I, Paul H Guentert, hereby submit this original work as part of the requirements for the degree of Master of Science in Mechanical Engineering.

It is entitled:
A Variable Pitch Quadrotor with Quaternion Based Attitude Controller

Student's name: Paul H Guentert

This work and its defense approved by:

Committee chair: Manish Kumar, Ph.D.

Committee member: Rajnikant Sharma, Ph.D.

Committee member: David Thompson, Ph.D.
A Thesis
entitled

A Variable Pitch Quadrotor with Quaternion Based Attitude Controller

by
Paul Hans Guentert

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the Master of Science Degree in Department of Mechanical and Materials Engineering

Dr. Manish Kumar, Committee Chair

Dr. Rajnikant Sharma, Committee Member

Dr. David Thompson, Committee Member

Paul Orkwis, PhD, Interim, Dean
College of Graduate Studies

University of Cincinnati
March 2017
Variable Pitch Quadcopter (VPQ) platforms offer unique capabilities over traditional quadcopters by producing thrust through change in blade pitch instead of change of rotor speed. A helicopter swash plate and servo controls the pitch actuation. This allows VPQ to achieve near instantaneous thrust change as well as inverted thrust. This versatility makes VPQ able to achieve aggressive inverted flight trajectories and aggressive transition maneuvers such as the “tic-toc” maneuver where the VPQ transitions back and forth between normal and inverted flight which is unachievable by tradition quadrotors. There are no existing commercial autopilots for the VPQ. This thesis focuses on construction of the VPQ quadcopter, building an autopilot system, and observes the requirement for an attitude control system that is singularity free for continuous rotation caused by inverted flight. A simplified motor model based on Blade Element Theory and Momentum Theory is used to describe the thrust and torque generated by the change in pitch of the blades. This simplified model has irrational yaw terms and therefore calculated through the derivative in a virtual control allocation. A quaternion attitude system is used in the description of orientation for its lack of rotation singularities. However, a new control law is needed to determine desired angular jerk for use in the virtual control allocation. A cascaded controller is constructed that implements LQR position controller and geometric attitude tracking to produce aggressive attitude tracking. This controller shows robust tracking for following different trajectories in simulation.
## Contents

**Abstract** .................................................. ii

**Contents** ................................................. iv

**List of Tables** ........................................... vi

**List of Figures** .......................................... vii

**List of Abbreviations** ................................... viii

**List of Symbols** .......................................... ix

1 **Introduction** ............................................. 1
   1.1 Motivation ........................................... 1
   1.2 Purpose .............................................. 2
   1.3 Contributions ....................................... 4

2 **Literature Review** ....................................... 5
   2.1 Vertical Takeoff and Vertical Landing Vehicles .. 5
   2.2 Quadrotors .......................................... 6
      2.2.1 Modeling Quadrotors ......................... 7
   2.3 Quaternions ......................................... 8
      2.3.1 Quaternion Control Law ....................... 11
   2.4 Variable Pitch Quadrotors ......................... 13
   2.5 Trajectory ........................................... 14

3 **Dynamic Model** ......................................... 16
   3.1 Quadrotor Flight Dynamics ......................... 16
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Reaper 500 Properties</td>
<td>31</td>
</tr>
<tr>
<td>4.2</td>
<td>Blade Properties</td>
<td>32</td>
</tr>
</tbody>
</table>
List of Figures

1-1 Heavy Lift Concept Drawings .................................................. 2
2-1 Axis-Angle Rotation Quaternion Concept .................................. 9
3-1 Quadrotor Dynamics ............................................................... 16
3-2 Quadrotor Dynamics ............................................................... 16
3-3 Coordinate Frame System ......................................................... 20
4-1 Simulation Control Structure ................................................... 33
4-2 Attitude Tuning Results ............................................................. 38
5-1 Attitude Tuning Results ............................................................. 42
5-2 Attitude Tuning Results ............................................................. 43
5-3 Circle Trajectory ................................................................. 44
5-4 Circle Trajectory States ........................................................... 45
5-5 Figure 8 Trajectory ................................................................. 46
5-6 Figure 8 States ................................................................. 46
6-1 Reaper 500 Platform ............................................................... 47
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>UAV</td>
<td>Unmanned Aerial Vehicle</td>
</tr>
<tr>
<td>ESC</td>
<td>Electronic Speed Controller</td>
</tr>
<tr>
<td>WPN</td>
<td>Way Point Navigation</td>
</tr>
<tr>
<td>DF</td>
<td>Differential Flatness</td>
</tr>
<tr>
<td>VTOL</td>
<td>Vertical Take Off and Landing</td>
</tr>
<tr>
<td>CW</td>
<td>Clockwise</td>
</tr>
<tr>
<td>CCW</td>
<td>Counter-Clockwise</td>
</tr>
<tr>
<td>BET</td>
<td>Blade Element Theory</td>
</tr>
<tr>
<td>VRFT</td>
<td>Virtual Reference Feedback Tuning</td>
</tr>
<tr>
<td>RC</td>
<td>Remote Control</td>
</tr>
<tr>
<td>RPM</td>
<td>Revolutions per Minute</td>
</tr>
<tr>
<td>UAS</td>
<td>Unmanned Aerial System</td>
</tr>
<tr>
<td>3D</td>
<td>Three dimensional</td>
</tr>
<tr>
<td>6-DOF</td>
<td>Six Degrees of Freedom</td>
</tr>
<tr>
<td>VPQ</td>
<td>Variable Pitch Quadrotor</td>
</tr>
<tr>
<td>FTPR</td>
<td>Flatness-based Trajectory Planning/Re-planning</td>
</tr>
</tbody>
</table>
List of Symbols

\( \eta \)          Quaternion Scalar
\( \epsilon \)       Quaternion Vector
\( Q \)            Quaternion Pair Combination
\( Q^* \)        Quaternion Conjugate
\( R(Q) \)         Rotation Matrix Transformation
\( S(Q) \)         Skew Matrix Transformation
\( \Phi(Q) \)     Rate Matrix Transformation
\( C_{Ti} \)     Coefficient of Thrust
\( F^i \)         Force Inertial-Frame
\( F^b \)         Force Body-Frame
\( m \)            mass
\( \mathbf{r} \)   Trajectory
\( \omega \)       Angular Velocity
\( p, q, r \)     Angular Velocity about the primary axes in the body frame
\( \mathbf{i} \)  reference frame
\( \mathbf{b} \)  body frame
\( \tau \)         Torque
\( l, m, n \)     Torque about the primary axes in the body frame
\( x, y, z \)     Coordinate Frame in the Inertial Frame
\( \mathbf{I} \)   Moment of Inertia Matrix
\( J_x, J_y, J_z \) Moments of Inertia
\( C_t \)          Coefficient of Thrust
\( \rho \)         Density of Air
\( T_i \)         Thrust
\( R \)            Blade Radius
\( V_{tip} \)     Velocity of Blade Tip
\( C_{la} \)     Coefficient of lift vs Angle of Attack
Chapter 1

Introduction

1.1 Motivation

Unmanned autonomous vehicles (UAVs) have expanded from simple toys and oddities to an industry of autonomous functionality and complex problem solving. Quadrotors, a type of UAV, have enabled short-term low altitude dexterous exploration, monitoring, and task management. To expand the capabilities of quadrotors, scalability offers the opportunity to expand the missions that these platforms can perform. Through results from a customer survey, requirements for a heavy lift vertical takeoff/vertical landing vehicles that use heavy fuel show a possible demand for quadrotors if they were scalable. However, the torque requirement to overcome the inertia of the blades to produce responsive control makes the scalability of quadrotors difficult. Variable pitch rotors (VPQ) provide a partial solution by generating thrust through change in pitch of the rotors, thus solving the issue of overcoming great inertial forces to reduce or increase the rotational speed of the rotors. Investigation into VPQs shows little work into simulation robust enough for scalability. Scalability would offer the only opportunity to build a heavy lift/heavy fuel quadrotor platform.
Examining quadrotors from a technical perspective in terms of hardware and coding, the necessity for theoretical fundamentals such as reference frames, linear control, and non-linear control methods to discover autonomous functionality is apparent. As quadrotors are under-actuated systems, they are highly nonlinear systems operating in 6 degree of freedom environments making them a challenge to model. Cursory investigation into VPQs unveiled a research domain of versatile functionality that VPQs could achieve with instantaneous inverted thrust. As noted in [7], “traditional pod-and-boom autonomous helicopters have demonstrated agile maneuvers outside of the flight regime of small fixed-pitch quadrotors”. These features implemented on a quadcopter should increase the range of achievable trajectories of the traditional quadcopter. Such uniqueness of this platform offers a plethora of unknown applications. Initial experimentation began with a VPQ to see if autonomous way-point navigation could be achieved with a traditional quadrotor controller. Initial flight tests proved successful in terms of attitude stabilization and developed into a path for proving theoretically that a VPQ could fly with autonomous way-point navigation using existing quadrotor control methods with a few modifications.

1.2 Purpose

Initiated by the challenge to construct a VPQ that flies using way-points, the purpose of this research investigation is to pursue the construction of a VPQ controller that can use way-points and trajectories. As the initial experimentation proved successful, it was deemed worthwhile
to take a systematic approach to modeling the variable pitch quadcopter to fundamentally understand and achieve the features of inverted flight. The process began with developing a mathematical model that describes the flight dynamics of the VPQ. The initial experimental implementation focus uses a Pixhawk running PX4 firmware, an open source platform that is versatile and offers a good structure for closed loop control. It proved useful as a working framework for development. Simulation of different controller methods are used as proof of concept and perform analysis of controller performance. The first approach used a traditional quadrotor controller that takes roll, pitch, yaw, and thrust commands and mixes the control inputs to individual motor commands. These motor commands are manipulated to produce similar pitch commands. The control signal that operates on a normalized range of 0-1 was readjusted to the PWM range of the servos. This approach was successfully implemented on hardware.

A simulation model was constructed to model the variable pitch dynamics and demonstrate the required control inputs. This systematic approach uses blade element theory to model the desired pitch angle in simulation. The first attempted simulation approach used Euler angles with PID position, attitude, and attitude-rate controllers. This controller performs well but has points of failure during inverted flight because singularities occur in Euler-based controllers. A singularity free solution is required to pursue inverted flight features of VPQs and makes a quaternion-based control method appropriate. The final controller design combines innovative methods from quaternion-based controllers, blade element theory based motor models, and allows for differential-flatness-based control schemes. The simulation model follows the back-stepping control process for variable pitch control presented in [12]. This model describes the motor model in Blade Element Theory to derive the force of the blades of the quadrotor. In this process, the derivative of the forces acting on the rigid body is performed. This operation results in demanding an input of angular jerk. The second order error dynamics process used to calculate angular jerk is demonstrated in [14].

The goal of the new simulation model is to improve the above process by implementing a quaternion attitude system to achieve inverted flight trajectories. This new system is no longer reliant on the irrational concept of angles [5]. Instead, the system will derive desired rotations
using geometric attitude tracking techniques. A Linear Quadratic Regulator is implemented to track aggressive trajectory maneuvers.

1.3 Contributions

The main contributions of this work is the process and construction of a VPQ simulation using a cascaded controller design. A dynamic motor model in conjunction with quaternion based attitude controller. A PID attitude controller is implemented to produce the desired angular acceleration as inputs to the motor model. A linear quadratic regulator is implemented as a position controller and proven in simulation successful in following aggressive trajectories. Experimental proof of quad dynamics with existing variable pitch frames is demonstrated. This work is different from [5] because it uses a simplified motor model. It is also separate from [14] because it uses quaternion based attitude control.
Chapter 2

Literature Review

2.1 Vertical Takeoff and Vertical Landing Vehicles

Aerial vehicles have been the subject of research endeavors with breakthroughs cascading through both industry and society. Vertical takeoff and vertical landing craft, aerial vehicles characterized by their ability to vertically takeoff from a stationary position without use of a runway, have been studied since the 19th century but development was limited until the invention of the engine (as noted by Thomas Edison). Unlike typical winged aircraft, the takeoff and landing sequence requires downward generated thrust from a propeller or redirected thrust from a jet engine. Innovators pioneered the development into today’s companies: Bell, Sikorsky, Boeing-Vertol helicopters; VTOL craft; to name a few. These craft would enable closer proximity to destinations and landing in areas which airplanes could not reach due to the limitations of large areas required for runway access.

Vertical take-off and vertical landing vehicles are a subgroup of aerial vehicles that serve a particular mission which gliding or winged aircraft cannot achieve, operating at low altitude to dexterous operation. The operational range of the VTOL craft is typically much shorter than the winged aircraft and also has less lift efficiency than an airplane. The most notable –manned VTOL craft is the helicopter with long blades that generate thrust as they are rotated at different pitch angles. There are ranges of hybrid tilt rotorcraft such as the Osprey, which seek to fill the operational capacity of a winged aircraft and the dexterity of a helicopter with a transitioning system from vertical take-off to horizontal winged flight. These craft serve particular operation purposes that airplanes cannot achieve. The quadrotor has demonstrated itself as a
dependable and dexterous development platform. Therefore, it has become the primary focus of autonomous operational research in the 21st century.

### 2.2 Quadrotors

Multirotor crafts have been pursued since the 18th century. The multirotor family contains vehicle platforms such as coaxial helicopters, tricopters, hexacopters, and many more configurations of multiple rotors or blades. A subset of this multirotor group is the quadrotor. Louis Breguet developed one of the first quadrotor-like platforms in 1907. This large, four-rotor helicopter was ultimately abandoned by the U.S. military due to its mechanical complexity. Multirotor research was rarely seen until the 21st century with the resurgence propelled by strides in the micro-electronics industry. The success of quadrotors would ultimately come with the advent of electric motors and integrated chips. The reduced cost of micro-electronics brought the entry cost of research and development down which allowed for less costly ventures in multirotor craft research for autonomous functionality.

The modern definition of quadrotors is a vertical takeoff, vertical landing vehicle that uses four propellers or rotors mounted near the perpendicularity of the plane created by the arms on which they are mounted. Although ever-evolving, these systems typically use four electric motors opposed to heavy fuel based engines. Each motor is independently controlled by an electronic speed controller, which receives control signals from a central flight controller. Each rotor produces a thrust perpendicular to the plane of the arms, and a torque along the x-y plane. As each motor is controlled independently, combinations of forces and toques allow the quadrotor to fly dexterous trajectories at low altitudes.

Multirotor craft have distinct advantages compared to other VTOL craft. The endurance of a rotor aircraft in the most simplistic terms comes down to rotor length: the slower the revolutions per minute (RPM) of the engine, the greater the endurance of the aircraft. With a slower spinning rotor due to increased size of the rotor, helicopters have a better hover efficiency than quadrotors of the same capacity. However, the inertia becomes greater as the rotor is increased. Four rotors of a multirotors can generate the equivalent amount of thrust to a similiar size
helicopter but must spin faster. The series of separate rotors gives the quadrotor a similar lift capability of a large rotor but rotors have reduced inertia. Coaxial setups allow for the same amount of footprint of a non-coaxial vehicle and can produce up to 40 percent more thrust with the second blade beneath. Helicopters have variable pitch blades, unlike traditional quadrotor, and therefore do not need to overcome the greater inertia of a larger blade. [25]

Quadrotors are particularly popular in research, industry, and personal life due to the lowering cost of hardware and level of control. The smaller blades of a quadrotor are safer as the inertia of the blades is less damaging than a helicopter. Research has now been heavily focused on autonomous control, path planning, and decision-making. As the size of multirotors is scaled to the maximum capacity of 50 lbs per the FAA rules, the endurance of electric motor based multirotors will be limited by the battery capacity. Hexacopter or octacopter configurations such as the Falcon 8 by Intel or the Typhoon H Hexacopter show how industry will add more rotors to increase lift capability instead of adding larger rotors to a quadrotor configuration.

2.2.1 Modeling Quadrotors

In controls, a model is a mathematical representation of a system that can be analyzed to develop control laws for the system. The model can demonstrate the reaction to inputs or a combination of inputs to the system. Often, with complex models, highly nonlinear models are simplified for developing a controller. It is modeled mathematically though a combination of kinematics and classical mechanics. Interesting, the quadrotor contains 4 actuators but operates in a 6 degree of freedom environment. The quadrotor therefore is defined as under-actuated with a total of 3 moments, and one thrust vector controlling the system in the 6 degree of freedom environment. The nonlinearity is described by the coupled rotational and translational motions due to under actuation. [12] This requires the quadcopter to provide its own dampening force (in terms of control law) as there are no opposing forces such as friction which interact with the quadrotor.

Most aspects of quadrotor control have been highly investigated over the past few years from system modeling, feedback control, and state estimation. The University of Pennsylvania Grasp lab has been a large contributor to autonomous multirotor exploration and mapping. [17]
explores the multitude of previous and current quadcopter control structures and methods. ETH Zurich has published a large quantity of work relating to multi-rotors with demonstrations from cooperative task management to autonomous sensing such as "Collaborative Aerial Robotic Workers" and topics such as search and rescue. Brigham Young University covers quadcopter state estimation modeling with and linearized control method with vision assisted state estimation.

There are many methods of quadcopter control under the branches of linear, nonlinear, robust, and adaptive control. Linear control starts with building a model and linearizing the equations of motion and motor model at a point of equilibrium, which transfer functions and pole placement methods can be used to develop controllers gains. The difference between linear and non-linear systems is described by the superposition principle, if an input $A$ to a system produces response $X$ and input $B$ to a system produces response $Y$ then input $(A + B)$ produces a system response $(X + Y)$. A nonlinear system describes a mathematical model which contains nonlinear functions. In controls, these nonlinear functions are linearized to build a controller.

Quadcopter model, or plants use non-linear dynamics to simulate reaction of control inputs. A cascade controller design breaks the controller into separate controller segments and is easy to visualize. The cascading control sequence starts with a position controller that determines current position and desired position, the output is typically a desired acceleration vector. The attitude controller translates this acceleration vector to desired orientation and compares it to the current orientation of vehicle to find a desired attitude orientation as well as desired thrust. The error between the current attitude orientation and desired attitude orientation and the derivative of error is used to calculate the desired angular acceleration. This angular acceleration can be used to derive torque from change in momentum. Lastly, the angular jerk is multiplied by the moment of inertia.

### 2.3 Quaternions

Measuring orientation of a rigid body in three-dimensional space is crucial to deriving the equations of motion and kinematics. The most well-known system, Trait-Bryan angles, (roll,
pitch, yaw angles), use a sequence of rotations about a fixed point of the rigid body to describe change in attitude in respect to a reference frame as either extrinsic or intrinsic. Euler was able to further realize that two rotations could be described as rotation about a fixed, arbitrary axis. With the invention of matrix algebra, orthogonal rotation matrices could be generated and showed the Euler Angle-Axis rotation vectors were actually the eigenvector of the rotation matrices. Rotation matrices are used in robotic applications to describe the translation and rotation between reference frames.

Euler rotations face an issue called Gimbal Lock, where a single rotation is broken into three constitutive rotations in which the proceeding rotation occurs about the new frame’s most recent rotation. This can be visualized when the orientation of an object is pointed straight up or straight down, two rotation axes will be aligned and further rotations will not rotate the shortest path. Sometimes this problem can be fixed by changing the rotation order. Euler angles face another issues associated with gimbals when the rotation order causes the rotation not to take the most efficient path. Euler angles use trigonometric functions such as tangent, which carry ambiguity at 90 and 270 degrees of rotation. Quaternions overcome ambiguity, complexity, and computational intensity by encoding a rotation into a single scalar, vector combination.

(a) Euler Rotation Axis
(b) Quaternion Concept

Figure 2-1: Axis-Angle Rotation Quaternion Concept

One method of visualizing quaternions is to use the rotation axis or Euler rotation theorem
which any rotation in the SO(3) can be represented by an arbitrary axis and rotation about that axis, this transformation is shown in equation 2.1. The rotation or change in pose of an object can be described as a unit vector which the object is rotated about quaternions are a vector based system but in the 4 dimensional domain. Quaternions encode a single rotation $\theta$ around a single arbitrary axis $\vec{u} = <u_x, u_y, u_z>$ opposed to 3 rotation matrices. The resulting vector is represented a form of real components $a, b, c, d$ and quaternion units $i, j, k$ in equation 2.1.

$$Q = e^{\frac{\theta}{2}(u_xi + u_yj + u_zk)} = \cos \frac{\theta}{2} + (u_xi + u_yj + u_zk) \sin \frac{\theta}{2} = a + bi + cj + dk$$  \hspace{1cm} (2.1)

When quaternions are multiplied, the complex components, $i, j, k$, will produce new imaginary components based on the order and combination of the complex components. This is the reason for the non-commutative property that exists for quaternions.

$$i^2 = j^2 = k^2 = ijk = -1$$  \hspace{1cm} (2.2)
$$ij = -ji = k$$  \hspace{1cm} (2.3)
$$jk = -kj = i$$  \hspace{1cm} (2.4)
$$ki = -ik = j$$  \hspace{1cm} (2.5)

The inverse dynamics of Euler angles also carry singularities. Quaternion based kinematics are numerically stable and are free of singularity. Quaternion algebraic operations are different from Euler based systems. Instead of the final position being a sum of rotations, the quaternion representation is a product of subsequent rotations as shown equation 2.7. More formal definitions and derivations can be found in resources [16].

$$P^* = Q \otimes P \otimes Q^{-1}$$  \hspace{1cm} (2.7)

Group theory allows the description of number system properties through grouping symmetric properties. Often quaternions are described apart of the $SO(3)$ group which denotes the
special orthogonal group. The unit quaternion corresponds of the $su(2)$ or special unitary group with the properties of 2:1 homomorphism from unit quaternion to the $SO(3)$ group. This states that the unit quaternion covers the sphere of rotation twice over. Since unit quaternion group $su(2)$ double covers the $SO(3)$, unit quaternions are not unique, each $\pm q \in S^3$ corresponds to the same attitude. [2] This aspect of ambiguity in quaternions must be handled in order to achieve an asymptotically stable yet robust attitude control representation.

### 2.3.1 Quaternion Control Law

A control law is a mathematical relationship of inputs to outputs. It is standard to have the inputs be a state variable or a function of a state variable. The output is either an input to the system itself or another controller apart of the system. An attitude controller translates desired orientation, current orientation, desired angular velocity, and current angular velocity to desired moments to act on the rigid body. The attitude controller can be represented in many different formats and is sometimes broken into two separate controllers, one for desired attitude orientation and another for desired angular velocity. The effectiveness of a control law is examined by its ability to handle uncertainty and over all robustness to handling environments outside of its designed operating environment.

Quaternion attitude based control has been investigated for its robustness and usefulness in aerospace as well as computer image processing but not typically in controls engineering due to nonlinearity. Aerospace applications have transitioned away from gimbal measurement systems to “strapped down” methods which integrate rotation and acceleration to find orientation and position. Quaternion control methods have shown ability to overcome singularities of angle based methods, as well as reduce computational complexity as seen in matrix rotation methods.

Proof of attitude controller stability is built on small angle approximations assumption. [22] and [23] have forayed into this assumption in seeking to develop robust global attitude tracking control law based on quaternion representation of attitude. This research exacts singularities caused from using memoryless quaternion feedback control laws and proposes solutions which handles an issue known as the unwinding phenomenon. This phenomenon is described in the
situation where the quaternion will rotate about the axis of rotation through the larger angle. They use a hybrid control law which incorporates the sign of the quaternion scalar value. This insures that the quaternion error vector is rotating appropriately to the desired angular velocity term.

A more formal PID quaternion controller approach is presented by [28]. This method seeks to offer the same resolution presented by [22] using a filtered command equation which can compute the analytical derivative of the quaternion using the transformation of angular velocity without differentiation. The method enables implementation of back-stepping control in quaternion based feedback control from analytical derivation of the quaternion and relationships to angular velocity. [5], [11] works with full quaternion attitude control for quadrotors. [33] explores the complexity of second order derivatives of quaternion error for proof of Lyaponuav stability using velocity measurements in the derivation of angular velocity.

Robustness of quaternion control law can be tested through an adaptive controller type, which manages a system in changing uncertainty such as changing moments of inertia due to fuel consumption. This control approach is tested by [4] in a controller with the simple quaternion error tracking that lacks full state velocity terms updates. [9] takes an optimal control approach to quaternion attitude control to find the minimal quaternion rotation using Hamiltonian methods. This method is superseded by methods presented in [21] which can use resulting inertial force and body force vectors to determine the best transition. This process was ultimately adopted in the [5].

The attraction to VPQs is the instantaneous inverted thrust capability which means that VPQs are able to exhibit aggressive inverted flight characteristics and aggressive transitions. Switch mechanisms for inverted flight orientation must reverse the resultant trajectory force vector. [5] has identified switching mechanisms related to the quaternion representation in the scalar value.

With the quaternion representation rotational group double covering the SO(3), rotating an object 360 degrees results in half a quaternion rotation. A full quaternion rotation results in a 720 degree rotation. LQR is the most common control method is shown to work with aggressive trajectory following for quaternion based designs. There are many methods for implementing
LQR controls,\cite{32}, a linearized quaternion based control is derived but only locally stable but is proven to be a robust pole assignment design. Coupling LQR control with quaternion attitude control is further investigated by \cite{15} for aggressive trajectory tracking with a micro-quadrotor UAV. However, \cite{15} did not fully include the global stability of quaternion control where the sign of the quaternion scalar is included to denote the correct rotation. The method proposed in this paper uses a LQR position controller design.

### 2.4 Variable Pitch Quadrotors

The VPQ is characterized as a multirotor with varying pitch blades that are controlled by four independent swash plates manipulated servos. In theory the change in thrust is instantaneous opposed to a traditional quadrotor that must overcome the inertia of the blade. The variable pitch system also allows for instantaneous reverse thrust for applications such as inverted flight or braking maneuvers.

VPQs offer many advantages to the multi-rotor craft platform family. Scaling quadrotor platforms can be limiting due to the current required size of batteries and motors. Variable pitch platforms allow for a simple mechanical design and single motor instead of four independent motors. Overcoming the inertia control allocation problem, use of heavy fuel based engines (which can refueled, unlike battery systems) is possible. While flight maneuvers are still being explored, "traditional pod-and-boom autonomous helicopters have demonstrated agile maneuvers outside of the flight regime of small fixed-pitch quadrotors" \cite{7}.

A VPQ is a type of quadrotor configuration studied by the notable few from MIT \cite{6}, \cite{7}, \cite{5} Nanjing University \cite{28}, University of Cape Town \cite{26} and Indian Institute of Technology Kanpur \cite{14}. This collection of work explores different control methods and novel implementations of Blade Element Theory to model VPQs.

\cite{5} is the seminal study of small scale VPQs. This work implements a variable pitch quaternion based controller on a quadrotor platform with four independent motors as well as four independent pitch servos. The control structure models the entire vehicle with actuator modeling and experimental testing. The simulation implements a motor model that was derived and then
correlated with experimental test data. The experimental test data allowed derivation of a con-
troller that correlated voltage with thrust. The quaternion-based controller in [5] demonstrated
inverted flight trajectories without the issues familiar with Euler angle singularities.

[14] uses cascading control coupled with Blade Element Theory with Momentum Theory
to model the VPQ. This work uses an Euler Angle based attitude controller scheme and re-
quires the derivative of moments to calculate the change of pitch angle. The method allows the
implementer to apply it to a range of different blade configurations using known blade lift and
drag coefficients. This method was verified in simulation using real known platform data of a
Reaper 500 sold by Hobby King.

While pitch change can generate thrust, the change in rotational speed of the rotors can
as well. Therefore, there exist two separate inputs for control can affect lift of the VPQ, the
RPM of the blades and angle of attack of the blades. [28] focuses on energy efficient control of
the variable pitch platform, building a controller that determines the best RPM and pitch angle
based on environmental and state conditions.

2.5 Trajectory

The quadrotor operates in a six degree of freedom environment with four actuators making
it an under-actuated system. Therefore trajectories must be constrained to the feasible operation
of the under-actuated system. To generate these trajectories, the system must be shown to be
differentially flat. A differentially flat system can express the states and inputs as algebraic
functions of flat outputs and their subsequent derivatives [30].

Achievable trajectories are limited by quadrotor dynamics as most position control methods
are limited by the range of stability of the attitude controller which is typically stable for small
angle approximations. A few researchers have attempted to build new control structure which
can take in more desired states of the quadrotor based on the trajectory generation.

Attempting to achieve coordinate free tracking scheme reduces singularities and removes
the need for local coordinate systems [18]. The usefulness of nonlinear geometric tracking
methods is proved for global asymptotic tracking in the case where the rotation is not 180
Minimum snap trajectory generation and control for quadrotors uses this process to extend the ability of aggressive trajectory following. This concept is further extended to trajectory design and control for aggressive formation flight with quadrotors in [29].
Chapter 3

Dynamic Model

3.1 Quadrotor Flight Dynamics

A VPQ uses four rotors which generate lift and drag. Varying the pitch of the rotors independently can produce different resultant force and moments upon the quadcopter. The resultant forces acting on the quadcopter are thrust and three moments in the body frame are model
about each of the primary axes in the orthogonal coordinate frame. The force vector can be broken into a thrust vector perpendicular to the x-y plane of the quadrotor and moment vector in-line with the x-y plane. At level flight and stable hover, each thrust vector is equal and each moment is demonstrated in 3-1a. To produce a roll to port, the thrust vectors \( F_3 \) and \( F_4 \) are increased while \( F_1 \) and \( F_2 \) are decreased as shown in 3-1b. It can be noted that the increased moment in rotor \( M_3 \) is negated by \( M_4 \), as well as the decreased drag moment one with drag \( M_2 \).

It can be inferred to create a negative pitch that \( F_1 \) and \( F_4 \) are decreased while \( F_2 \) and \( F_3 \) are increased. In 3-2a adjacent rotors are manipulated to induce yaw. To yaw to port, \( F_1 \) and \( F_4 \) are decreased while \( F_2 \) and \( F_4 \) are increased. The resulting force creates a yawing moment while maintaining enough lift force to hold altitude.

### 3.2 Quaternions Rotations

Often, mathematical systems used in the description of orientation contain singularities which can cause failure in the system. Robust design seeks to utilize systems which reduce singularities. The singularities of Euler-based pose tracking have already been mentioned in 1.3. In the following sections, resulting force vectors will be derived to produce a resulting orientation which a vector system is used to describe angular velocity in the body frame. The transition from a force vector to resulting angle based system and then back to vector based system seems unwieldy. A quaternion based system which uses linear algebraic like processes can be used to reduce the unnecessary conversions.

The quaternion system is defined a combination of scalar and vector of three complex numbers. The scalar is represented as:

\[
\eta = q_w
\]  
(3.1)

The vector of complex numbers is described in equation 3.2.
\[\epsilon = \begin{bmatrix} q_x \\ q_y \\ q_z \end{bmatrix} \quad (3.2)\]

The ordered pair a quaternion representation \(Q\) in 4 dimensional space. It is represented in the form of equation 3.3

\[Q = \begin{bmatrix} \eta \\ \epsilon \end{bmatrix} \quad (3.3)\]

The quaternion conjugate, 3.4 defines the reversal of the complex elements in quaternion or reverses the direction of the quaternion. The conjugate can be used to extract the scalar and vector parts of the quaternion.

\[Q^* = \begin{bmatrix} \eta \\ -\epsilon \end{bmatrix} \quad (3.4)\]

The quaternion normal operation is described in equation 3.5

\[\|Q\| = \sqrt{q_w^2 + q_x^2 + q_y^2 + q_z^2} \quad (3.5)\]

To satisfy the properties of symmetry for the SO(3) group, the rotation quaternion must be a unit quaternion, \(Q\) and must be satisfied through equation 3.6

\[\|Q\| = 1 \quad (3.6)\]

The inverse of the quaternion is the conjugate divided by its norm. Thus, the quaternion is assumed to be a unit quaternion and quaternion conjugate \(Q^*\). Since the controller is constructed with quaternion rotations, it can be assumed that all quaternions used in rotation sequences are unit quaternions.

\[Q^{-1} = \frac{Q^*}{\|Q\|} \rightarrow Q^{-1} = Q^* \quad (3.7)\]

The derivation shown in [28] equations 3.8 3.9 3.10 3.11 use the same representation.
Equation 3.8 represents the skew matrix which can be used as to represent the cross products of vectors as matrix multiplication.

\[
S(x) = \begin{bmatrix}
0 & -x_3 & x_2 \\
x_3 & 0 & -x_1 \\
-x_2 & x_1 & 0
\end{bmatrix}
\]  \hspace{1cm} (3.8)

The rotation matrix format is useful for transformation operations from inertial frame to body frame and vice-versa. Equation 3.9 is a representation for determining the rotation matrix from the quaternion state. \(I\) represents a 3x3 identity matrix:

\[
r(Q) = I_{3 \times 3} - 2\eta S(\epsilon) + 2S(\epsilon)^2
\]  \hspace{1cm} (3.9)

The quaternion cross product is separate from the linear algebra cross product operation and holds distributive and associative, but not communicative properties. 3.10 represents quaternion multiplication \(\otimes\), essentially the sum of two rotations.

\[
Q \otimes P = \begin{bmatrix}
\eta_Q\eta_P - \epsilon_Q^T \epsilon_P \\
\eta_P \epsilon_Q - \eta_Q \epsilon_P - \epsilon_Q \times \epsilon_P
\end{bmatrix}
\]  \hspace{1cm} (3.10)

The rotation property of the quaternions is such that the rotation \(Q\) and the rotation \(P\) when combined is a resulting rotation matrix of the quaternion product of the quaternion \(Q\) and \(P\) as shown in 3.11

\[
R(Q)R(P) = R(Q \otimes P)
\]  \hspace{1cm} (3.11)

The operation to determine the quaternion error, \(\tilde{Q}_b\), described in the body frame, or the vector relating the error from the current quaternion orientation, \(Q_i\), to the desired orientation, \(Q_{id}\), is in the following equations 3.12 and 3.13 where \(\tilde{Q}\) is quaternion error.

\[
Q_{id} = Q_i \otimes \tilde{Q}_b
\]  \hspace{1cm} (3.12)
\[ \dot{Q}^b = Q_i^* \otimes Q_d^* \]  

(3.13)

The rate transformation matrix of the \( \dot{\omega} \) to the rate quaternion used for solving the ODE in the dynamic equations is represented as a function of the quaternion in 3.14.

\[ \Phi = \begin{bmatrix} S(\epsilon) + \eta I \\ \epsilon^T \end{bmatrix} \]  

(3.14)

### 3.3 Reference Frames

Reference frames with respect to orientation are mentioned in Chapter 1, however, they play a bigger role when describing position. The two common orthogonal coordinate frames used to describe position and pose are the inertial frame and body frame. They are denoted as subscripts \( i \) and \( b \). The inertial frame can be described as the world frame and use the cardinal directions North, East, and Up in right hand convention. The body frame is fixed to the center of gravity of the object and aligned to the principal axes of rotation which allows simplification of the equations of motion. The inertial frame is typically used to describe displacement, velocity, and acceleration while orientation, angular velocity, and angular acceleration is described in the body frame. Accelerometers measure changes in orientation with respect to the
body frame. Their signals are integrated to determine pose and position in the body frame. The relationship between the body frame and inertial frame is used to describe changes in pose. Angular rate sensors called rate gyros measure the changes in angular velocity. Modeling the different states of a quadrotor in these two reference frames greatly simplifies the equations of motion and dynamics. Frame rotation enables a translation between the inertial and body frame axis. The quaternion vector transformation, equation \(3.15\) shows the kinematic equation and relationship velocity from the body frame to the inertial frame.

\[
\frac{d}{dt} \begin{bmatrix} 0 \\ r^i \end{bmatrix} = Q^* \otimes \begin{bmatrix} 0 \\ v^b \end{bmatrix} \otimes Q
\]  

(3.15)

The kinematic equation for the relationship of the quaternion time derivative and angular velocity is shown in \(3.16\)

\[
\dot{Q} = \frac{1}{2} \Phi(Q) \omega
\]

(3.16)

3.4 Dynamics

The quaternion vector rotation property can be used with Newton’s second law of motion to derive the translational dynamics in the inertial frame as shown in \(3.17\) where \(\ddot{r}^i\) denotes the acceleration in the inertial frame, and \(F^b\) force in the body frame.

\[
\begin{bmatrix} 0 \\ \ddot{r}^i \end{bmatrix} = \frac{1}{m} Q^* \otimes \begin{bmatrix} 0 \\ F^b \end{bmatrix} \otimes Q - \begin{bmatrix} 0 \\ g^i \end{bmatrix}
\]

(3.17)

Euler’s equations of rigid body dynamics describe the resulting torques acting on the body in 3 dimensional space. It is a function of the angular velocity, angular acceleration, and the principal moments of inertia. The moment of inertia is the ratio of angular momentum to the angular velocity about the corresponding principle axis. The time derivative of the angular momentum equation is the resultant torque. This is important as quadrotors are modeled as a rigid body in 3 dimensional space with 4 torques and thrust vectors that are applied to the rigid body at some length from the center of mass. The vector form can be expressed in \(3.18\) with \(\tau\)
denoting torque, \( \mathbf{I} \) denoting the moment of inertia, and \( \omega \) angular velocity.

\[
\tau = \mathbf{I} \cdot \dot{\omega} + \omega \times (\mathbf{I} \cdot \omega) .
\]

The angular acceleration can be solved as function of torque in \( \text{3.19} \)

\[
\dot{\omega} = \mathbf{I}^{-1} (\tau - \omega \times (\mathbf{I} \omega)) \tag{3.19}
\]

Angular rates describe the angular velocity \( \mathbf{w} \) in the body frame \( b \) are represented in equation \( \text{3.20} \)

\[
\omega^b = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.20}
\]

The inertia matrix is denoted as \( \text{3.21} \)

\[
\mathbf{I} = \begin{bmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{bmatrix} \tag{3.21}
\]

Applying the orthogonal coordinate system to \( \text{3.19} \) the torque about each axis can be re-defined using \( \text{3.21} \) and \( \text{3.20} \) to form \( \text{3.22} \). The moments of inertia are fixed principle moments of inertia in the body frame and their for are easier to manage where \( l, m, n \) denote the torque about the X, Y, and Z axis in the body frame.

\[
l = J_x \dot{p} + (J_z - J_y) qr
\]

\[
m = J_y \dot{q} + (J_x - J_z) rp
\]

\[
n = J_z \dot{r} + (J_y - J_x) pq \tag{3.22}
\]
3.5 Time Derivative Between Two Frames

The relationship of the time derivative between two frames is important for solving rigid body dynamics. It describes how unit vectors \( \hat{u} \), a descriptor for orientation or force, is rotated over time. This can be described by an example of unit vector, \( \hat{u} \), and the angular speed, \( \omega \). The time derivative of the \( \hat{u} \) can be described at the a function of \( \omega \) as shown in \( 3.23 \):

\[
\frac{d}{dt} \hat{u} = \omega \times \hat{u}
\]  \( (3.23) \)

This approach can be applied to a vector function which is described in terms of force such as in \( 3.24 \):

\[
f(t) = f_x(t)\hat{i} + f_y(t)\hat{j} + f_z(t)\hat{k}
\]  \( (3.24) \)

The derivative of the function in equation \( 3.24 \) returns:

\[
\frac{d}{dt} f = \frac{df_x}{dt} \hat{i} + \frac{df_y}{dt} \hat{j} + \frac{df_z}{dt} \hat{k} + \omega \times (f_x\hat{i} + f_y\hat{j} + f_z\hat{k})
\]  \( (3.25) \)

From the above equation, the derivative of the vector \( \mu \) can be rewritten as the second half of the equation below:

\[
\frac{d}{dt} f = \left( \frac{df}{dt} \right)_{\text{body}} + \omega \times f(t)
\]  \( (3.27) \)

Simplifying the above equation, the basic kinematic equation can be realized. Here the derivative of the force with respect to time is understood in the body frame known as the Transport Theorem.

This concept is used to derive the orientation. It can be applied to the rigid body dynamic that describes the change force with respect to the inertial frame of the quadrotor:
\[
\frac{d}{dt} \begin{bmatrix} 0 \\ F^i \end{bmatrix} = \frac{d}{dt} (Q^i \otimes \begin{bmatrix} 0 \\ F^i \end{bmatrix} \otimes Q) + \begin{bmatrix} 0 \\ \omega_t^i \times F^i \end{bmatrix}
\] (3.28)

As discussed in [5], the inertial frame force when rotated to the body can be realized as the normalized force in the body frame, which is constant (and therefore the derivative is 0). The equation simplifies to 3.29

\[
F^b_i = F^i \times \omega_{dy}^b
\] (3.29)

The rearrangement of 3.29 allows the description of the angular velocity in 3.30.

\[
\omega_{dy}^b = F^i \times \dot{F}^i
\] (3.30)

### 3.6 Blade Element Theory

Quadrotors are similar to helicopters such that they both use propellers to increase air velocity in an assumed disk. Blade element theory provides a relationship between coefficient of thrust, pitch angle, and inflow ratio while momentum theory offers the relationship between the coefficient of thrust and inflow ratio. [27] Momentum theory based on fluid mechanics theory, law of conservation of momentum, equal and opposite reactions, and incompressible flow dictate that air accelerated through a column will have an equal and opposite reaction force called thrust. Modeling with BET is highly nonlinear, therefore, [14] derives a simplified approach to modeling variable pitch blades. Blade Element Theory describes the coefficient of thrust as

\[Thrust = C_T \times K\]

where K is described in 3.32. This can be more formally presented in 3.31.

\[
C_T = \frac{T_i}{(\rho \pi R^2 V_{tip}^2)}
\] (3.31)

The term K, represents force which when multiplied by the coefficient of lift represents the actual force in the model. This equation is a function of the rotor radius R, and the velocity of the blade tip, \(V_{tip} = \omega R\).

\[
K = \rho \pi R^2 V_{tip}^2
\] (3.32)
The collective pitch of the blade can be determined solving for equation 3.33 which is a function of the rotor solidity, \( \sigma \), and the coefficient lift vs alpha \( C_{la} \).

\[
\theta_0 = \frac{6C_T}{\sigma C_{la}} + \frac{3}{2} \sqrt{\frac{C_T}{2}} \tag{3.33}
\]

The solidity of the rotor, \( \sigma \), can be determined by equation 3.34, a function of the number of blades \( N_b \), the chord length \( c \), and the radius of the rotor \( R \).

\[
\sigma = \frac{N_b c}{\pi R} \tag{3.34}
\]

The yaw moment can be solved as the function of the torque \( Q_{\text{blade}} \). This torque can be described as a function of the coefficient of torque, \( C_Q \) in equation 3.35.

\[
Q_{\text{blade}} = KRC_Q \tag{3.35}
\]

The coefficient of torque, \( C_Q \) can be related to \( C_T \) in equation 3.36 where the term \( \sigma \frac{c}{8} \) is assumed to remain constant for the simplification of the motor model.

\[
C_Q = \frac{C_T^2}{\sqrt{2}} + \sigma \frac{c}{8} \tag{3.36}
\]

Equation 3.36 describes the collective pitch with relationship to the thrust coefficient and blade solidity. The resulting thrust and moments as a function of the coefficient of thrust can described in matrix format in equation 3.37.

\[
\begin{bmatrix}
T \\
l \\
m \\
n
\end{bmatrix} =
\begin{bmatrix}
K & K & K & K \\
KL & -KL & -KL & KL \\
KL & KL & -KL & -KL \\
KR \sqrt{\frac{C_T}{2}} & -KR \sqrt{\frac{C_T}{2}} & KR \sqrt{\frac{C_T}{2}} & -KR \sqrt{\frac{C_T}{2}}
\end{bmatrix}
\begin{bmatrix}
C_{T1} \\
C_{T2} \\
C_{T3} \\
C_{T4}
\end{bmatrix} \tag{3.37}
\]

For simplification purposes, the mixing matrix will now be referenced as shown in 3.38.
\[
T_h = \begin{bmatrix}
K & K & K & K \\
K L & -K L & -K L & K L \\
K L & K L & -K L & -K L \\
KR \sqrt{\frac{C_{T1}}{I}} & -KR \sqrt{\frac{C_{T2}}{I}} & KR \sqrt{\frac{C_{T1}}{I}} & -KR \sqrt{\frac{C_{T4}}{I}}
\end{bmatrix}
\]

(3.38)

3.7 Geometric Attitude Tracking

Geometric attitude tracking uses geometric algebra methods to derive pose and angular velocity. [18] and [10] show how a geometric attitude tracking controller successfully handles complex, acrobatic maneuvers for quadrotors. Position controllers return desired acceleration as a function of position error, velocity error, as well as acceleration feed forward terms. The orientation of quadrotor is denoted as unit quaternion vector \( Q_i \) in the inertial frame. The goal is to find a desired rotation from the current attitude to the desired attitude. The desired force vector in the inertial frame described in 3.39. The terms \( \ddot{r}_{fb} \) and \( \ddot{r}_d \) delineate between the desired acceleration of the feedback loop and the feed-forward term desired acceleration. The desired force in the body frame can be calculated using the quaternion rotation of a vector in 3.40.

\[
F_i = m(\ddot{r}_{fb} + g^i)
\]

(3.39)

To ensure proper rotation between the current body frame and the desired inertial frame, the desired inertial force vector is normalized to produce a unit vector. Remember that the unit quaternion describes orientation as a rotation from the inertial frame to the body frame. Therefore, a rotation between the \( F_i \) and \( F_b \) would be described as an intermediate rotation in the body frame between the two vectors. The desired quaternion is solved using 3.12.

\[
\begin{bmatrix}
0 \\
F^i
\end{bmatrix} = (Q_d^{NoYaw} \otimes \begin{bmatrix}
0 \\
F_b
\end{bmatrix} \otimes Q_d^{NoYaw})
\]

(3.40)

However, the axis of rotation is arbitrary and not always optimal. A method to overcome this issue is presented by [21] which concludes using a minimal loss function that the minimal
Quaternion rotation can be written as a function of body frame $\vec{b} \in \mathbb{R}^{3x1}$ and reference frame $\vec{r} \in \mathbb{R}^{3x1}$. The symbol $\times$ denotes the cross product, and $\ast$ denotes the dot product.

\[
Q_{\text{min}}^{\text{NoYaw}} = \frac{1}{\sqrt{2(1 + \vec{b} \ast \vec{r})}} \begin{bmatrix}
1 + \vec{b} \ast \vec{r} \\
\vec{b} \times \vec{r}
\end{bmatrix}
\]  

(3.41)

\[
Q_{180}^{\text{NoYaw}} = \frac{1}{\sqrt{2(1 + \vec{b} \ast \vec{r})}} \begin{bmatrix}
1 + \vec{b} \ast \vec{r} \\
0
\end{bmatrix}
\]  

(3.42)

With the minimal rotation between the Desired force in the inertial frame and current body force in the inertial frame, the desired orientation of the quadrotor can be added to the desired quaternion orientation with equation 3.12 which is a function of the desired yaw angle $\psi_d$.

\[
Q_d = Q_{\text{min}}^{\text{NoYaw}} \otimes \begin{bmatrix}
\cos(\frac{\psi_d}{2}) & 0 & 0 & \sin(\frac{\psi_d}{2})
\end{bmatrix}
\]  

(3.43)

### 3.8 Quaternion Control Law

\[
\vec{Q}^b = Q_d^i \otimes Q_i^i
\]  

(3.44)

Quaternion error, $\vec{Q}$ is the product of the sequential rotation from the current pose in the inertial frame to the desired pose in the inertial frame with the resulting error in the body frame. A simplified controller relating quaternion error and angular velocity can be implemented but as noted in [30] and further investigated in [22] is prone to the unwinding phenomenon. This issue of a quaternion based attitude controller seeks the larger angle of rotation about the axis of rotation. To overcome this issue, [23] developed the control law, equation 3.45, to consider the scalar element sign to ensure proper rotation. This simple proportional $K_p$, derivative $K_d$ control law as a function of the quaternion scalar error $\vec{\eta}$ and the quaternion vector error $\vec{\epsilon}$.

\[
u = -\text{sgn}(\vec{\eta})K_p \vec{\epsilon} - K_d \vec{\omega}
\]  

(3.45)

Emulating the process in [14], the derivation of $\vec{\omega}_d$ requires solving kinematics equation.
for $\dot{\omega}$ and taking the time derivative. The desired body accelerations are calculated following [8] using the previous quaternion acceleration.

\[
\dot{Q} = \frac{1}{2} \Phi(Q) \omega
\]  

(3.46)

\[
\begin{bmatrix}
0 \\
\dot{\omega}_d \\
\dot{p}_d \\
\dot{q}_d \\
\dot{r}_d
\end{bmatrix}
= 2 \dot{Q} \otimes Q + 2 \left\| \dot{Q} \right\|^2
\]  

(3.47)

Using kinematic equation [3.47], a control law is derived such angular acceleration desired is a function of the quaternion rate second derivative which in itself is a PID function shown in equation [3.48]

\[
\ddot{Q} = \ddot{Q}_d + K_d * Q_{error} + K_p * Q_{error} + K_i * \int Q_{error} dt
\]  

(3.48)

Following the assertion that quaternion error is already in body frame, [3.48] and [3.47] are combined to form equation eq:17.

\[
\dot{\omega}_d = -\text{sgn}(\tilde{\eta}) * Kp * (\tilde{\epsilon}) - Kd * (\dot{\tilde{\epsilon}}) - Ki * \int \tilde{\epsilon}
\]  

(3.49)

Equation [3.49] is similar to the equations presented in [28] with the exception of quaternion error derivative.

### 3.8.1 Analytical Desired Angular Jerk

The analytical method which was pursued but later discarded due to computational complexity is presented here. The following equations are derived from work presented in [28] with an equation derivation from [13].

\[
\dddot{Q}^b = Q^i_d \otimes Q^i
\]  

(3.50)
\[ \ddot{\omega} = \omega - \omega_d \]  
(3.51)

\[ \ddot{\tilde{Q}} = \frac{1}{2} \Phi(\tilde{Q}) \ddot{\omega} \]  
(3.52)

The control law presented in [28] is as follows. It was sought to be used as the desired angular acceleration control law initially but was later discarded for computational complexity in the analytical derivative.

\[ \omega_d = -K_3 h \varepsilon - W^T \tilde{v} + R(\tilde{Q}) \omega \]  
(3.53)

The time derivative of equation 3.53, desired angular acceleration can be represented in 3.54.

\[ \dot{\omega}_d = -K_3 h \dot{\varepsilon} - \dot{W}^T \tilde{v} - W^T \dot{\tilde{v}} + \frac{dR(\tilde{Q})}{dt} \omega + R(\tilde{Q}) \dot{\omega} \]  
(3.54)

The derivative of the rotation matrix shown in 3.9 is expanded in 3.55.

\[ \frac{dR(\tilde{Q})}{dt} = -2 \eta S(\varepsilon) - 2 \eta S(\dot{\varepsilon}) + 2(S(\varepsilon) \times S(\dot{\varepsilon}) + S(\dot{\varepsilon}) \times (S(\varepsilon))) \]  
(3.55)

The rotation of the linear velocity described in the initial control equation 3.54 is expanded to 3.56 and its time derivative shown below it in 3.57.

\[ W^T(Q', \tilde{Q}, F') = 2 \ast \frac{F_i}{m} (\tilde{\eta} I - S(\tilde{\varepsilon}))S(R(Q')e_3) \]  
(3.56)

\[ \dot{W}^T = 2 \ast \frac{F_i}{m} (\tilde{\eta} I - S(\tilde{\varepsilon}))S(R(Q')e_3) + 2 \ast \frac{F_i}{m} (\tilde{\eta} I - S(\tilde{\varepsilon}))S(R(Q') \frac{dR(\tilde{Q})}{dt} e_3) \]  
(3.57)

As shown above, the compilation of 3.57 and 3.56 in 3.54 is algebraically intensive even after attempted simplification processes attempted in MATLAB.
Chapter 4

Theory of Developing Controller

4.1 Analytical Dynamics

The system is described as dynamical because the effects of the inputs are delayed time dependent due to the actuation constraints and how inputs are correlated to desired states. In the most simple case, a desired velocity would require an acceleration but the acceleration is limited by the system actuators and therefore the system acceleration is limited, therefore the desired velocity if large is unable to be achieved instantly.

Taking the time derivative of Euler’s second law of angular momentum is equivalent to torque. This can be solved for the change in angular velocity as a function of torque as in equation 4.25.

\[
\dot{\omega} = \begin{bmatrix} \dot{\rho} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} q r (J_z - J_y) \\ -p r (J_z - J_x) \\ p q (J_z - J_x) \end{bmatrix} \frac{1}{J_x} + \begin{bmatrix} \frac{\mathbf{L}}{J_x} \\ \frac{\mathbf{M}}{J_y} \\ \frac{\mathbf{N}}{J_z} \end{bmatrix} \tag{4.1}
\]

The dynamics used to calculate the change in coefficient of thrust is derived from the derivative of equation 4.25 which has been solved for thrust and torque as demonstrated in 4.3.

\[
\dot{T} = K_p (T_d - T) \tag{4.2}
\]
As noted in [14] and shown above in 3.37, dynamic inversion of the simplified motor model would result in possible irrational solutions, therefore the derivative of 3.37 is performed to calculate change in thrust coefficient which is used as a virtual input in the model shown in equation 4.27:

\[
\begin{bmatrix}
\dot{i} \\
\dot{m} \\
\dot{n}
\end{bmatrix} =
\begin{bmatrix}
J_x \ddot{p} + (J_y - J_z) (q \dot{r} - \dot{q} r) \\
J_y \ddot{q} + (J_x - J_z) (p \dot{r} - \dot{p} r) \\
J_z \ddot{r} - (J_x - J_y) (p \dot{q} - \dot{p} q)
\end{bmatrix}
\] (4.3)

\[
\begin{bmatrix}
\dot{C}_T^1 \\
\dot{C}_T^2 \\
\dot{C}_T^3 \\
\dot{C}_T^4
\end{bmatrix} = Th^{-1}
\begin{bmatrix}
I \\
\dot{m} \\
\dot{n}
\end{bmatrix}
\] (4.4)

### 4.2 Model Parameters

The controller was constructed and tuned using attributes of the Reaper 500 VPQ. The resulting moments of inertia were constructed using known weights of individual components and their distance from the assumed center of gravity.

<table>
<thead>
<tr>
<th>Total Mass</th>
<th>1.240</th>
<th>kg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Length</td>
<td>0.635</td>
<td>m</td>
</tr>
<tr>
<td>Total Width</td>
<td>0.365</td>
<td>m</td>
</tr>
<tr>
<td>Rotor Radius</td>
<td>0.118</td>
<td>m</td>
</tr>
<tr>
<td>Mass Less Battery</td>
<td>0.986</td>
<td>kg</td>
</tr>
<tr>
<td>Battery Mass</td>
<td>0.2540</td>
<td>kg</td>
</tr>
<tr>
<td>Jx</td>
<td>0.0262</td>
<td></td>
</tr>
<tr>
<td>Jy</td>
<td>0.0680</td>
<td></td>
</tr>
<tr>
<td>Jz</td>
<td>0.0917</td>
<td></td>
</tr>
</tbody>
</table>
The properties of the rotors were related to the blades that come standard with the Reaper 500. The rotational speed of the blades was measured at nominal operation to be about 7000 RPM. The blade design best matched the NACA 0010 airfoil design and was modeled using a Reynolds number of 100,000.

Table 4.2: Blade Properties

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Air Density</td>
<td>1.225</td>
<td>kg/m3</td>
</tr>
<tr>
<td>Max Thickness</td>
<td>2.95</td>
<td>mm</td>
</tr>
<tr>
<td>Chord Length</td>
<td>28.21</td>
<td>mm</td>
</tr>
<tr>
<td>Number of Blades</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>Reynolds Number</td>
<td>80000</td>
<td></td>
</tr>
<tr>
<td>Angular Speed</td>
<td>7000</td>
<td>RPM</td>
</tr>
<tr>
<td>NACA</td>
<td>0010</td>
<td></td>
</tr>
</tbody>
</table>

4.3 Simulation Model

In [5], the control method seeks to reduce the necessity for trigonometric functions such as sine, cosine, tangent contain singularities that must be managed and therefore cumbersome to the control process. To overcome these singularities, a quaternion based controller is implemented. The motor model derived from BET is implemented in similar fashion to [14] to solve for thrust and torque of the blade as a function of pitch. The derivative of the motor model is used to solve for the irrational solutions for yaw. The inputs to this new motor model are angular jerk. To derive angular jerk a new control law is presented, similar [5] but with added changes influenced by [28] proposed command filter. The overall control system architecture is shown in [4-1]
Figure 4-1: Simulation Control Structure

The feedback force is formed from the positional PID loop which outputs feedback acceleration which is inherently a function of desired acceleration, velocity error, and positional error.

\[
\ddot{r}_{fb} = \ddot{r}_d + K_d \times \dot{r}_{error} + K_p \times r_{error} + K_i \times \int r_{error} dt \tag{4.5}
\]

From (4.5) a resulting vector is produced which in Euler based methods would covert to Euler angles. Instead, this vector termed as force in the inertial frame commands the trajectory which the quadrotor will follow.

\[
F_i = m(\ddot{r}_{fb} + g^i) \tag{4.6}
\]

As noted in [5], the Newton-Euler forces can be represented with quaternion operation shown in (4.7)

\[
\begin{bmatrix}
0 \\
\vec{r}
\end{bmatrix} = \begin{bmatrix}
0 \\
-mQ \otimes 0 \\
-F^b \otimes Q + 0 \\
g^i
\end{bmatrix} \tag{4.7}
\]

The desired quaternion is described in [21], we can find the the minimal rotation between to vectors at some arbitrary angle in (4.11) where the vectors are: the normalized inertial force vector (4.8) and the normalized body force vector (4.9) Finding this rotation vector is akin to finding the desired rotation vector or in terms of quaternions, the desired quaternion rotation.

Force bar inertial is the normalized force inertial as shown in (4.8) and the normal of the force in the body frame in (4.9).
\[
F^i = \frac{F^i}{\|F^i\|} \quad \text{(4.8)}
\]

\[
F^b = \frac{F^b}{\|F^b\|} = \begin{bmatrix} 0 & 0 & \pm 1 \end{bmatrix}^T \quad \text{(4.9)}
\]

\[
F_b = \begin{bmatrix} 0 & 0 & F_{\text{total}} \end{bmatrix}^T \quad \text{(4.10)}
\]

\[
Q_{d,NoYaw}^{NoYaw} = \frac{1}{\sqrt{2(1 + F^T F^b)}} \begin{bmatrix} 1 + F^T F^b \\ F^i \times F^b \end{bmatrix} \quad \text{(4.11)}
\]

\[
Q_{d,NoYaw}^{NoYaw} = Q_d \otimes \begin{bmatrix} \cos\left(\frac{\phi_d}{2}\right) & 0 & 0 \\ 0 & \sin\left(\frac{\phi_d}{2}\right) \end{bmatrix} \quad \text{(4.12)}
\]

To find the desired body angular rates, we take the cross product of the normalized force inertial and the derivative of the normalized force inertial as shown in 4.14.

\[
\frac{d}{dt} \begin{bmatrix} 0 \\ F^i \end{bmatrix} = \frac{d}{dt} \left(Q^* \otimes \begin{bmatrix} 0 \\ F^i \end{bmatrix} \otimes Q\right) - \begin{bmatrix} 0 \\ \omega^b_d \times F^i \end{bmatrix} \quad \text{(4.13)}
\]

\[
\omega^b_{d,xy} = F^i \times \dot{F}^i \quad \text{(4.14)}
\]

The derivative the normalized force inertial is derived in 4.15.

\[
\dot{F}^i = \frac{\dot{F}^i}{\|F^i\|} - \frac{F^i (F^T \dot{F}^i)}{\|F^i\|^3} \quad \text{(4.15)}
\]

The derivative of force inertial can be described through the numerical derivative of jerk as shown in 4.16.

\[
\dot{F}^i = m * (\ddot{r}_{fb}) \quad \text{(4.16)}
\]

This formulation describes pose without desired yaw, desired body yaw is described in 4.17.
\[ \omega_{dc}^b = \dot{\psi}_d \] (4.17)

As noted in 3.44, quaternion error is described as the quaternion rotation between the desired orientation and the actual orientation. This rotation would in effect describe the rotation in the body frame.

\[ \tilde{Q}^b = Q_i^* \otimes Q_d^i \] (4.18)

The angular velocity error can be calculated, as the desired angular velocity is derived in equation 4.17. This is required for calculation of the derivative of quaternion error which uses the angular velocity error in equation 4.21.

\[ \tilde{\omega} = \omega_d - \omega \] (4.19)

The time derivative of the quaternion is a skew systematic multiplied by the angular velocity.

\[ \dot{Q}' = \frac{1}{2} \Phi(Q')\omega^b \] (4.20)

\[ \dot{\tilde{Q}} = \frac{1}{2} \Phi(\tilde{Q})\tilde{\omega}^b \] (4.21)

Equations 4.21 and 4.20 are derived in [28] but are used differently for the purposes of generating a new control law show in equation 4.22.

\[ \omega_d = -sgn(\tilde{\eta}) * Kp * (\tilde{e}) - Kd * (\dot{\tilde{e}}) - Ki * \int \tilde{e} dt \] (4.22)

Following the process presented in [14], the angular jerk is calculated as they are used as inputs in 4.26.

\[
\begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{bmatrix} =
\begin{bmatrix}
\dot{p}_d \\
\dot{q}_d \\
\dot{r}_d
\end{bmatrix} +
Kd
\begin{bmatrix}
\dot{q}_d - \dot{q} \\
\dot{r}_d - \dot{r}
\end{bmatrix} +
Kp
\begin{bmatrix}
p_d - p \\
q_d - q
\end{bmatrix} +
Ki
\int
\begin{bmatrix}
p_d - p \\
q_d - q
\end{bmatrix} dt
\] (4.23)
In the attitude controller, the current resulting torques are solved as a function of the current thrust coefficients as demonstrated in 4.24.

\[
\begin{bmatrix}
T \\
l \\
m \\
n
\end{bmatrix} = Th
\begin{bmatrix}
C_{T1} \\
C_{T2} \\
C_{T3} \\
C_{T4}
\end{bmatrix}
\] (4.24)

The actual angular acceleration is then calculated from the current torques acting the rigid body is shown equation 4.25.

\[
\begin{bmatrix}
\dot{p} \\
\dot{q} \\
\dot{r}
\end{bmatrix} = \begin{bmatrix}
\frac{q r (J_x - J_y)}{J_x} \\
-p r (J_x - J_y) \\
p q (J_x - J_y)
\end{bmatrix} + \begin{bmatrix}
l \\
m \\
n
\end{bmatrix}
\] (4.25)

The dynamics used to calculate the change in coefficient of thrust is derived from the derivative of equation 4.25 which has been solved for thrust and torque as demonstrated in 4.26.

\[
\begin{bmatrix}
\dot{T} \\
l \\
m \\
n
\end{bmatrix} = K_p (T_d - T) \\
J_x \dot{p} + (J_x - J_y) (q \dot{r} - \dot{q} r) \\
J_y \dot{q} + (J_x - J_z) (p \dot{r} - \dot{p} r) \\
J_z \dot{r} - (J_x - J_y) (p \dot{q} - \dot{p} q)
\] (4.26)

Using the kinematics described in equation 3.37 we can find the change in coefficient of thrust through the inverse and derivative shown in equation 4.27.

\[
\begin{bmatrix}
C'_{T1} \\
C'_{T2} \\
C'_{T3} \\
C'_{T4}
\end{bmatrix} = Th^{-1} \begin{bmatrix}
\dot{T} \\
l \\
m \\
n
\end{bmatrix}
\] (4.27)

Using the mathematical models described above and derived control laws, a simulated VPQ can be constructed using ordinary differential equation solvers in programs such as MATLAB.
4.4 Control Methods

The control method presented in this paper follows traditional back stepping methods. Starting with the PID attitude rate controller, a linearized form is generated at hover conditions and then used to produce transfer functions that allows pole placement and tuning of the attitude rate controller. Due to the motor model, the transfer functions are second order equations.

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \quad (4.28)$$

Transfer functions are generated using linearized equations at hover conditions. The PID gains were placed at the resulting poles for resulting attitude orientation controller. The desired orientation inputs are Euler representations converted to quaternions for use by the controller. A quaternion to Euler angle conversion process is used on the output data as a sanity check for the controller. Figure 4-2 shows the results from the tuning process of the attitude pose controller.

4.4.1 LQR Trajectory Control

Linear Quadratic Regulators are used for loop closure of multiple input, multiple output systems. They are useful when systems are greater than 2nd order and begin to lose intuitive controls. LQR controls exhibit the effect of many inputs such as position error, velocity error, acceleration error, or combination of these elements.
Figure 4-2: Attitude Tuning Results

To determine effective pole placement, a cost function $J$ is used where state cost $x$ and control cost $u$, state cost weight $Q$, and control cost weight $R$. $Q$ matrices combine different states while $R$ combines different inputs. $Q$ and $R$ matrices are designed such that the cost function $J$ will go to zero as $T$ goes to infinity. The $Q$ and $R$ matrices prioritize states and combinations of states of positions, velocities, or combinations of positions and velocities. $Q$ and $R$ must be insured to be positive definite to return appropriate values and settle at zero. Negative matrices or semi-positive definite matrices will cause $J$ to goto negative infinity.

$$J = \int_{0}^{\infty} (x^T Q x + u^T R u + 2x^T Nu) \, dt$$

(4.29)

The input $u$ vector is function of the gain matrix $k$ and the states as shown in [4.30]

$$u(t) = -Kx(t)$$

(4.30)

$$\dot{x} = (A - Bk)x$$

(4.31)

the feedback control law that minimizes the value of the cost is where $K$ is given by

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

(4.32)
and $\hat{P}$ is found by solving algebraic Riccati equation:

$$A^T P + PA - (PB + N)R^{-1}(B^T P + N^T) + Q = 0 \quad (4.33)$$

Starting with the linear state equation represented as in

$$\dot{x} = Ax + Bu \quad (4.34)$$

Since the LQR only is concerned as a position controller, the states can be reduced to:

$$x = \begin{bmatrix} r_n & r_e & r_d & \dot{r}_n & \dot{r}_e & \dot{r}_d & \psi \end{bmatrix}' \quad (4.35)$$

The state equations can be linearized to the following forms:

$$A = \begin{bmatrix} 0_{3x3} & I_{3x3} & 0_{3x1} \\ 0_{3x3} & 0_{3x3} & 0_{3x1} \\ 0_{1x3} & 0_{1x3} & 0_{1x1} \end{bmatrix}' \quad (4.36)$$

$$B = \begin{bmatrix} 0_{3x3} & 0_{3x3} \\ I_{3x3} & 0_{3x3} \\ 0_{1x3} & 0_{1x3} \end{bmatrix} \quad (4.37)$$

$$b = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}' \quad (4.38)$$

This equation can be solved for the $K$-gain matrix that is used in the Linear quadratic regulator position controller. This $K$-gain matrix is used to multiply the current states to generate the inputs.

$$u_r = \begin{bmatrix} u_p \\ \dot{u}_p \\ \dot{u}_\psi \end{bmatrix} \quad (4.39)$$
\[
\begin{bmatrix}
\ddot{\phi}_n \\
\ddot{\phi}_c \\
\ddot{\phi}_d
\end{bmatrix} = u'_p + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{4.40}
\]

\[
u_\phi = \dot{\psi}' \tag{4.41}
\]

\[
\ddot{u} = -K \ddot{x} \tag{4.42}
\]

\[
F_i = m * ((u_r + \ddot{u})_{1:3} + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}) \tag{4.43}
\]

PID controllers are found by tuning from pole place to transfer functions generated from the linearization process of the attitude controller at hover conditions.
Chapter 5

Simulation Results

5.1 Attitude Controller Simulation

The attitude controller was simulated in a decoupled environment to test for stability of the chosen gains. As quaternions are not intuitive descriptors of orientation, the recorded quaternion states of the system are converted in the Euler Angles after the simulation allow a more capable understanding of the reaction of the system. Small angle perturbations were first simulated at a nominal range of 20 degrees for roll, pitch, and yaw. All the results are similar to the system reaction in 5-1.
The controller was also tested for large angle recovery. As the control law was not proven asymptotically globally stable using Lyapunov Functions, large angle tracking scenarios were simulated to study large angle response. While the results in Figure 5-2 present slight overshoot, the stability of the controller is noted as the angular velocity states settle at zero as time continues from 3 seconds.
Figure 5-2: Attitude Tuning Results

The small angle perturbations and large angle recovery experimentally proved correct operation of the attitude controller as well as stability in design.

5.2 Aggressive Trajectory Maneuver with LQR Position Control

After the successful tuning of the attitude controller, the LQR position controller was coupled to the system and made the system closed loop. Using time parametric trajectory generation, the controller model was tested through a range of aggressive trajectories. The results of two separate aggressive trajectory maneuvers, the circle and Figure 8 are studied. Bryson’s
Rule was used as a starting point in the tuning of the LQR position control, only minor changes to the Q and R matrices for then on. The first trajectory follows a circle in the 3 dimensional field. \[5-3\] is a circle trajectory that varies on all three axis.

![Aggressive Trajectory following with LQR and Differential Flatness](image)

**Figure 5-3: Circle Trajectory**

The control effort is a concern for any controller, in \[5-3\] the control effort in monitored in the change of Coefficient of Thrust shown in \[5-4\] stay in the manageable range of available thrust of the blades in the given environment that is modeled at STP where the air density is assumed to be \(1.225 \text{ kg/m}^3\). Better LQR tuning may have resulted in a more stable roll command as can be seen as gradually smoothing out over time in the Angular Velocity graph of \[5-4\].
The figure eight trajectory shown in Figure 5-5 examines the further tracking ability of the controller in simulation. Again with minimal tuning using Bryson’s Rule, the resulting trajectory confirms the established qualities of the variable pitch quadcopter in simulation. The time allotted to the completion of one completed circuit was 8 seconds. The aggressiveness of the trajectories could be increased by decreasing the time allotment.
Aggressive Trajectory following with LQR and Differential Flatness

Figure 5-5: Figure 8 Trajectory

Quadrotor States: Figure 8 Trajectory

Figure 5-6: Figure 8 States
Chapter 6

Experimentation

The commercially available Reaper 500 VPQ is used as a test bed for the constructed autopilot. This platform comes with a RC based control with no autopilot features. It makes a useful platform as it supports the plug and play ability of a Pixhawk autopilot. This platform can be easily modeled in simulation as all the parts are standard. The platform can be easy to replicate. The firmware structure and ground station software used is available in the open-source community. All of the software packages are interloped using an open source communication protocol called Mavlink.

![Reaper 500 Platform](image)

Figure 6-1: Reaper 500 Platform

The robotic operating system provides an infrastructure useful to developing controllers for
small scale robotics. The support packages that integrate into the infrastructure make it a robust development platform. Each carry integration packages to available existing flight controllers available. The Pixhawk flight controller is a robust hardware platform that carries all sensors required for achieving inertia guided autonomous flight. The Pixhawk platform also supports a range of open source flight controller firmwares. This is useful to developing a controller using previously developed controller substructures such as low end controllers for managing servos, motors, and other peripherals. The PX4 firmware (not to be confused with Ardupilot) supports a broad base of development objectives and was chosen due to its wide support community and development environment. PX4 has previously been integrated into a simulation environment called Gazebo. This simulation environment can be used to test position controller scripts that either send commands via MAVROS to the Pixhawk PX4 firmware or directly communicate with PX4 firmware. MAVROS is a ROS package that supports the communication between the PX4 firmware and ROS environment. This allows for quick paced development of flight controllers as most of all the Pixhawk sensor data can be subscribed to and much of the actuator topics, high level, and low level commands can be published to.

The PX4 firmware already implements a quaternion based attitude controller, way point trajectory controller, and attitude rate controller. The firmware also supports a range of control commands such as takeoff, land, and follow a mission. Using this controller architecture, a linear quadratic regulator position controller was written in Python that follows a deferentially flat trajectory. The outputs of the LQR are commanded quaternion pose, desired angular velocity, desired angular acceleration, desired velocity, and desired acceleration. All of these feed command topics to the PX4 firmware attitude and throttle controllers. The implementation of the quaternion-based LQR system is crucial to the development of the VPQ and does not use Euler angles or include any trigonometric based functions so as to limit the occurrence of singularities.

The PX4 attitude controller is tuned experimentally in a flight test cage using a telemetry based communication system between a ground station and the Pixhawk. The ground control station runs QGroundcontrol which can be used to tune the PID attitude controller on the PX4 firmware. The throttle is hard coded to 80% of the full throttle of the motor. When the vehicle
is armed, the throttle will spin up to 80% and does not change until the vehicle is disarmed. During the tuning process all the Integral and Derivative gains are reduced to zero. Proportional gain tuning is performed in small 10% increments until takeoff and hover in ground effect is achieved. The attitude is slowly tuned to a stabilized control in this ground effect hover, slowly tuning the roll and pitch proportional gains. After a mostly stabilized control is achieved, the quadrotor is flown outside of the ground effect hover to a hover about a meter and a half from the ground. At this height the the integral and derivative gains are slowly tuned to garner the appropriate response of the stabilization of the attitude controller. The PX4 records the flight logs during each of its flights and this data can be used to analyze the response of the flight controller. This data will show the inputs of the raw IMU (gyro and accelerometer) data, the filtered IMU data, the controller response, and actuator outputs. All of this data can be used to validate the stability of the tuned controller.
Chapter 7

Conclusion and Future work

7.1 Conclusions

VPQs are multirotor platform that remove the constraints of rotor size that is the limiting factor for traditional IC based VPQs. The characteristics of the VPQ enable near instantaneous thrust as well as the ability for instantaneous inverted thrust. To build an autonomous control for this platform, a novel motor model must be used to describe the change in thrust generated from the change in the rotors blade angle of attack. A method developed from Blade Element Theory and Blade Element Momentum are used along with their derivative functions as a virtual control allocation to solve for irrational yaw components.

To investigate the capabilities of the variable pitch quadcopter, an aggressive attitude tracking control structure was implemented using quaternions for their lack of singularities as well as studied global asymptotic stability for large rotations. A quaternion control equation is implemented using features studied in past quaternion error rotation controllers to develop an equation for angular acceleration which is more stable to integrate in the attitude rate controller. The controller is linearized at hover for pole placement. A cascade attitude rate, attitude orientation, and position controller are used to enable feedforward terms for further aggressive trajectory tracking.

A Linear Quadratic Regulator position controller is used to generate aggressive trajectories for a geometric attitude tracking scheme. This VPQ controller was tested under a circle, figure 8, and helical trajectory. The controller was also tested for a full 360 degree flip and showed no signs of instability.
7.2 Future Works

The investigation into VPQ controls shows promising control structure for aggressive flight maneuvers that can not be achieved by traditional variable RPM quadrotors. A model was constructed to enable aggressive flight trajectories with cascaded control design which enables future feedforward terms from the trajectory generation. The simulation model proves that second order quaternion error dynamics prove useful for deriving a control structure that uses angular acceleration as an input.

In accordance with the objectives of achieving aggressive and inverted flight maneuvers, a singularity free attitude controller must be proven asymptotically stable in the global SO(3) realm. This would be accomplished by proving the Lyapunov Stability analysis of the second order quaternion control law. Much previous work has shown proof of global asymptotic stability with the use of managing the sign of the quaternion scalar. These prove stability of the above presented control law.

Further experimental work should be performed to show the differential flatness trajectories being processed by the Linear Quadratic Regulator. Test flights would prove the effective integration of the LQR controller as shown in the implementation shown in this paper. Experimentation could also verify the tuning requirements used in this paper’s implementation.

The last objective sought by this paper was to verify experimentally the ability of the VPQ to fly way points. This experiment would prove the effectiveness of the contrived control method and the usefulness of the quadrotor simulation model. The model could be further used in an adaptive control structure as well.
References


[10] T. Fernando, J. Chandiramani, T. Lee, and H. Gutierrez. Robust adaptive geometric tracking controls on so (3) with an application to the attitude dynamics of a quadrotor uav. In


