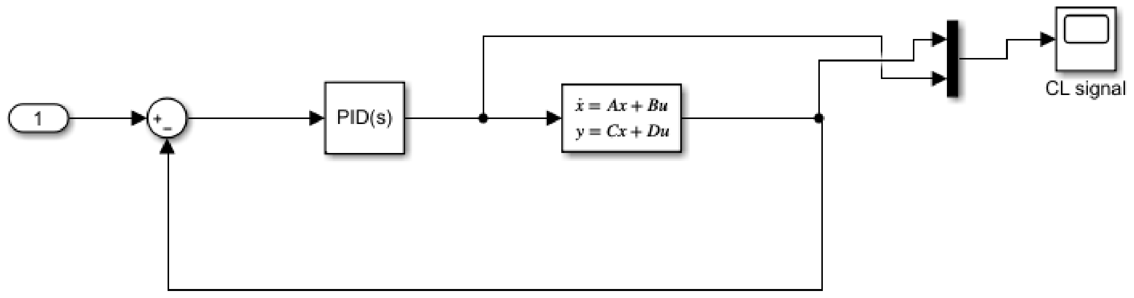


Figure 11: PID control

The PID block may be tuned using different methods, however, Simulink auto-tuning option in the PID block diagram properties will be used. Once certain gains are obtained, trial and error can be used to obtain an even more optimized response.
Tuned values for P, I, D are:

Table 5: PID gains

P	I	D
157.71	75.26	48.55

With a pitch angle reference signal of 0.5 amplitude, and tuned PID gains, the full closed loop response is obtained and graphed (Figure 12).

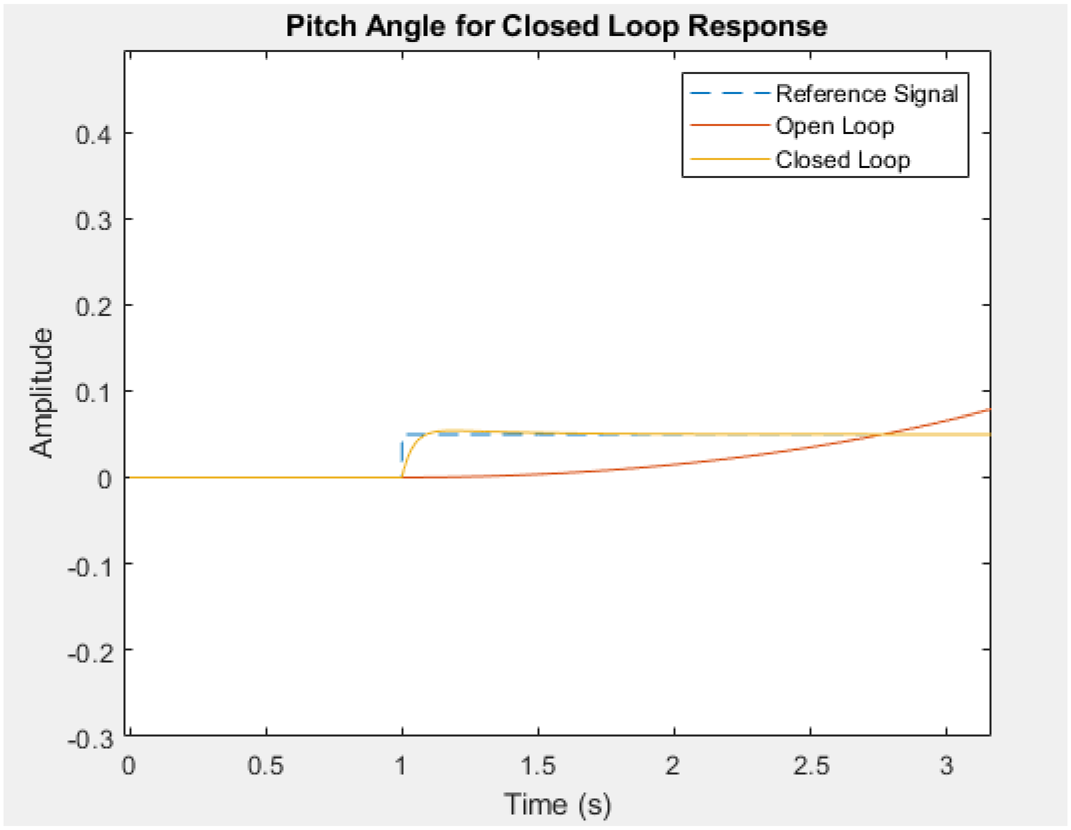


Figure 12: Closed loop PID controlled response

In comparison to the open loop, PID controlled response is converging to the desired response within 7 – 8 seconds with very minimal overshoot.

Chapter 4: LQR Controller Implementation

4.1 Theory

LQR is an optimal control law that uses a quadratic performance index *J* function, or a cost function, to obtain the optimal weighing factors *Q* and *R* and find the LQR gain matrix *K*.

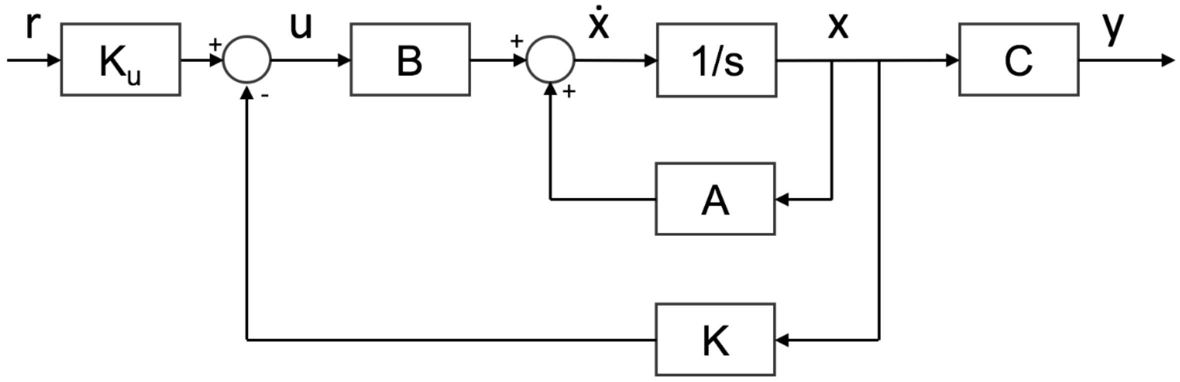


Figure 13: LQR block diagram [7]

For linear time-invariant system, an optimal control law seeks to find an input that allows the system to follow an optimal, predetermined path that minimizes the cost function. The system,

$$\dot{x} = g(x(t), u(t), t) \tag{4.1}$$

requires a cost function or a performance criterion:

$$J = \int_{t_0}^{t_1} h(x(t), u(t), t)dt \tag{4.2}$$

for optimal control. Hamilton-Jacobi equation can be solved using a quadratic performance criterion to obtain the necessary parameters for an optimal gain *K*. Defining a function:

$$f(x,t) = min \int_{t_0}^{t_1} h(x,u)dt \tag{4.3}$$

the Hamilton-Jacobi equation becomes:

$$\frac{\partial f}{\partial t} = -min\left[h(x,u) + \left(\frac{\partial f}{\partial x}\right)^T g(x,u)\right] \tag{4.4}$$

where, if Eq 4.2 is a quadratic function, then the quadratic performance index becomes:

$$J = \int_0^\infty (x^T Qx + u^T Ru)dt \tag{4.5}$$

Substituting equations 4.4 and 4.5 turns to:

$$\frac{\partial f}{\partial t} = -min\left[x^T Qx + u^T Ru + \left(\frac{\partial f}{\partial x}\right)^T (Ax + Bu)\right] \tag{4.6}$$

Once the LQR gain is found, the new input becomes:

$$u = -Kx \tag{4.7}$$

Where K is defined as:

$$K = R^{-1}B^T P \tag{4.8}$$

and P is found from the Riccati equation:

$$PA + A^T P + Q - PBR^{-1}B^T P = 0 \tag{4.9}$$

Since all the other matrices are known or already pre-defined, the solution to the LQR gain can be calculated.

4.2 Cost Function

The Q and R matrices are *l x* l and *m x m* symmetric positive-definite matrices that represent the weights assigned to the state and the input parameters. Looking at the cost function:

$$J = \int_0^\infty (x^T Qx + u^T Ru)dt \tag{4.10}$$

the parameters can be independently configured to reach an optimal solution. In this system, Q would be correlated with the pitch, pitch rate or the drift of the rocket. By increasing or decreasing the value of Q, the state parameters are affected according to the “weight” that is placed upon them. Similarly, the R value would affect both input vectors, pitch in this case, and can adjusted according to the designer’s need or hardware limitations.

The Q and R matrices for this rocket underwent a trial-and-error method to find the most optimal path for a desired input.

$$Q = \begin{bmatrix} 10000 & 0 & 0 \\ 0 & 150000 & 0 \\ 0 & 0 & 7000 \end{bmatrix} \tag{4.11}$$

$$R = \begin{bmatrix} 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}$$

4.3 MATLAB and Simulink of LQR Controller

For MATLAB and Simulink approach, it is simple to find the necessary gains of the LQR, the eigenvalues and the P value of the Riccati equation. First, the state space of the model must be created in MATLAB using $system = ss(A, B, C, D)$ code and the parameters using:

$$[K \quad P \quad E] = lqr(system, Q, R) \tag{4.12}$$

Once the gain is found, using a preset initial condition (Eq 4.12) and a reference input (Eq 4.11) for pitch, the SIMULINK block diagram can be created for simulations.

$$r = \begin{bmatrix} 0.05 \\ 0 \\ 0 \end{bmatrix}$$

$$x_0 = \begin{bmatrix} 0.012 \\ 0 \\ 0 \end{bmatrix} \tag{4.13}$$

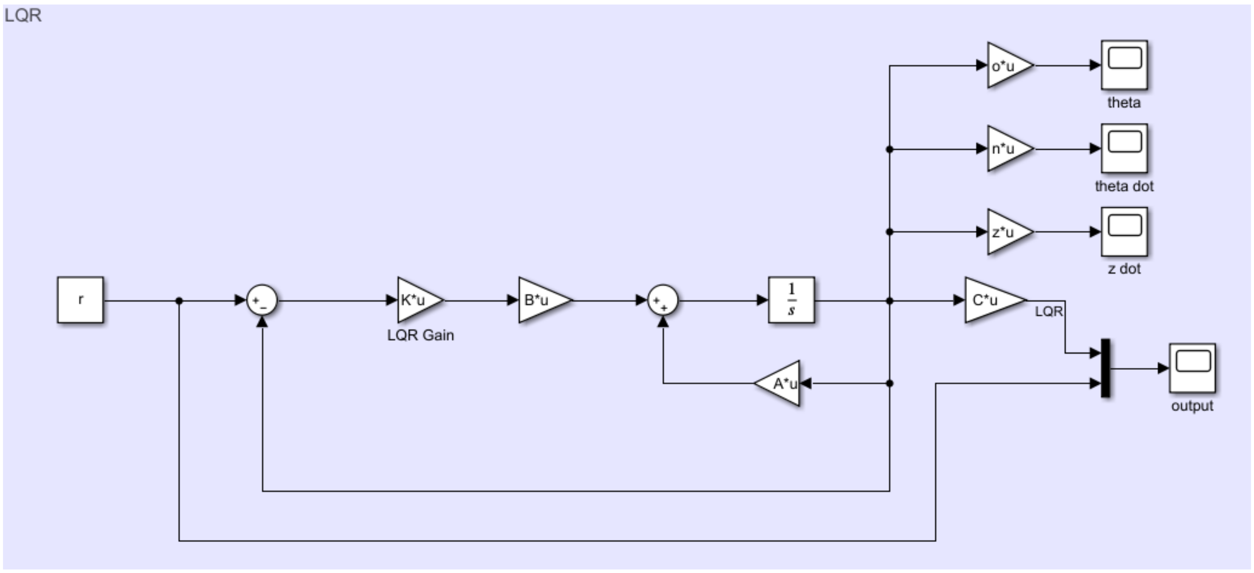


Figure 14: LQR SIMULINK block diagram

With the full closed loop response with LQR gain for the error signal, the convergence of pitch angle of the rocket, the pitch rate and the lateral drift were found.

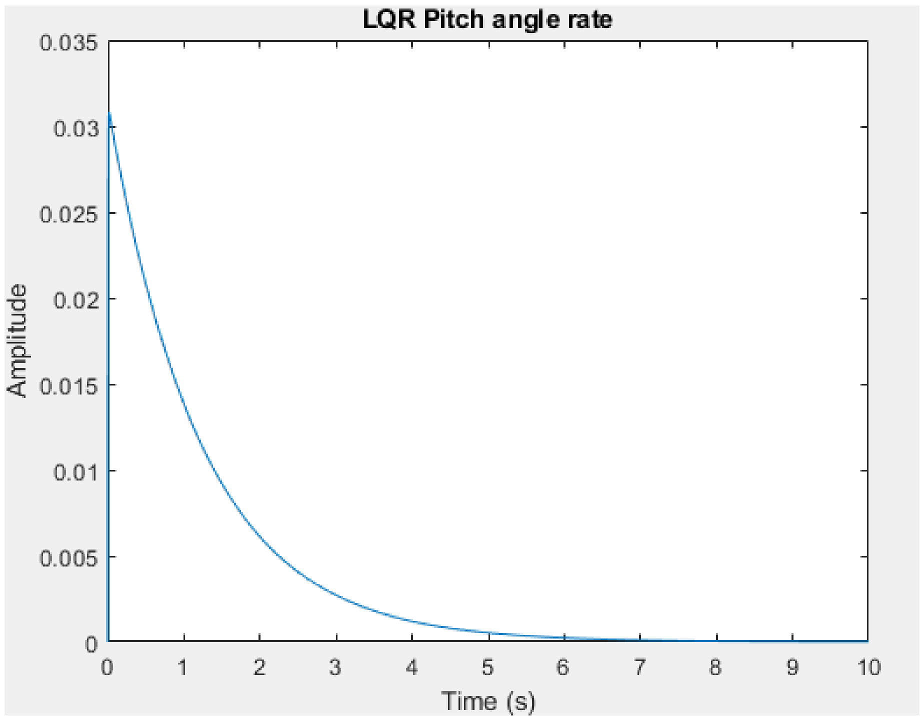


Figure 15: LQR pitch angle rate (deg/s)

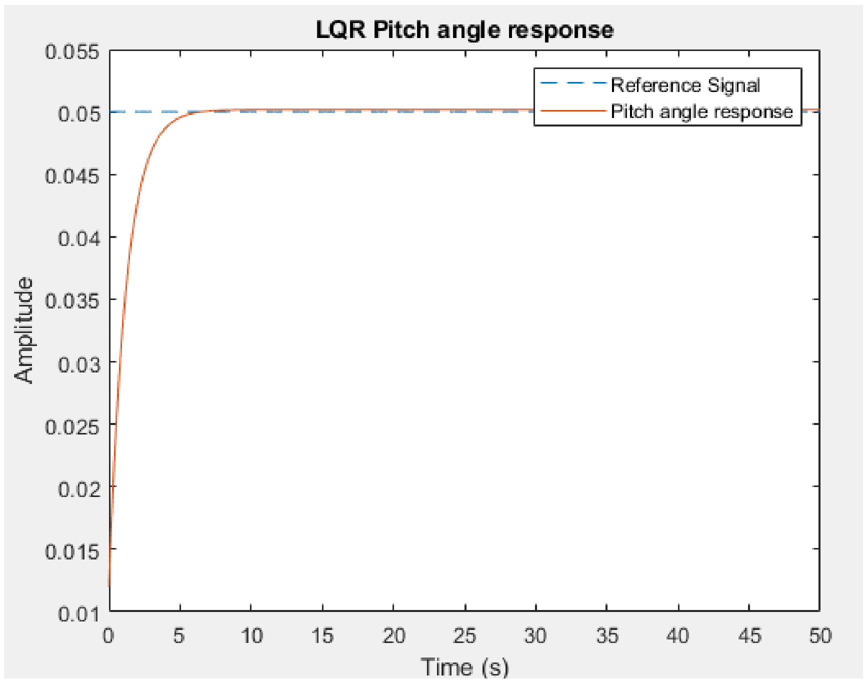


Figure 16: LQR pitch angle response (deg)

The LQR controller sufficiently controls the response to the input, however, the response converges at much slower speeds than expected. It takes about 5-6 seconds to reach the commanded pitch angle.

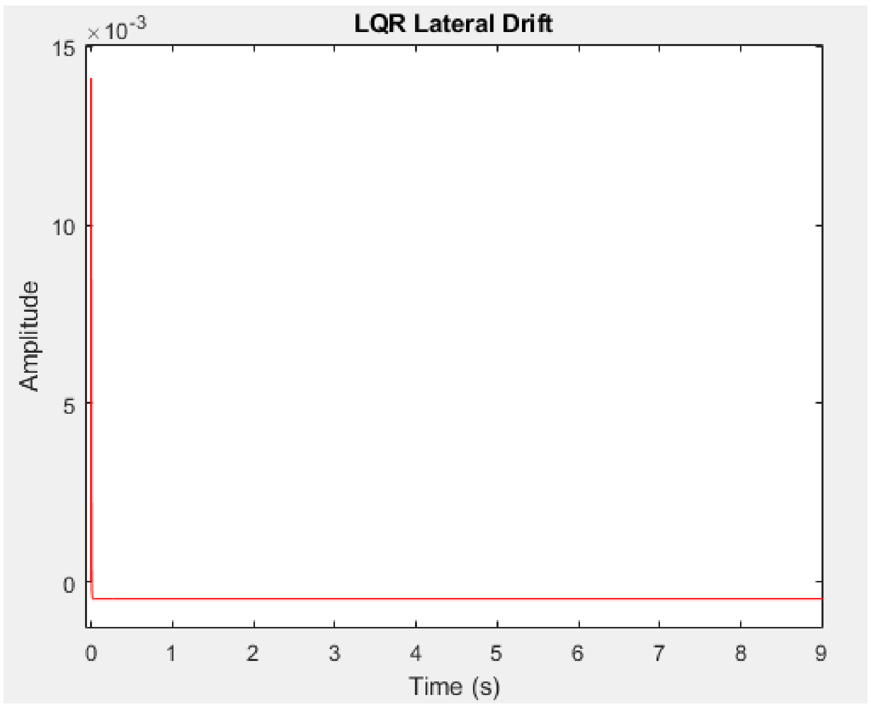


Figure 17: LQR lateral drift (m)

The miniscule change in the lateral drift doesn't affect the rocket dynamics as significantly, thus resulting in a satisfying response from the LQR controller.

4.4 Comparison of Classical and Modern Control

There are lot of merits for both PID and LQR control methods. Considering the advantages and the disadvantages, both theories converged to the reference input at a similar rate. Although PID has a small amount of overshoot and instability in maintaining the response, simple tuning of each gain for their respective parameters will be sufficient in fully implementing the controller on the rocket.

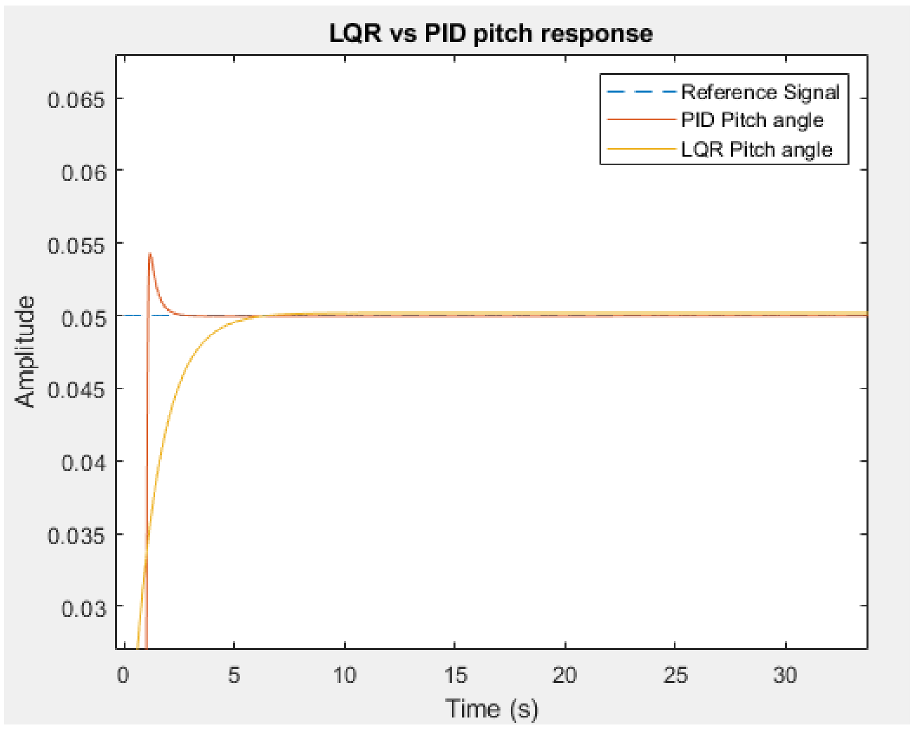


Figure 18: LQR vs PID controller pitch response comparison (deg)

Compared to PID, LQR controller holds its stability very well, which can attribute to high tuned optimal gains and the weighing matrices. Overall, although LQR seems to obtain results with less overshoot for this rocket plant, PID is able to converge faster to the given input with little overshoot and a minimal steady state error. Even if LQR is an optimal feedback controller with better cost function to obtain optimal gain, PID control is more intuitive and robust in its control and able to bring forth superior results.

Chapter 5: Kalman Filter

5.1 Overview of KF

In real life application of control theory, there are numerous uncertainties that must be considered during the design process of any system. When analyzing any system, it is crucial for the control engineer to understand the full states, which are the variables that are necessary in the dynamics of a system, that are involved in the behavior to implement a correct method. Although it is necessary, the states of the system are often not known, thus requiring ways to obtain them indirectly. Observers, primarily Kalman Filter, are one of the methods that can be used to estimate the full state of the system, using the input and the output of the system response. Kalman filter is an optimal state estimator that used the process and the measurement noises, including the input and the output, to estimate the full state of the system. Since there are numerous disturbances in real life cases, including sensor and environmental noise, that must be considered as it can affect the performance of a system, Kalman Filter becomes a powerful analysis tool for a control engineer.

Suppose we have estimate state $\vec{x}$ which estimates the full state vector x:

$$\dot{\vec{x}} = Ax + Bu \tag{5.1}$$
$$\dot{e} = Ax - A\vec{x} \tag{5.2}$$

then the state estimation error becomes:

$$e = x - \vec{x} \tag{5.3}$$

Combining the equations 5.2 and 5.3, we can see that,

$$\dot{e} = Ae \tag{5.4}$$

This means that if matrix A is asymptotically stable, the error will converge to zero for any input; meaning that as time reaches to infinity, the estimated state $\vec{x}$ will converge to the true state x. Alternatively, if A is an unstable vector, then the estimated state will diverge away from the actual state values. In order for the error term to reach a desired value (i.e. zero), a necessary Kalman gain value L is applied to equation.

$$\begin{gathered}\dot{\vec{x}} = Ax + Bu + L(y - \vec{y}) \\ \vec{y} = C\vec{x}\end{gathered} \tag{5.5}$$

It is important to note that in Eq 5.5, as $\vec{x}$ converges to x, $\vec{y}$ also converges to y, negating the gain value and the system reaching its true state.

The error term, on the other hand, becomes:

$$\dot{e} = Ax - A\vec{x} - L(Cx - C\vec{x}) = (A - LC)e \tag{5.6}$$
$$\dot{\vec{x}} = (A - LC)x + Bu + Ly \tag{5.7}$$

Eq 5.7 is what is known as the full order observer (as shown in Fig. 19), which considers both input and the output for its estimation method.

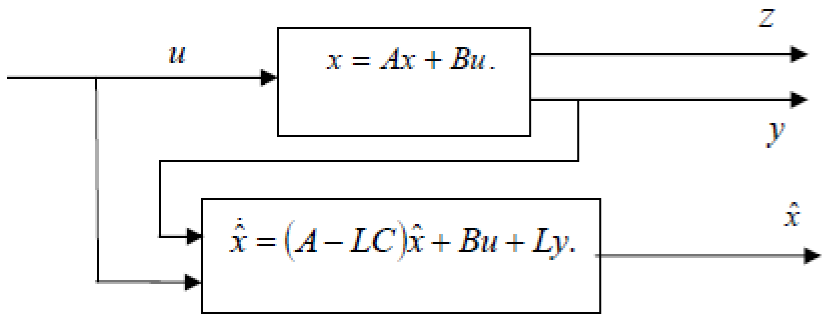


Figure 19: Full-order observer [9]

For Kalman Filter, an optimal estimator compared to an observer, introduces two different parameters:

$$\begin{aligned} \dot{\vec{x}} &= A\vec{x} + B\vec{u} + Gw(t) \\ \vec{y} &= C\vec{x} + v(t) \\ S_w(\omega) &= Q_N \\ S_v(\omega) &= R_N \end{aligned} \tag{5.8}$$

where $w(t)$ and $v(t)$ are zero-mean Gaussian noises with Q and R being process noise covariance and measurement noise covariance matrices, respectively.

5.2 State Estimation

The most important parameter of the system that must be considered when an observer is involved, is to verify if the system is observable or not. The observability of a system can be calculated by finding the rank of the system, using Eq. 5.9. If the system has *n* number of unknowns, then the system is only observable if the rank of the observability matrix is also *n*.

$$rank[B \quad AB \quad A^2B \quad \cdots \quad A^{n-1}B] = n \tag{5.9}$$

From Chapter 2, we found that the rank of the system to be full rank and the observability matrix:

$$rank[B \quad AB \quad A^2B \quad \cdots \quad A^{n-1}B] = n \tag{5.10}$$

$$Ob = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0.3807 & 0 & 0.0009 \end{bmatrix}$$

Using the pre-programmed MATLAB function for observability, it is determined that the system is fully observable, thus a full Kalman Filter estimator can be implemented.

For the noise and measurement covariance parameters, simple trial and error to find the most optimal Kalman Filter gain were found and utilized:

$$S_w(\omega) = Q_N = 0.1 \tag{5.11}$$

$$S_v(\omega) = R_N = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.12}$$

Using the given model with disturbances, MATLAB function:

$$[Kest\ L\ P] = kalman(system, Q, R, N) \tag{5.13}$$

can be used to find the Kalman gain, the Riccati equation matrix P and the estimated states *Kest*.

5.3 Optimal State Estimation and MATLAB Implementation

In Simulink and MATLAB, the Kalman Filter observer is calculated using the input with B gain matrix and the output. As shown in Figure 20, different gains were used to separate each parameter for better comparison of the response.

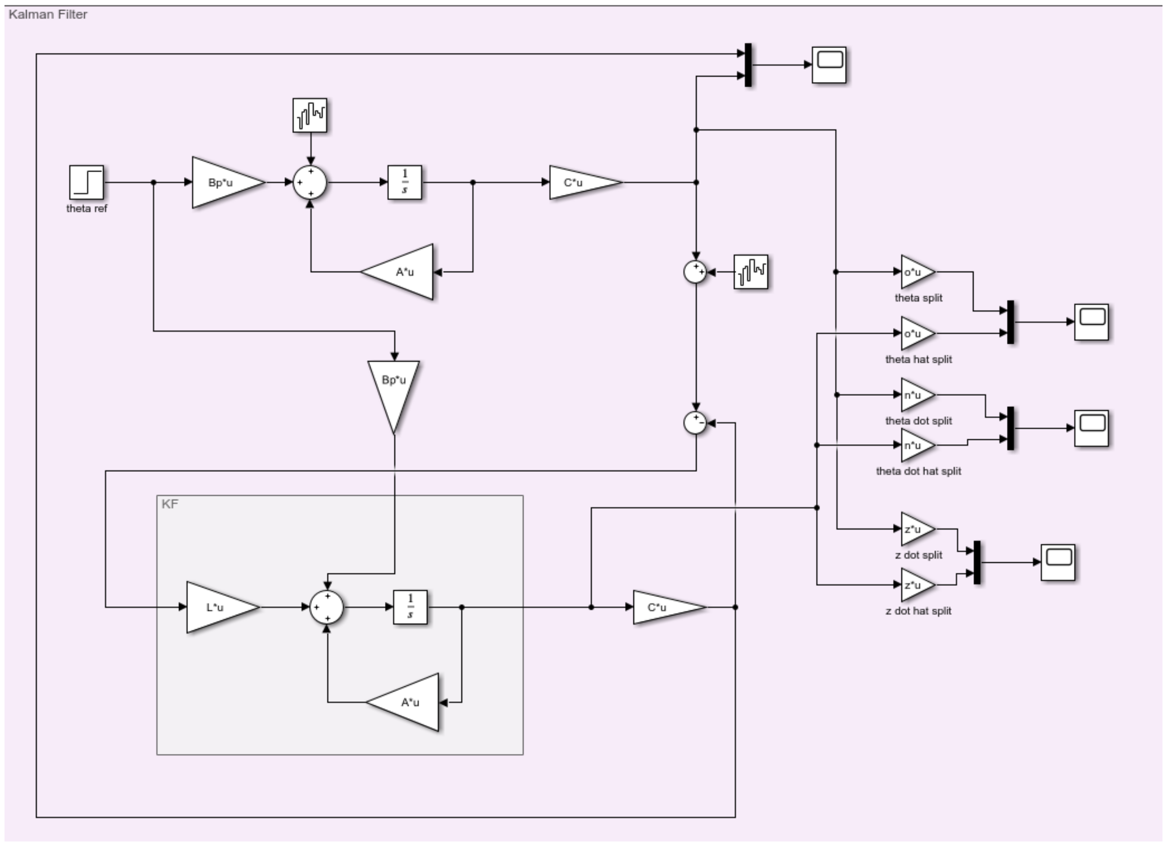


Figure 20: Simulink kalman filter diagram

With a simple step input of 0.05 amplitude, each state has a different response. Since an observer is being applied to an open loop system, the response is very unstable and diverges fast. It is shown that in Figures 21, 23 and 23, the Kalman Filter is able to closely estimate the true state of the system.

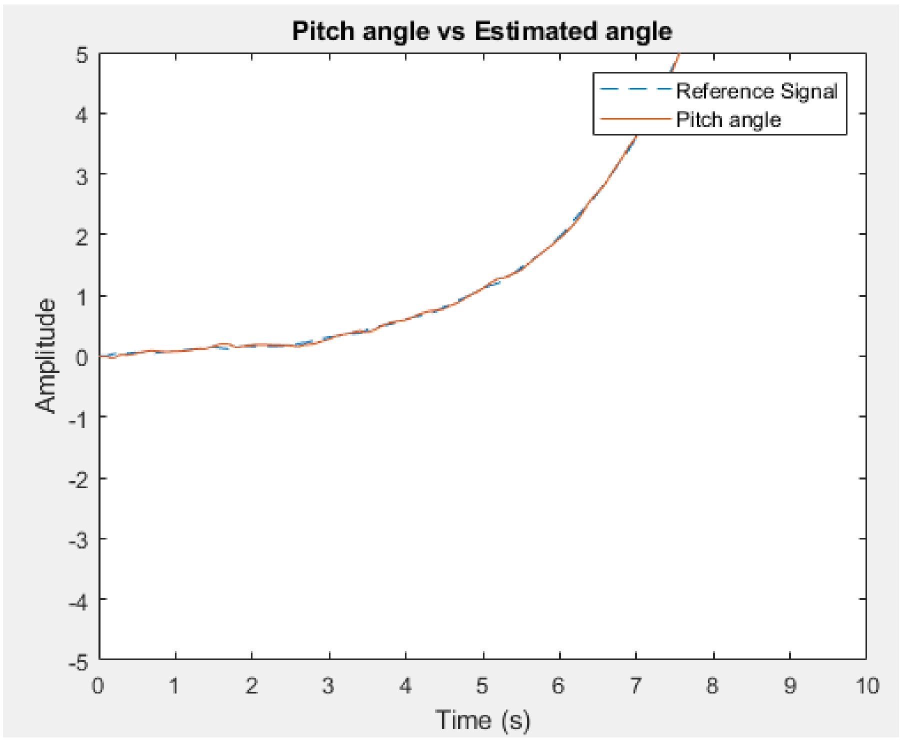


Figure 21: KF true vs estimated angle (deg)

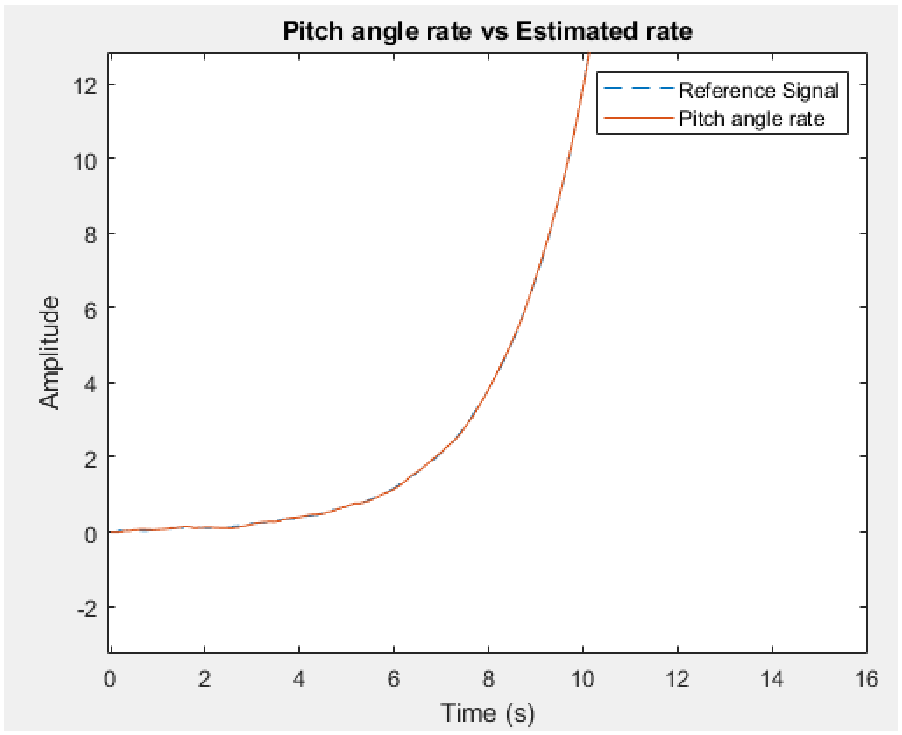


Figure 22: KF true vs estimated angle rate (deg/s)

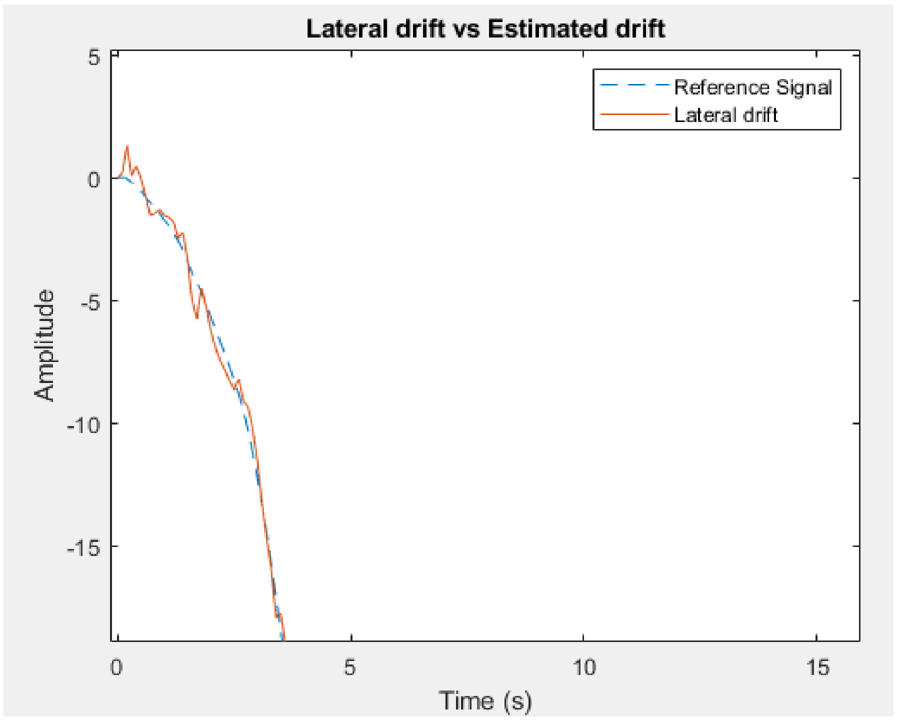


Figure 23: KF true vs estimated lateral drift (m)

Chapter 6: LQG Control

6.1 Overview of LQG

With the LQR controller obtaining the optimal control gain K and the Kalman Filter estimating the true states of a system, the Linear Quadratic Gaussian controller can be obtained by combining LQR and Kalman Filter. The prerequisite of a LQR controller is to have every state variable known beforehand. As it is improbable to obtain every state of the system, the Kalman Filter is to compensate for it. This is one of the most important advantages of an LQG controller and the primary reason for its use in control design.

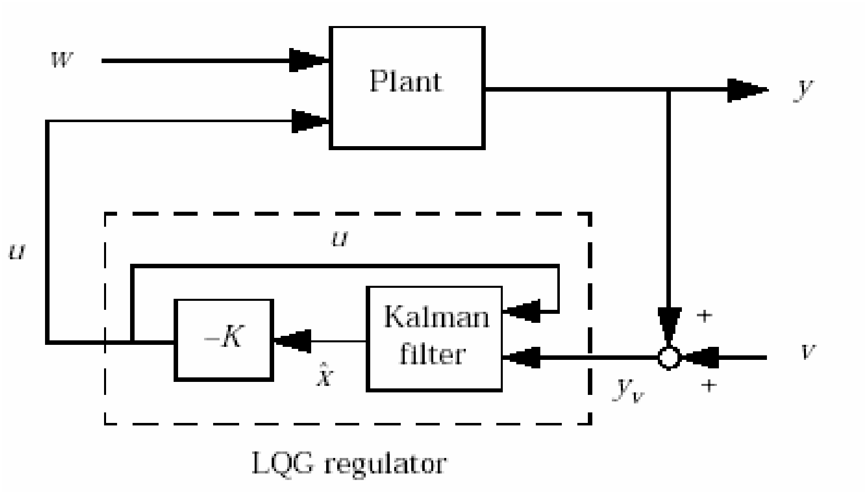


Figure 24: LQG general setup [10]

With a LQR controller gain connected to the estimated open loop response, an optimal closed loop response can be generated, and the error signal will converge to zero.

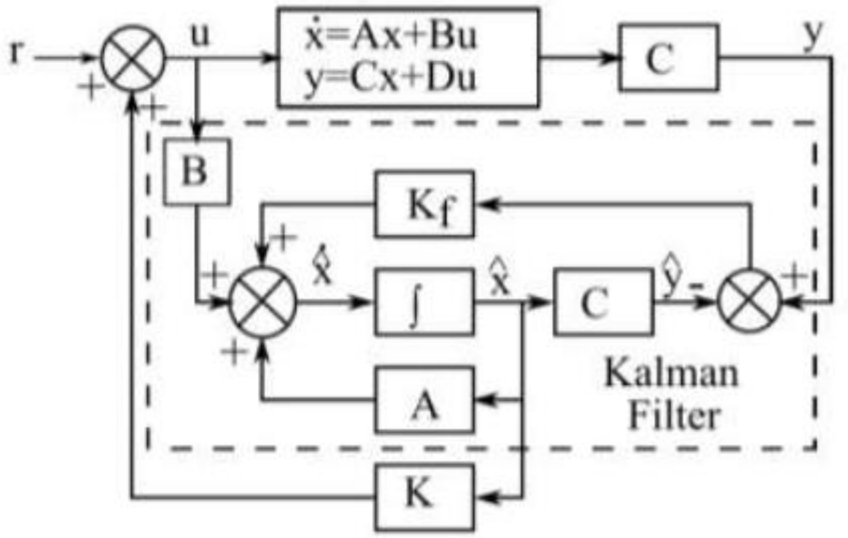


Figure 25: LQG detailed block diagram [11]

It is important to note that, although LQG controller is applicable to most systems, the disadvantage of Kalman filter based LQR design is that it makes the open-loop system have very low stability margins.

6.2 Separation Principle

For a linear time-invariant system, the separation principle states that an optimal observer and a feedback controller can be designed separately and combined to create a stable response of a system. From the observer equations,

$$\dot{\vec{x}} = (A - LC)\vec{x} + B\vec{u} + Ly \tag{6.1}$$

the estimated state variables $\vec{x}$ are obtained and is used in the LQR gain equation,

$$u = -K\vec{x} \tag{6.2}$$

to obtain the optimal controlled input for the system. Substituting eq. 6.1 and 6.2 to the error equation $e = x - \hat{x}$,

$$\begin{aligned} \dot{e} &= (A - LC)e \\ u &= -K(x - e) \end{aligned} \tag{6.3}$$

the matrix form of closed loop dynamics can be obtained,

$$\begin{bmatrix} \dot{x} \\ \dot{e} \end{bmatrix} = \begin{bmatrix} A - BK & BK \\ 0 & A - KC \end{bmatrix} \begin{bmatrix} x \\ e \end{bmatrix} \tag{6.4}$$

Since the eigenvalues of the matrix consist of A-BK and A-KC, both parameters are independent of each other.

6.3 Combination of LQR and Kalman Filter

The Simulink implementation of the LQG is simple as it combines the block diagram of Kalman Filter with the addition of LQR gain.

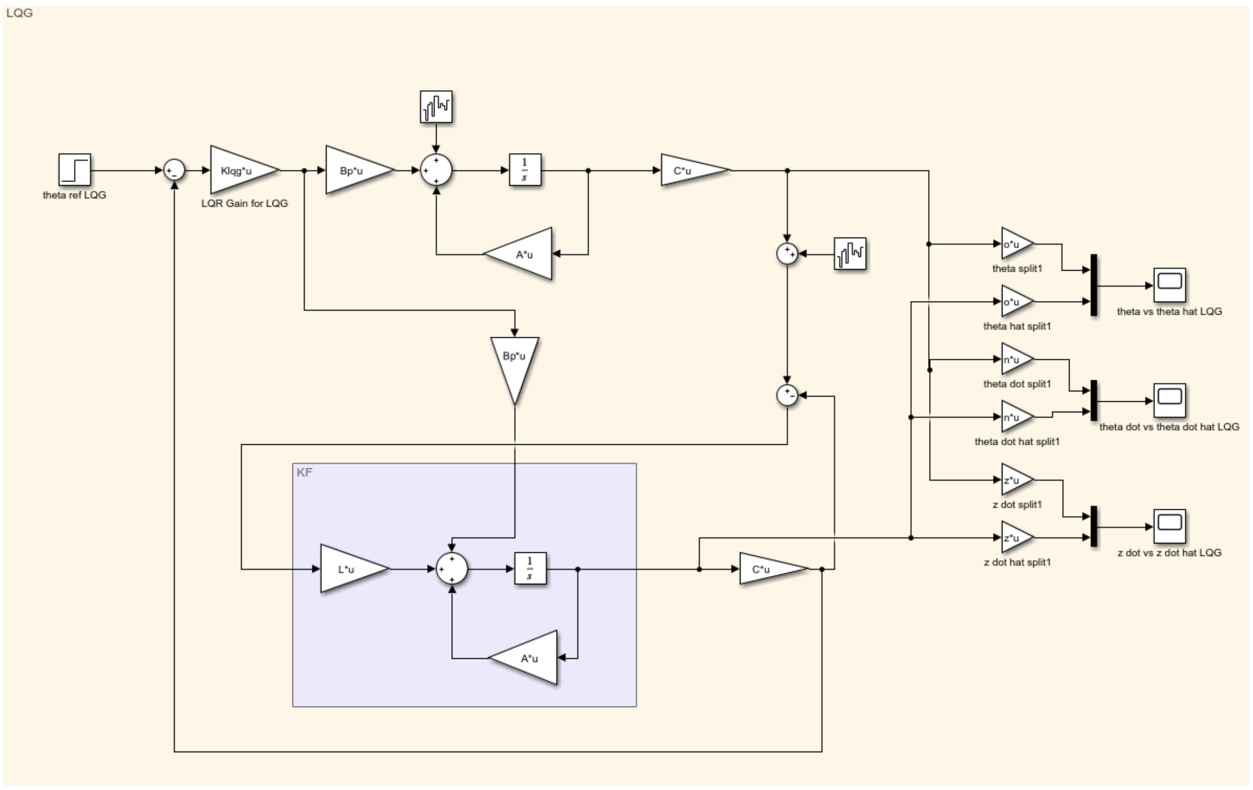


Figure 26: LQG Simulink block diagram

A step input of 0.05 amplitude is also applied to the entire system and the response of each state variable, pitch angle, angle rate and the lateral drift, were documented for analysis.

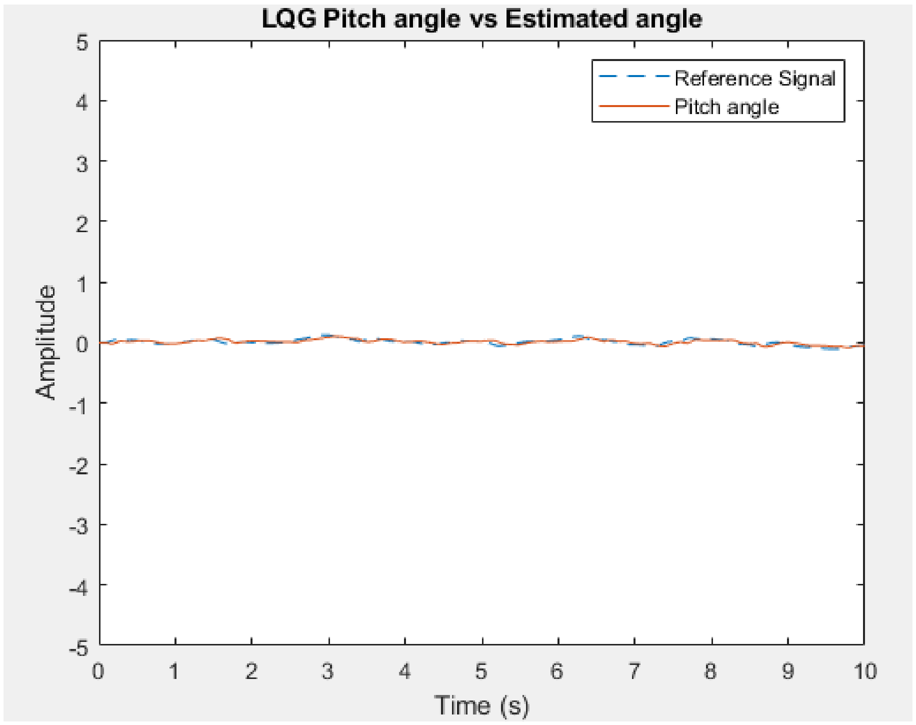


Figure 27: LQG estimated vs true angle (deg)

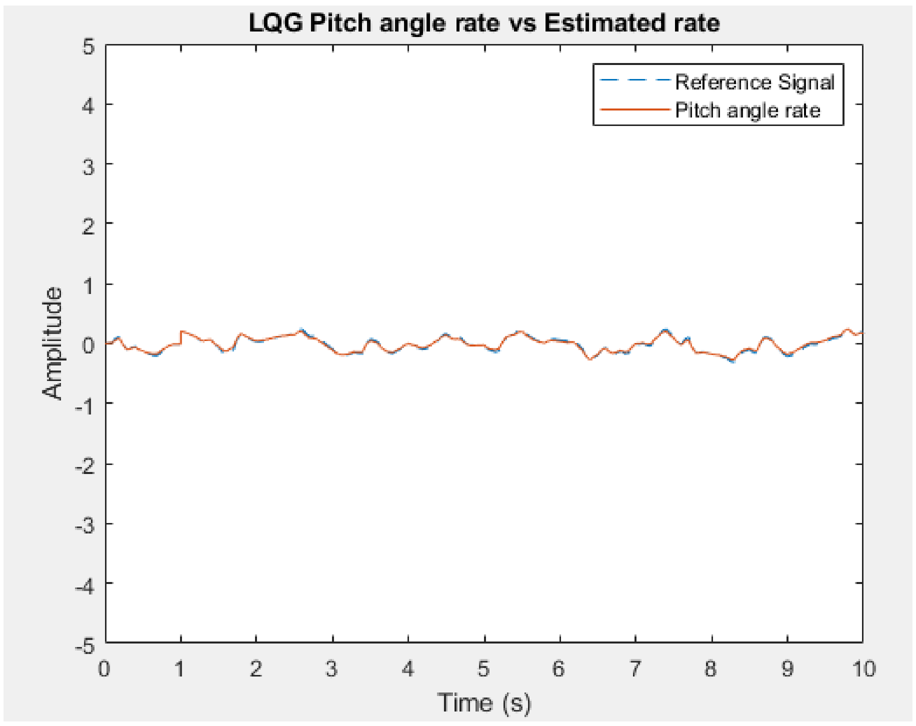


Figure 28: LQG estimate vs true angle rate (deg)

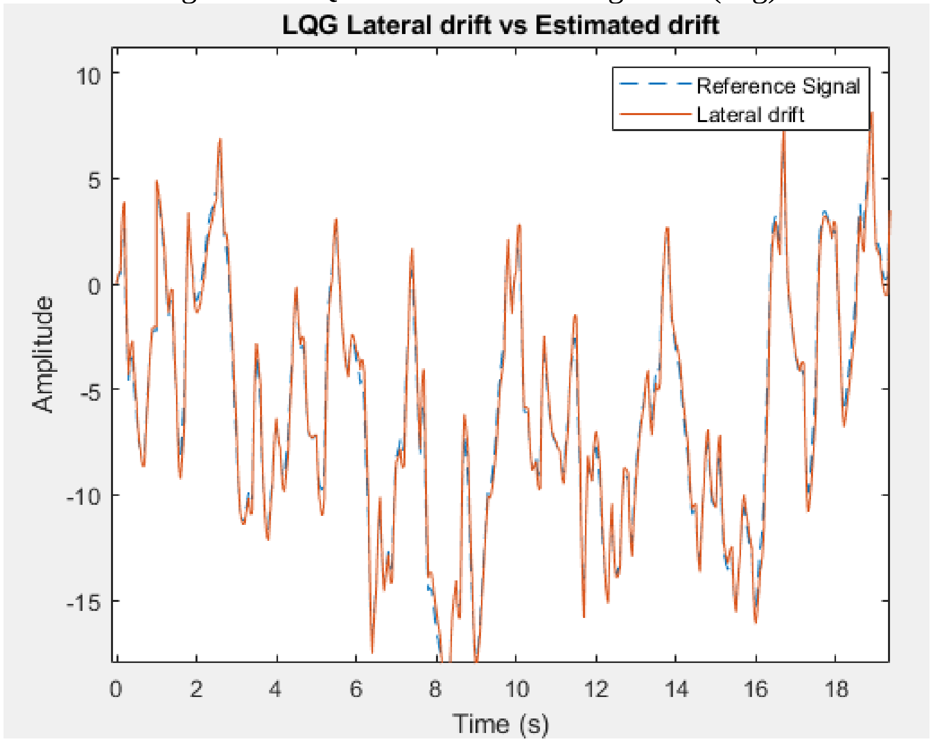


Figure 29: LQG estimated vs true lateral drift (m)

It is clear with LQR gain, the closed loop response is considerably stable for both pitch angle and the pitch angle rate, as predicted. The error term is also quite small for both of the states, excluding the lateral drift which is still noticeably unstable. Although the controller is compensating for the error term, the lateral drift is larger than expected.

Chapter 7: Discussion

In the early 1940's, simple PIDs were used to control the majority of the systems, but with the creation of parameter tuning, several different types of tuning were introduced to the method [22]. Introduction of auto-tuning, self-tuning, robust, optimal tuning and etc..., the simple PID controller was transformed into a system with a wide variety of applications and uses. In addition to the use of PID in the aerospace industry, the PID control method is also used in process control, robotic fields, biomedical applications and in mechanical and power systems [22].

Even though the open loop response of this system was significantly unstable, the feedback control systems were able to correct any errors that occurred for the state variables. Since the rocket doesn't possess any control surfaces to provide static stability, PID provided simple, yet effective control over the response. With the given gains for *P*, *I* and *D*, and using the auto-tuning software provided by Simulink block set, an acceptable response with minimal overshoot and damping was created. Even without any auto-tuning methods, with the provided table of PID gains and their effects, a simple trial-and-error method is sufficient in finding satisfying gains for the method. One of the main advantages of using PID is that it is very robust and intuitive compared to methods such as LQR or LQG due to its simplicity and implementation. If a system has a steady-state error, the *I* gain is increased until the error disappears or, if the system is under or overdamped, the *D* gain is manipulated until desired damping of the system is obtained.
LQR, on the other hand, uses a cost function to find the optimal control gain, *K*. After deriving the Ricatti equation to find the P matrix and manipulating the *Q* and *R* weighting matrices, the LQR response of the system is obtained. In this case, the response converges on the input with no overshoot and no steady state error but the time to reach full convergence was an issue. The LQR solution took significantly longer than the PID to reach the desired command as the rise time for PID was considerably faster. Although it is possible to tune the LQR gain, the process is less intuitive and requires more in-depth knowledge of how LQR processes and matrices affect the gain.

With the use of a linear quadratic estimator (LQE), the Kalman Filter gain of the system was also obtained to estimate the full state variables. Using the output, the input, process, and measurement noises, estimating the full state is a very powerful tool in real life applications. To fully utilize the LQR method on a system, Kalman Filter or any other LQE must be employed simultaneously to obtain the states and control the response after.
As such, the Linear Quadratic Gaussian controller is formed to fully observe and control the given unstable system. By incorporating the KF gain for the state and manipulating the estimated states using LQR gain, the response of the combined methods is significantly more robust and optimized than individual control systems.

Chapter 8: Conclusion

The design process included deriving the equations of motion of the rocket, linearizing the equations, and obtaining a linear time-invariant state space model. All of these steps are essential in designing a suitable control system for any rigid body. Depending on the type of system, there can be several different methods that can be used to control it. From classical to modern systems, different methods can be appropriate for different situations. With PID being the most robust and easy to use method, the majority of the engineering industry is implementing PID in most control architectures. PID also requires the least amount of derivation and knowledge to fully take advantage of its benefits. With its simplicity and ease of use, there are less avenues of error and simple troubleshooting. The usage of Kalman Filter estimator is necessary for any control problem due to its function in understanding the full system. By estimating every state variable of the system, a feedback control can be designed to command any of the observed state variables.

Each classical or modern control method has its own advantages and disadvantages. However, in comparing the responses of PID, LQR and LQG systems, given that the states are known or estimated, PID appears to be a superior and practical system out of all. For any method, the simplest and easiest, yet effective system will be highly prioritized. For most systems, the control engineer can estimate how the system behaves and implement a simple or complex PID controller that can negate most of the undesired responses that may arise.

References

[1] "Earth Centred Inertial Frame", *ADCS For Beginners*.
https://adcsforbeginners.wordpress.com/tag/earth-centred-inertial-frame/

[2] Nourmohammadi, H., Keighobadi, J., Fuzzy adaptive integration scheme for low-cost SINS/GPS navigation system, *Mechanical Systems and Signal Processing*, Vol. 99, pp. 434-449. June 2017.

[3] Li, Y., Ang, K.H., Chong, G.C.Y., PID control system
analysis and design, *IEEE Control Systems Magazine*, Vol. 26, 2006, pp. 32-41
<http://eprints.gla.ac.uk/3815/1/IEEE_CS_PID_01580152.pdf >

[4] Thrust Vectoring, *Wikipedia*. <https://en.wikipedia.org/wiki/Thrust_vectoring>

[5] Gimbaled Thrust, *NASA*. <https://www.grc.nasa.gov/www/k-12/rocket/gimbaled.html>

[6] Bansal, H., Tuning of PID Controllers using Simulink, *International Journal of Mathematical Modeling, Simulation and Applications*, 2009. <https://www.researchgate.net/figure/Block-diagram-of-a-system-with-PID-controller_fig1_268802558>

[7] Simulating and controlling an inverted pendulum, MIT.
<https://fab.cba.mit.edu/classes/864.17/people/copplestone/final_project/index.html>

[8] Demir, A., E., Designing a 6-DOF Model of a Rocket With Realistic Sensor Models, *Istanbul Technical University*, 2021.

[9] Mukhopadhyay, A., Ganguly, A., State Estimation by Orthogonal Hybrid (combination of SHF and RHTF) Function, 2016.
<https://www.researchgate.net/publication/300417683_State_estimation_by_orthogonal_hybrid_combination_of_SHF_and_RHTF_function>

[10] Maddi, A., Guessoum, A., Berkani, D., Using Linear Quadratic Gaussian Optimal Control for Lateral Motion of Aircraft, 2009.
<https://www.researchgate.net/publication/336856020_Using_Linear_Quadratic_Gaussian_Optimal_Control_for_Lateral_Motion_of_Aircraft>

[11] Hossain, A., Linear Quadratic Gaussian Regulator Based Frequency Control of Power System to Enhance the Continuity of Power Flow, *American Journal of Engineering Research*, Vol. 8, Issue 4, pp. 348-354, 2019. <https://www.ajer.org/papers/Vol-8-issue-4/ZZK0804348354.pdf>

[12] Kisabo, A., B., Adebimpe, A., F., Samuel, S., O., Pitch Control of a Rocket with a Novel LQG/LTR Control Algorithm, *Journal of Aircraft and Spacecraft Technology*, 2019.

[13] Shaji, J., Aswin, R., B., Pitch control of aircraft using LQR & LQG control, *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, Vol. 4, Issue 8, Aug. 2015.

[14] Kisabo, A., B., Modern Approach to Aerospace Vehicle Navigation and Motion Control Systems Design, *Saint-Petersburg State University of Aerospace Instrumentation*, 2010.

[15] Sopegno, L., Livreri, P., Stefanovic, M., Valavanis, K., P., Linear Quadratic Regulator a Simple Thrust Vector Control System for Rockets, *2022 30th Mediterranean Conference on Control and Automation*, 2022.

[16] Schwab, A., How to Draw Euler Angles and Utilize Euler Parameters, 2006. <https://www.researchgate.net/publication/257921931_How_to_Draw_Euler_Angles_and_Utilize_Euler_Parameters>

[17] Kisabo, A., B., Adebimpe, A., F., Okwo, O., C., Samuel, S., O., State-space Modelling of a Rocket for Optimal Control System Design, *Journal of Aircraft and Spacecraft Technology*, 2019.

[18] Zanatta, R., Sousa, M., S., A 6-DOF Rocket Model For Control Analysis, *International Journal of Engineering Applied Sciences and Technology,* Vol. 1, No. 1, pp. 1-6, 2015.

[19] Das, S., Halder, K., Missile Attitude Control via a Hybrid LQG-LTR-LQI Control Scheme with Optimum Weight Selection.

[20] Phyo, K., Nyein, A., Control of Two-Wheeled Balancing Robot Using PID, 2019. <https://www.researchgate.net/publication/338674100_Control_of_Two-Wheeled_Balancing_Robot_Using_PID>

[21] Astrom, K., J., Introduction to Stochastic Control Theory, *Academic Press*, 1970.

[22] Borase, P., R., Maghade, D., K., Sondkar, S., Y., Pawar, S., N., A review of PID control, tuning methods and applications, *International Journal of Dynamics and Control*, Jun., 2020.

Appendices

A1 Derivations

A1.1 Translational Motion

The force and torque equations:

$$\sum \boldsymbol{F} = m\boldsymbol{a} \tag{A1.1}$$

$$\sum \boldsymbol{T} = \frac{d}{dt}(r \text{ x } m\boldsymbol{V}) \tag{A1.2}$$

can be considered as a conservation of linear and angular momentums:

$$\sum \boldsymbol{F} = \frac{d(m\boldsymbol{V}_m)}{dt} \tag{A1.3}$$

$$\sum \boldsymbol{M} = \frac{d\boldsymbol{H}}{dt} \tag{A1.4}$$

Considering the six degrees of freedom for the equations, Eq A1.1 and Eq A1.4 can be written in the forms of:

$$F_x = \frac{d(mu)}{dt}$$
$$F_y = \frac{d(mv)}{dt} \tag{A1.5}$$
$$F_z = \frac{d(mw)}{dt}$$

and

$$L_x = \frac{dH_x}{dt}$$
$$L_y = \frac{dH_y}{dt} \tag{A1.6}$$
$$L_z = \frac{dH_z}{dt}$$

For the translational degrees of freedom, the frame of reference must be changed from inertial to body frame using the Eq A1.7:

$$\left(\frac{d\boldsymbol{A}}{dt}\right)_{body} = \left(\frac{d\boldsymbol{A}}{dt}\right)_{inertial} + \boldsymbol{\omega} \times \boldsymbol{A} \tag{A1.7}$$

with *A* vector being the main vector. With inertial velocity vector, V_m, the equation becomes:

$$\left(\frac{d\boldsymbol{V}_m}{dt}\right)_{body} = \left(\frac{d\boldsymbol{V}_m}{dt}\right)_{inertial} + \boldsymbol{\omega} \times \boldsymbol{V_m} \tag{A1.8}$$

The vector product of the angular rates and the linear velocity can be calculated using vector multiplication:

$$\boldsymbol{\omega} \times \boldsymbol{V}_m = \begin{pmatrix} i & j & k \\ p & q & r \\ u & v & w \end{pmatrix} = (qw - rv)i + (ur - pw)j + (pv - qu)k \tag{A1.9}$$

with Newton’s second law, the translational equations become:

$$\begin{aligned} \sum F_x &= m(\frac{du}{dt} + qw - rv) \\ \sum F_y &= m(\frac{dv}{dt} + ru - pw) \\ \sum F_z &= m(\frac{dw}{dt} + pv - qu) \end{aligned} \tag{A1.10}$$

Rewriting Eq A1.2.1:

$$\begin{aligned} \frac{du}{dt} &= \left(\frac{\sum F_x}{m}\right) + rv - qw \\ \frac{dv}{dt} &= \left(\frac{\sum F_y}{m}\right) + pw - ru \\ \frac{dw}{dt} &= \left(\frac{\sum F_z}{m}\right) + qu - pv \end{aligned} \tag{A1.11}$$

with aerodynamic, propulsive, and gravitational forces included in the sum.

A1.2 Rotational Motion

The net torque on the rocket is the rate of change of the angular momentum:

$$\sum \boldsymbol{M} = \frac{d\boldsymbol{H}}{dt} \tag{A1.12}$$

with angular momentum vector and expressed in the form:

$$\boldsymbol{H} = I\boldsymbol{\omega} \tag{A1.13}$$

with the general inertial matrix:

$$I = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \tag{A1.14}$$

Using the same multiplication rule from translational derivation, the reference frame is converted to the body frame:

$$\left(\frac{d\boldsymbol{H}}{dt}\right)_{body} = \left(\frac{d\boldsymbol{H}}{dt}\right)_{inertial} + \boldsymbol{\omega} \times \boldsymbol{H} \tag{A1.15}$$

with the cross product:

$$\boldsymbol{\omega} \times \boldsymbol{H} = \begin{pmatrix} i & j & k \\ p & q & r \\ H_x & H_y & H_z \end{pmatrix} = \left(H_z q - H_y r\right)i + (H_x r - H_z p)j + \left(H_y - H_x q\right)k \tag{A1.16}$$

Since the torque equals the derivative form of the angular momentum:

$$\frac{d\boldsymbol{H}}{dt} = \boldsymbol{I} \times \dot{\boldsymbol{\omega}} = \begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \tag{A1.17}$$

$$= \left(I_{xx}\dot{p} + I_{xy}\dot{q} + I_{zz}\dot{r}\right)\hat{\imath} + \left(I_{yx}\dot{p} + I_{yy}\dot{q} + I_{yz}\dot{r}\right)\hat{\jmath} + \left(I_{zx}\dot{p} + I_{zy}\dot{q} + I_{zz}\dot{r}\right)\hat{k}$$

Assuming the rocket is symmetric along xy, yz and xz planes, the angular momentum equations turn to:

$$\boldsymbol{H} = \ \boldsymbol{I} \times \boldsymbol{\omega} = (I_{xx}p)\hat{\imath} + \left(I_{yy}q\right)\hat{\jmath} + (I_{zz}r)\hat{k} \tag{A1.18}$$

Also, rewriting the cross product of eq. A1.16:

$$\boldsymbol{\omega} \times \boldsymbol{H} = \left(-I_{yy}rq + I_{zz}qr\right)\hat{\imath} + (I_{xx}pr - I_{zz}rp)\hat{\jmath} + \left(I_{yy}pq - I_{xx}qp\right)\hat{k} \tag{A1.19}$$

Then, the body reference torque for each axis turns to:

$$\begin{aligned} \sum \boldsymbol{L} &= \left(\frac{dH_x}{dt}\right) = \left(I_{xx}\dot{p} + I_{zz}rq - I_{yy}qr\right)\hat{\imath} \\ \sum \boldsymbol{M} &= \left(\frac{dH_y}{dt}\right) = \left(I_{yy}\dot{q} + I_{xx}pr - I_{zz}rp\right)\hat{\jmath} \\ \sum \boldsymbol{N} &= \left(\frac{dH_z}{dt}\right) = \left(I_{zz}\dot{r} + I_{yy}pq - I_{xx}qp\right)\hat{k} \end{aligned} \tag{A1.20}$$

Simplifying $I_{xx} = I_x, I_{yy} = I_y, I_{zz} = I_z$ and re-arranging:

$$\begin{aligned} \dot{p} &= \frac{\boldsymbol{L}_A + \boldsymbol{L}_p - qr(I_z - I_y)}{I_x} \\ \dot{q} &= \frac{\boldsymbol{M}_A + \boldsymbol{M}_p - rp(I_x - I_z)}{I_y} \\ \dot{r} &= \frac{\boldsymbol{N}_A + \boldsymbol{N}_p - pq(I_y - I_x)}{I_z} \end{aligned} \tag{A1.21}$$

A2 MATLAB Code

```
close all, clear all, clc

% Initial Parameters and Characteristics
m = 567718; % kg
Iy = 296*10^6; % kgm^2
T0 = 10506450; %N
Td = 10506450; %N
F = T0 + Td; % Total Thrust
V = 410.56; % m/s
N_alpha = 3056.34; % kN/rad
M_alpha = 0.3807; % s^-2
M_delta = 0.5726; % s^-2
x_cg = 16.21; % m
x_cp = 39.94; % m
M = 1.4;
h = 10; % km
d = 903.3; % kN

% State Space (*d are values for single input [Pitch])
A = [0 1 0; M_alpha 0 M_alpha/V; (d-F-N_alpha)/m 0 -N_alpha/(m*V)];
B = [0 0; M_delta M_alpha; Td/m -N_alpha/m];
Bp = [0; M_delta; Td/m];
C = [1 0 0; 0 1 0; 0 0 1];
Cp = [1 0 0];
D = [0 0; 0 0; 0 0];
Dp = 0;


% Controllability and Observability check (good if rank = 3)
rank(ctrb(A,Bp))
rank(obsv(A,Cp))

% System creation and Simulink
system = ss(A,B,C,D);
sys_P = ss(A,Bp,Cp,Dp);  % Pitch only system

sys_B = ss(A,Bp,C,Dp);

[b,a] = ss2tf(A,Bp,Cp,Dp);
sys = tf(b,a);

%% PID

% PID gains
P = 157.71;
I = 75.26;
D_PID = 48.55;

pid_k = tf([P,I,D_PID],[1 0]);
```

```
CL = feedback(series(pid_k, sys),1);

bode(sys)
hold on
bode(CL)
legend('plant', 'CL')

%% LQR

% LQR gains
Q = [100000 0 0; 0 150000 0; 0 0 7000];
R = [.1 0; 0 .1];


[K, S, E] = lqr(system,Q,R);

% reference signal with initial conditions
r = [0.05 0 0]';
x0 =[0.012  0 0]';

% separating vertices
o = [1 0 0];
n = [0 1 0];
z = [0 0 1];


%% KF

Qn = 0.1;
Rn = [1 0 0; 0 1 0; 0 0 1];

Nkf = 0;
[kalmf,L,P] = kalman(sys_B,Qn,Rn,Nkf);

%% LQG
Rlqg = 0.1;

[Klqg, Slqg, Elqg] = lqr(sys_B,Q,Rlqg);

%% Simulink
open_system('State')
sim('State')


%% Plotting

% PID

figure
plot(ans.ref(:,1),ans.ref(:,2),'--')
hold on
```

```
plot(ans.pitchOL(:,1),ans.pitchOL(:,2))
hold on
plot(ans.pitchCL(:,1),ans.pitchCL(:,2))
hold off
ylim([-.5 2])
xlim([0 10])
legend('Reference Signal','Open Loop','Closed Loop')
xlabel('Time (s)')
ylabel('Amplitude')
title('Pitch Angle for Closed Loop Response')

% LQR
figure % Pitch
plot(ans.output(:,1),ans.output(:,5),'--')
hold on
plot(ans.lqr_theta(:,1),ans.lqr_theta(:,2))
hold off
title('LQR Pitch angle response')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Pitch angle response')

figure % Pitch Rate
plot(ans.lqr_thetadot(:,1),ans.lqr_thetadot(:,2))
title('LQR Pitch angle rate')
xlabel('Time (s)')
ylabel('Amplitude')

figure % Lateral Drift or Z dot
plot(ans.lqr_zdot(:,1),ans.lqr_zdot(:,2),'r')
title('LQR Lateral Drift')
xlabel('Time (s)')
ylabel('Amplitude')


% PID vs LQR comparison
figure
plot(ans.output(:,1),ans.output(:,5),'--')
hold on
plot(ans.pitchCL(:,1),ans.pitchCL(:,2))
hold on
plot(ans.lqr_theta(:,1),ans.lqr_theta(:,2))
hold off
title('LQR vs PID pitch response')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','PID Pitch angle','LQR Pitch angle')

%KF
figure,
plot(ans.KF_theta(:,1),ans.KF_theta(:,2),'--')
hold on
```

```
plot(ans.KF_theta(:,1),ans.KF_theta(:,3))
axis([0 10 -5 5])
title('Pitch angle vs Estimated angle')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Pitch angle','Estimated angle')

figure,
plot(ans.KF_theta_dot(:,1),ans.KF_theta_dot(:,2),'--')
hold on
plot(ans.KF_theta_dot(:,1),ans.KF_theta_dot(:,3))
axis([0 10 -5 5])
title('Pitch angle rate vs Estimated rate')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Pitch angle rate','Estimated rate')

figure,
plot(ans.KF_Z_dot(:,1),ans.KF_Z_dot(:,2),'--')
hold on
plot(ans.KF_Z_dot(:,1),ans.KF_Z_dot(:,3))
axis([0 10 -5 10])
title('Lateral drift vs Estimated drift')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Lateral drift','Estimated drift')

%LQG
figure,
plot(ans.LQG_theta(:,1),ans.LQG_theta(:,2),'--')
hold on
plot(ans.LQG_theta(:,1),ans.LQG_theta(:,3))
axis([0 10 -5 5])
title('LQG Pitch angle vs Estimated angle')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Pitch angle','Estimated angle')

figure,
plot(ans.LQG_theta_dot(:,1),ans.LQG_theta_dot(:,2),'--')
hold on
plot(ans.LQG_theta_dot(:,1),ans.LQG_theta_dot(:,3))
axis([0 10 -5 5])
title('LQG Pitch angle rate vs Estimated rate')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Pitch angle rate','Estimated rate')

figure,
plot(ans.LQG_z_dot(:,1),ans.LQG_z_dot(:,2),'--')
hold on
plot(ans.LQG_z_dot(:,1),ans.LQG_z_dot(:,3))
```

```
axis([0 10 -5 10])
title('LQG Lateral drift vs Estimated drift')
xlabel('Time (s)')
ylabel('Amplitude')
legend('Reference Signal','Lateral drift','Estimated drift')
```