5-1-2011

Perching Using a Quadrotor with Onboard Sensing

Jeremy C. Goldin
Utah State University

PERCHING USING A QUADROTOR WITH ONBOARD SENSING

by

Jeremy C. Goldin

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Electrical Engineering

Approved:

Dr. Wei Ren Major Professor	Dr. David Geller Committee Member
Dr. Jacob Gunther Committee Member	Dr. Byron R. Burnham Dean of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2011

Abstract

Perching Using a Quadrotor with Onboard Sensing

by

Jeremy C. Goldin, Master of Science

Utah State University, 2011

Major Professor: Dr. Wei Ren
Department: Electrical and Computer Engineering

This thesis presents an implementation of autonomous indoor perching using only onboard sensors on a low-cost, custom-built quadrotor. The perching aggressive maneuver is representative of a class of control problems for aerobatics that requires an agile and robust control system for maneuvering accurately at high speeds. Such research extends the typical functionality of micro air vehicles (MAV) from low speed and stationary observation to dynamic aerobatic transitions for broader operational capabilities including confined landings and evasive maneuvering. To achieve this, three major challenges are overcome: precise and real-time positioning, sensing of the perch and path to the perch, and control methods for robust and accurate tracking at high speeds. Navigation in unstructured, global positioning system (GPS)-denied environments is achieved using a visual Simultaneous Localization and Mapping (SLAM) algorithm that relies on an onboard monocular camera. A secondary camera, capable of detecting infrared light sources, is used to locate the pathway for the maneuver and the perch, simulating sensing of the actual perch, for perching without prior knowledge of the location of the perch. The full physical system architecture is covered in detail, indicating the components and integration necessary to obtain effective aggressive control of an inexpensive quadrotor. The difficulties of attitude stabilization on noisy and lower-quality sensors are successfully addressed so that the air vehicle can be treated as a

simple second-order system for the purposes of navigation and response to dynamic maneuvering commands. The system utilizes nested controllers for attitude stabilization, vision-based navigation, and perching guidance, with the navigation controller implemented using novel nonlinear saturation control within a Proportional-Integral-Derivative (PID) structure. The quadrotor is therefore able to autonomously sense the perch, reach initial high speeds for obtaining rapid deceleration from aerodynamic effects, dynamically transition to a high angle of attack post-stall configuration, and make a low-speed accurate landing on an inclined surface, using only onboard sensors.

(216 pages)

To Nayana

Acknowledgments

This thesis represents a lot of time, effort, and knowledge, much of which would not have been possible without the important people that, together, made this research possible and successful.

I first want to acknowledge my appreciation for the scholarship I was awarded and support from my work that enabled me to pursue a thesis-oriented degree. Recognition goes to the Science, Mathematics & Research for Transformation (SMART) scholarship, 2009 cohort, www.asee.org/fellowships/smart. I sincerely thank my supervisors at the Landing Gear Office, Hill Air Force Base, specifically Doug Ball and Mike Schow, for working with me in my transitioning situation, Ron Montgomery for choosing me as a SMART scholar recipient and making my tenure a success and an enjoyment, and Dan Christensen for his efforts in approving my two-year master's degree and receiving extensive financial support.

I want to recognize my advisor, Dr. Wei Ren, for providing and supporting this research project and allowing me the freedom to learn and choose my own path in accomplishing my ever-changing goals. Thanks go to Dr. Yongcan Cao and Dr. Don Cripps for their expert controls advice and assistance in many of the idiosyncrasies involved in a custom-built flying vehicle. As part of the latter, I must give extensive gratitude to Heidi Harper for her constant assistance in obtaining just the right part that we needed at any given moment. And I want to thank Dave Hylands at the Gumstix OldNabble mailing list for his expert advice and timely support for the numerous troubles we had with the Gumstix, Robostix, and I^2C protocol.

On a more personal note, I want to appreciate Richard Dunkley for our shared venture into academia from Hill AFB as SMART scholar recipients, and for all that I learned from him on the custom ground robot project we did together, which was a very nice stepping stone to the quadrotor platform. I also want to thank Mark Anderson for his excitement about the project, always wanting to hear about the latest results (or problems). Similar thanks also go to my old college buddy, Scot Myhr.

Of course no acknowledgments from me could be complete without mentioning the one person closest to me, my fiancée, Nayana Wagle. Her ever-present emotional support, boundless joy of life, and nonstop companionship were elemental in my getting through this research so unscathed. Her forgiveness of my quite frequent bouts of grumpiness are also much appreciated.

But most of all, I want to acknowledge my research partner, Vaibhav Ghadiok. His passion for research, for thinking big, his drive to work hard to achieve the best, his keen mind and can-do approach to problems, and his ever-ready desire to try something new have enriched my life, taught me new perspectives and made this project a crucial milestone in my career. But we are not done; together we will achieve new heights.

Jeremy C. Goldin

Contents

List of Tables

List of Figures

Acronyms

ADC	Analog Digital Converter
BL	Brushless
BLDC	Brushless DC
BPP	Bits Per Pixel
CCD	Charged Coupled Device
CG	Center of Gravity
COM	Computer On Module
DB	Decibel
DC	Direct Current
DCM	Direction Cosine Matrix
DOF	Degrees Of Freedom
ESC	Electronic Speed Controller
EKF	Extended Kalman Filter
EMI	Electro-Magnetic Interference
EMF	Electro-Magnetic Feedback
FAST	Features from Accelerated Segment Test
FIR	Finite Impulse Response
FOV	Field Of View
FPGA	Field Programmable Gate Array
FPU	Floating Point Unit
GPS	Global Positioning System
I^2C	Inter-Integrated Circuit
IC	Integrated Circuit
IIR	Infinite Impulse Response
IMU	Inertial Measurement Unit
IR	Infrared

Kbps	Kilobits Per Second
KBps	Kilobytes Per Second
LED	Light Emitting Diode
LQR	Linear Quadratic Regulator
LSB	Least Significant Bit
MAV	Micro Air Vehicle
NED	North-East-Down
PCB	Printed Circuit Board
PID	Proportional Integral Derivative
PPU	Price Per Unit
PTAM	Parallel Tracking And Mapping
PWM	Pulse Width Modulation
RC	Radio Control
RGB	Red Green Blue
RMS	Root Mean Squared
RPM	Revolutions Per Minute
SBC	Single Board Computer
SFM	Structure From Motion
SLAM	Simultaneous Localization And Mapping
SOAR	Silent Operating Aerial Reconnaissance
SONAR	SOund Navigation And Ranging
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicle
USB	Universal Serial Bus
VTOL	Vertical Take-Off and Landing

Chapter 1

Introduction

Flying vehicles have long held the imagination of many; turn them into flying robots and there is a guaranteed popular interest. After the invention of piloted aircraft, it was not long before the idea of completely unmanned aerial vehicles, UAVs, became a reality as well. Aircraft in general have changed the way people operate, and UAVs as a subset of these, incur their own impact on society.

1.1 Motivation

It is unlikely at this stage of technology that anyone would wonder about the application of flying vehicles, but what of UAVs? Given the basic difference between UAVs and regular aircraft being the lack of humans onboard, it becomes apparent that such vehicles can perform tasks that would be undesirable, dangerous, or plain impossible for a human pilot to execute. The elimination of the need for human onboard control also allows for obvious design changes, including shrinking in size, that would be impossible if a human crew were required to be on board. Lower operating costs are also an obvious result of such implementations, as a similarly capable vehicle can perform its mission while the human crew are able to perform other tasks, or at least reduce workload and strain.

1.2 Applications

General applications of UAVs can certainly include the same applications as currently piloted aircraft can perform, such as transportation of goods or passengers. The importance of distinguishing a difference is in outlining exactly why UAVs are desirable. Some of these differences are directly applicable for military applications.

1.2.1 Military Missions

High risk, precision specific or extensive mission lengths are a common requirement in military strategies. Such missions are ideally suited for UAVs, as this leaves the most important military assets, its people, free to continue to perform tasks best suited to their capabilities and long term health, and thus the long term success of the military.

The Growing Use of Military UAVs

During the past decade, UAVs have matured into fully controllable and reusable combat and observation aircraft. Their numbers have grown, in various sizes and forms, providing flexible sky intelligence information for ground troops. Pilot-less aircraft started as pre-programmed drones, evolved into remotely piloted vehicles or UAVs, and are now sometimes called unmanned aircraft systems (UASs), in appreciation of their role within a larger collection of complex mission assets.

In the recent decades, the primary mission of the United States (U.S.) military was defense against heavy weaponry and advanced technologies that were employed by a single large source, and so developed a formidable array of weapon systems and tactics. Lately, the U.S. military has found itself instead facing a growing collection of asymmetric threats that require different methods and means for identification and elimination. The military is shifting to consist of a collection of smaller, rapidly transportable units with the ultimate goal of deploying anywhere on the globe within a very short time [1]. This necessitates unprecedented levels of battlefield situational awareness, allowing the location and engagement of targets at distances exceeding enemy capabilities, such that enemy assets can be reduced enough to cripple their ability to fight effectively, before even closing within previous engagement distances.

Strong situational awareness capabilities dictate a highly sophisticated and robust tactical network, including multiple levels of functionality, from minimalist devices that might reside within the ground, to effective hubs that would enable command and control components to absorb and transmit information at large throughputs. All types of UAVs are critical asset contributions to this network and such UAVs are being identified within a

specific category known as silent operating aerial reconnaissance (SOAR).

Expansion of UAV Mission Capabilities

While initially set up to be a part of the SOAR category for use in reconnaissance or sustained surveillance, UAVs are becoming more capable; one of the more well-known US military UAV, the Predator, Figure 1.1, can be controlled remotely to detect enemies using an infrared camera and fire a laser guided missile [2]. At the smaller end of the spectrum are UAVs that a single soldier can carry around and launch by hand, such as the Raven, built by AeroVironment, Figure 1.2 [3].

Fueled by rapidly maturing technologies, the improved capabilities of all different kinds of UAVs, coupled with the recent desire by military planners to work out detailed tactics, techniques, and procedures for soldiers to use aerial vehicles in combat, a turning point for military UAVs has been reached. Robustness, longer operating life, and better sensing has enabled greater use of small UAVs, while computer and radio link advances allow operation of large UAVs to be shifted from one ground control station to another, so that UAV pilots located in combat areas can pass control to fellow soldiers stationed at home. Remote piloting capabilities have improved such that soldiers without piloting experience are successfully operating UAVs.

UAVs are even being equipped with automation functionalities, such as automated landings, reducing the number of incidents due to human error. Military goals are starting

Fig. 1.1: The Predator UAV, built by General Atomics.

Fig. 1.2: The hand-launched Raven UAV, built by AeroVironment.

to include such advanced possibilities as UAVs delivering cargo to soldiers on the battlefield or flying missions in coordinated swarms.

With the coming advances in low-cost, reliable, and safe UAVs, there is already a strong desire in the commercial sector for this type of technology for use in everything from postal shipping, pipeline surveys, forestry, ocean and weather observation, law enforcement, and more. Although the commercial sector may not have as much high-risk mission requirements like the military, it can benefit highly from UAV use for missions requiring high-precision, long-duration, or repetitive tasks, freeing up the human employees for more advanced duties.

1.2.2 Remote Operation vs Autonomous Capability

A key separation between UAV capabilities and applications is whether the vehicle has to be controlled remotely by a human operator, or whether it has some internal level of autonomy. Remotely operated vehicles have the obvious restriction that a remote-pilot, and often even an entire team, must always be directing the actions of the vehicle, limiting the ratio of vehicles to operators. UAVs with autonomous capabilities allow for more vehicles to be capable of being controlled by one pilot by using high-level mission guidance, greatly enhancing the amount of work that can be done by one person.

1.2.3 Fixed-Wing vs Rotary-Wing Aircraft

In addition to the types of actions a UAV can execute independently, there is the actual capabilities of different types of aircraft structures. One rough distinction is between fixed-wing aircraft - which as the name implies, have one set structure and fly via a separate propulsion system that propels the aircraft forward, using the airflow over its wings as it moves through the air to provide lift - while the contrasting rotary wing aircraft obtain lift through direct motion to the wing itself. Although this difference is often only important depending on the application, one wide-spread use of rotary wing type aircraft over the fixed-wing style is in vertical takeoff and landing (VTOL) capabilities. Rotary aircraft, such as helicopters, are the most common implementation of VTOL vehicles, with this capability enabling less restrictions in operational locations as long take-off and landing

runways are not required. In addition, they have hovering and low-speed flight capabilities, enabling certain functionalities for steady, consistent observation, as they can maintain a constant view without needing to do flyover passes that fixed-wing aircraft would require.

1.2.4 Aircraft Size

A more perceptive difference between aircraft styles is its size. Aircraft can be built of various sizes, from space shuttle large to even as small as an insect. Vehicle size obviously impacts the types of missions that can be performed; large aircraft can go long distances and carry large payloads, but are restricted in the types of spaces they can maneuver in. Smaller vehicles can be used to get very close to objects and even be used inside buildings, but are limited in the capacity of weight and power they can command. A subset of small vehicles, known as micro air vehicles (MAVs), consist of air vehicles that are roughly 50 cm or less in wingspan, and are capable of maneuvering in very tight spaces. Such vehicles are typically used for observation missions, such as: visual reconnaissance, situational awareness, damage assessment, surveillance, relaying communication messages, and sensing- or communication-based missions involving sensing of biological or chemical agents.

Helicopters have been compared against other aircraft in terms of miniaturization and capabilities for the missions described above [4]. Aircraft that fit both VTOL and MAV requirements edge out other aircraft types due to their wide flexibilities for completing observation mission requirements.

1.3 Challenges of VTOL MAVs

One obvious challenge of MAVs in general is their requirement for significant autonomy. Since a remote pilot cannot control them effectively from a distance, the vehicles must be fully self-stabilized as well as navigation capable without direct input from the pilot, while also being able to provide location feedback for mission guidance.

One significant problem with autonomous capability for VTOL aircraft is the problem of translational drift. Unlike fixed-wing aircraft, such as a glider or airplane, where the dynamics of which keep the aircraft close to within the desired path, VTOL aircraft can

be almost completely horizontal to the ground, but due to several factors, including an undetectable tilt, differences in airflow dynamics between the two sides of the aircraft, or from wind or other small disturbances, the VTOL aircraft can be moved translationally without detection by a body-frame based measurement. Thus, feedback using sensing that detects aspects external to the aircraft, such as a camera, is required to account for this effect.

In addition to the translational drift problem, VTOL MAVs typically have very fast dynamics and constant motion, making the control of these aircraft difficult, as well as restricting the sensors and methods that can be used to stabilize and navigate the vehicle. MAVs are also by definition small and light, limiting the general amount of sensing and processing capability that can be employed.

Aside from these general MAV obstacles, different forms of VTOL MAVs have individual challenges.

1.3.1 Rotorcraft Differences

The most common VTOL rotorcraft is the traditional helicopter, where a central rotary wing provides lift, pitch, and roll effects, while a smaller perpendicular rotor in the back controls the yaw. This aircraft style is used extensively for full-size systems, but at smaller scales, they offer significant challenges. In order to provide lift, pitch, and roll from the main rotor, complex mechanical linkages are required, which are difficult to provide effectively for good weight to payload ratios in small volumes. In addition, due to the same requirements, the system dynamics are also very complex, making autonomous control a big challenge. A rotorcraft that changes the design of the rotorcraft by using four main rotors, equally offset from the aircraft center, can drastically simplify the dynamics. These four-rotored helicopters are typically referred to as quadcopters, or quadrotors.

1.3.2 Quadrotor MAV

As MAVs became a popular field, quadrotors have taken a substantial section of the market, and their capabilities have been compared very favorably against other available

UAVs [5]. Quadrotors have a design advantage over traditional helicopters due to their simple actuation based on the separation of the motor-rotor assembly into four individual rotors at each corner of the quadrotor; a typical example of a quadrotor is shown in Figure 1.3 [6]. Their wide availability has been partially due to the enabling technology being driven by the consumer market. The development of fast, precise, and affordable accelerometers, initially driven for use in car airbags and now increasingly for consumer devices such as mobile phones have been key production components for quadrotors.

Using just thrust variations between its four rotors by simply speeding up or slowing down the motor, pitch, roll, yaw, and lift can be obtained. Each rotational freedom can be roughly treated as separate from each other, simplifying the dynamics, and thus the controls required for autonomous capabilities. This navigation capability in three dimensions, with just four moving parts, is a very desirable attribute. However, there is no advantage without some manner of trade-off, and so is the same for changing to the quadrotor design.

1.3.3 Tradeoffs of Quadrotor MAVs

The primary advantages and disadvantages of quadrotor helicopters in comparison to a traditional helicopter for VTOL MAV capabilities is indicated in Table 1.1. Similar to traditional helicopters, the quadrotor is an under-actuated system; it has six degrees of freedom, but only four control inputs. It is similarly a dynamically unstable system - without active control it will not be able to maintain flight. Some VTOL MAVs, such as coaxial helicopters or blimps, have self-stabilizing mechanics so that active control is not needed

Fig. 1.3: Typical quadrotor.

for attitude stabilization. Although the quadrotor has many practical advantages compared to the traditional helicopter, and the controller design is more intuitive, it similarly suffers from nonlinear dynamics that make it a very challenging control problem. This challenge, coupled with the advantages, is what makes this platform attractive for research; practical uses along with a problem to be solved.

1.4 Indoor GPS-Denied Environments

Indoor navigation on a MAV presents yet another challenge. Such environments limit the kind of sensors that can be used for determining location and heading. In open, outdoor settings, sensory perception through magnetometers, thermopiles, barometers, and GPS can allow full navigation capability at minimal cost and effort. In tunnels, urban canyons and inside buildings, such sensing is unreliable or unavailable, requiring different methods and means for obtaining navigation and guidance capabilities. Such navigation abilities are important to unrestricted UAV use, as search and rescue or surveillance often require operation within indoor or cramped environments.

Even the above mentioned sensing modalities are not good enough for indoor navigation, however, even if they were available. Navigation inside buildings requires excellent precision due to the more cramped nature of the setting, where just a meter on each side of the vehicle will be a wall or object.

Table 1.1: Quadrotor advantages and disadvantages compared to the traditional helicopter.

Advantages	Disadvantages
Simplification of Actuation Mechanics	High Energy Consumption
Reduction of Gyroscopic Aerodynamics	Total Weight Limitation
Large Relative Payload Capacity	Poor Survivability
Modular Components for Low Maintenance	
Symmetrical Dynamic Properties	
Reduced Safety Concerns from Small Rotors	
Can Enclose Within Frame for Protection	
Design Flexibility Gives Margin of Error	

1.5 Current Quadrotor MAV Solutions and Related Work

Quadrotors have been a relatively popular area of MAV research due to their capabilities and their interesting nonlinear control aspects. Once attitude stabilization was achieved in the latter part of the decade, navigation became possible. Outdoor navigation, with direct positioning using GPS and magnetometers, was quick to evolve. Indoor navigation, with position sensing restrictions, has been a very recent area of study with only minimal successes and few attempts due to the difficulties in sensing and processing the measurements for accurate, real-time perception.

Commercial quadrotors have become widely available in the last several years, with well-tested attitude stabilization and GPS capable navigation. Many of these systems are used to explore the problem of indoor navigation. Some of the more well known commercial quadrotors include the Ascending Technologies Hummingbird, the Mikrokopter, the Quansar QBall, the Draganflyer, the Microdrone, and the Aeryon Scout [6–11]. Some of these systems require an external sensing system in order to be able to operate effectively indoors, but some research groups have installed their own navigation systems and use just portions of the commercial system. The hobbyist community is also tightly involved with helicopter MAVs, including the quadrotor vehicle. One of the most popular flight control systems used by hobbyists for quadrotors is the Arducopter [12].

Some of the more successful indoor navigation capable quadrotor systems include: the design from Grzonka et al., using the Mikrokopter mechanical system and a laser range finder, is able to localize itself on the pre-generated map acquired from a ground robot as well generate a map onboard that closely matches the pre-generated map; the design from Bachrach et al., using an Ascending Technologies Hummingbird and a laser range finder or two stereo cameras, is able to navigate in unknown and unstructured environments; using a sophisticated offboard SLAM algorithm and an onboard downward facing camera, Blösch et al. enabled the Ascending Technologies Hummingbird to accurately hover and navigate indoors in unknown environments [13–15].

Although there have been successful implementations of indoor navigation using a

quadrotor, the problem is far from solved. Many practical limitations still exist that prevent widespread usage of these systems. One of these limitations is cost; current successful solutions make use of expensive sensing apparatus in order to obtain indoor navigation capability.

1.6 Goals of Novel Quadrotor Solution

In choosing to develop a new quadrotor solution, specific goals were outlined, either in regards to capabilities of the system or to contributions to the current state of the art.

1.6.1 Capabilities of System

For indoor navigation, certain obvious requirements are necessary, including:

- Small vehicle size for maneuvering within indoor environments,
- Attitude stabilization,
- Indoor environment sensing capability,
- Processing power necessary for extracting usable information from sensor for navigation.

1.6.2 Restrictions on Implementation Options

With these basic capabilities, restrictions are added in order to make the system different and require the application of new methods and implementations for a successful contribution to the field.

- Use only very low-cost consumer-grade materials and components.
- Perform indoor navigation using vision sensing only.
- All sensing capability must be performed on-board.

These restrictions yield a fundamentally new system that broadens the practical application of indoor navigation on a quadrotor.

1.7 Problem Statement

The hypothesis or problem statement for these goals, given the body of related work and successful implementation is:

- Autonomous indoor navigation using vision on a quadrotor aircraft with very low cost sensors is possible.

The solution approach shown below for this system, and the full implementation and experimental validation of the system covered in this thesis, answers this hypothesis with a strong affirmative.

1.8 Solution Approach

The basic solution approach for the completion of a quadrotor with the goals described above would be to develop the system from the ground up so that all components could be individually chosen based on cost and applicability to the end system. In this way, the design utilizes both a bottom-up and top-down design approaches. Bottom-up for assembling known components into a functioning design for rapid-prototyping and reduced costs, as well as top-down system development for meeting pre-determined performance and design metrics through novel application specific methods. Attitude estimation and control algorithms would be chosen based upon the requirements of the individual sensors, system dynamics, and experimental results, drawing upon current research in these areas as well as observed system response. The end system would be a full implementation of the best applicable research, modifications to current solutions as necessary, and the development of new methods for the requirements of the system.

1.9 Pushing the Envelope of Capabilities

Implementing the system outlined above will certainly address many of the needs of UAVs: it can ably hover and navigate within constrained GPS-denied environments and do so autonomously. However, such a system would provide even more benefits if it could maneuver within such constrained environments at high speeds and with high accuracy,

much like birds do everyday in forest environments. This would open up the possibility of missions involving evasive maneuvering, high wind operation, aggressive tactical maneuvers, and tracking fast moving targets.

1.9.1 Mimicking Bird Flight

Birds are highly capable flyers, able to routinely execute maneuvers far beyond the current state of the art of aeronautical and control engineering. They can perform very fast dives, experiencing forces many times their own weight, while at the same time being accurate enough to pick prey out of the air or from under the water. They perform such maneuvers through the manipulation of their wings, adjusting the way in which the wing surfaces move through the air. A common action of birds is the pull out of a dive or high speed flight, where the wings are flared at a high angle of attack, utilizing aerodynamic effects to rapidly decelerate for an accurate low-speed action, such as grabbing prey, as shown in Figure 1.4, or landing on a perch.

1.9.2 Landing on a Dime

As part of their daily routines, many birds perform a maneuver whereby they quickly land on a perch in order to survey the area or to rest. In the act of landing on this perch, due to their inability to hover using their wings, they will approach the perch rapidly, transition

Fig. 1.4: Bald eagle flaring its wings to slow down for grabbing prey out of the water.

their wings to a high angle of attack to quickly slow down before accurately landing on the perch. Such a maneuver exemplifies the requirements of both aggressive high-speed maneuvering and precision, aspects that together are still not achievable by the state-of-the-art vehicles, manned or unmanned. Thus, the ultimate goal of this thesis, in addition to the creation of a novel quadrotor solution for indoor navigation, is to develop a system capable of demonstrating accurate aggressive maneuvering using onboard sensing.

1.10 Overview of Contributions

This thesis presents the design, implementation, and experimental validation of a complete low-cost custom quadrotor system capable of autonomous attitude stabilization and indoor navigation for goal directed flight within an unknown and unstructured environment. In addition, such a system is used to showcase an autonomous perching aerobatic maneuver using only onboard sensors. The specific contributions are broken down as:

- The complete mechanical and electronic physical system design and adaptations necessary for autonomous flying on a quadrotor using only low-cost components,
- Attitude estimation techniques required for attitude stabilization on a heavy quadrotor with noisy sensor measurements,
- Attitude and navigation controllers implementation and experimental validation on a custom low-cost quadrotor,
- Adaptation of a real-time state-of-the-art visual simultaneous localization and mapping algorithm for use on a custom quadrotor,
- Attitude and navigation fusion methods for improved attitude and path tracking,
- A nonlinear controller for altitude and navigation that improves stability and tracking,
- Navigation control adjustments for obtaining accurate aggressive maneuvering,
- An onboard sensing approach to autonomous perching using a guidance level feedback and state transition controller.

1.11 Outline of Chapters

Chapter 2 describes the simplified quadrotor model that is used and the complete architectural framework that the quadrotor is built upon, including: frame and mechanical setup, sensors, processing hardware, and the communication system. All hardware and component adjustments necessary to meet flight requirements on the custom quadrotor are described and how it compares to other available systems.

With the physical system in place, the filtering, estimation, and control techniques are described in Chapter 3 for attitude stabilization on the quadrotor. The difficulties observed from the use of a problematic attitude sensor is described, along with the limitations of lower quality sensors. The control methodology used for attitude stabilization is covered, along with a discussion of techniques used by others as they apply to the system described.

Chapter 4 presents the upper level navigation system that allows for indoor navigation in unknown environments. The challenges and specifics of indoor navigation are covered along with a detailed reference to related work and successes in the area. The framework for simultaneous localization and mapping using a monocular camera is briefly covered, along with the adaptations of this algorithm necessary for proper use on the quadrotor. The navigation control system, and results from a typical hover and trajectory are presented.

In Chapter 5, a nonlinear control method is introduced to mitigate instability when the quadrotor is far from the desired set point. Experimental results are shown with a comparison to the linear controller method.

Bringing the system to the edge of its capabilities, an aggressive perching maneuver is demonstrated in Chapter 6 using only onboard sensing, with modifications to the base system described in the previous chapters and adding a guidance level feedback and state machine based upon an additional sensor.

Finally, Chapter 7 presents concluding remarks and a discussion of future work for the system.

Chapter 2

Physical System and Architecture

Any implementation of a system starts first with a general understanding of the system mechanics and functional requirements. A dynamic mathematical model of the system will yield the information necessary for initial system design.

The development of the quadrotor mechanical and electronic system involved a significant examination of the available commercial items, their costs, and an integration based upon previous quadrotor designs. Since this quadrotor design has significant cost requirements, the overall goal of each component was that it was easily obtainable, widely used in at least the hobbyist market, and would not contribute a significant percentage of the budget by itself. This led the component selection to be either off-the-shelf ready for use, or easily customizable in house, with limited tools and capabilities.

This chapter covers the functional and dynamic properties of a general quadrotor helicopter, specifics of the mechanical and electronic architecture of the quadrotor implementation and how all the components fit and work together.

2.1 Quadrotor Dynamic Model

A quadrotor helicopter is a rotor craft with two pairs of counter-rotating rotors of a fixed-pitch located at the four ends of the aircraft, as shown in Figure 2.1. The quadrotor is maneuvered by varying the rotational speed of the rotors in order to manipulate the thrust. Pitch and roll angles, defined as the front/back and left/right angles, are controlled using moments generated by a differential thrust between rotors on opposite sides of the vehicle. The yaw rotation is controlled using the difference in reaction torques between the pitch and roll rotor pairs, as each pair is rotating in opposite directions and thus generating a torque due to air friction that is opposite to each rotor's direction of rotation. Vertical motion

is controlled by adjusting the total thrust of all rotors together, and lateral acceleration is achieved through a pitch and/or roll of the aircraft.

With four actuators and six degrees of freedom (roll, pitch, yaw, x, y, z positions), the quadrotor is an underactuated system. Quadrotor helicopters, like traditional helicopters, are dynamically unstable. Unlike some fixed-wing aircraft, left without active control the quadrotor will diverge into instability.

The model has been derived using Newtonian mechanics under the following assumptions:

- The effects of the body moments on the translational dynamics are neglected,
- The center of mass and the body fixed frame origin coincide,
- The ground effect is neglected,
- Blade flapping is un-modeled,
- Friction is only considered in the yaw motion,
- The frame structure is rigid,
- The helicopter structure is symmetric (diagonal inertia matrix - no axis cross-coupling),
- Thrust and drag are proportional to the square of the speed of the propellor.

Aerodynamic effects, such as blade flapping, rotor body dynamics, rotor flapping due to yaw, and variable inflow velocities as a result of craft pitch and roll are ignored in the model presented here but have been modeled in other quadrotor systems [16–19].

The dynamic model, with the assumptions above and ignoring aerodynamic effects, is essentially a rigid-body model with just abstract force and torque actuators and no aerodynamics. The model here has adjustments to the well-known rigid-body model, with the inclusion of an additional gyroscopic term caused by the rotation of the airframe due to the counter-rotating propellors, as well as four additional equations describing the dynamics of the four rotors [20]. The derivation of the nonlinear dynamics is performed in North-East-Down (NED) inertial and body fixed coordinates. Let $\mathcal{I} = e_x, e_y, e_z$ denote the inertial

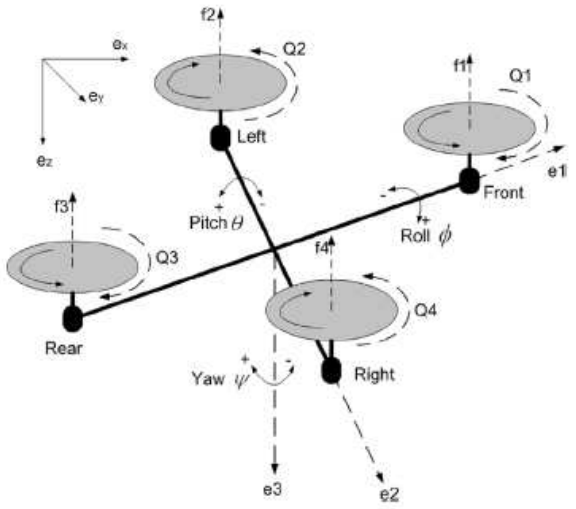


Fig. 2.1: Model of a quadrotor.

frame attached to the earth, relative to a fixed origin, and $\mathcal{B} = e_1, e_2, e_3$ denote the aircraft body frame as shown in Figure 2.1, where the origin of the frame is considered to coincide with the center of mass of the aircraft. Then the dynamic model is:

$$\dot{\xi} = \upsilon, \tag{2.1}$$

$$\dot{\upsilon} = ge_z - \frac{1}{m}TRe_z, \tag{2.2}$$

where the vector $\xi = [x\ y\ z]^T$ represents the position of the origin of the body-fixed frame, $\mathcal{B}$, with respect to the inertial frame, $\mathcal{I}$; the vector $\upsilon = [\upsilon_x\ \upsilon_y\ \upsilon_z]^T$ represents the linear velocity of the origin of $\mathcal{B}$, expressed in the inertial frame and $e_z = [0\ 0\ 1]^T$ is the unit vector in the inertial frame, $\mathcal{I}$; g is the acceleration from gravity (9.81 m/s^2) and m is the mass of the vehicle. The orientation of the vehicle frame in space is given by the orthogonal rotation direction cosine matrix (DCM), $R \in SO(3)$, and described by the three Euler angles, ϕ, θ and ψ of roll, pitch, and yaw. The representation of this rotation DCM, R, to rotate between the inertial NED axes to body fixed axes is the sequence of rotation in ψ, θ, ϕ, or yaw, pitch, roll, about the z axis, then the new y axis, and then the new x axis.

Therefore, R takes the form

$$R = \begin{pmatrix} \cos\psi\cos\theta & \cos\psi\sin\theta\sin\phi - \sin\psi\cos\phi & \cos\psi\sin\theta\sin\phi + \sin\psi\sin\phi \\ \sin\psi\cos\theta & \sin\psi\sin\theta\sin\phi + \cos\psi\cos\phi & \sin\psi\sin\theta\cos\phi - \cos\psi\sin\phi \\ -\sin\theta & \cos\theta\sin\phi & \cos\theta\cos\phi \end{pmatrix}. \quad (2.3)$$

In Equation (2.2), T is the thrust generated by the four rotors in free air and given by

$$T = b\sum_{i=1}^{4}\omega_i^2, \quad (2.4)$$

$$Q_i = k(\omega_i^2). \quad (2.5)$$

Q_i is the reaction torque generated in free air by the rotor due to drag. k and b are two constants of proportionality parameters that depend on aerodynamic effects, including the density of the air, and the size, shape, and pitch angle of the rotor blades. k is on the order of 1.1×10^{-6} and b is around 2.9×10^{-5}. The model continues with

$$\dot{R} = R \cdot sk(\Omega), \quad (2.6)$$

$$I_f\dot{\Omega} = -\Omega \times I_f\Omega - G_a + \tau_a, \quad (2.7)$$

$$I_r\dot{\omega}_i = \tau_i - Q_i,\ i \in \{1,2,3,4\}, \quad (2.8)$$

where Ω is the rotational velocity of the vehicle in the body frame, $\mathcal{B}$. $sk(X)$ denotes the creation of a skew-symmetric matrix generated using the vector inside the parenthesis, such that $sk(X)Y = X \times Y$ for any vector $X \in \mathbb{R}^3$ with $\times$ denoting the vector cross product. I_f is the inertia matrix of the airframe with respect to the body frame, $\mathcal{B}$, measured in $kg \cdot m^2$, where the center of mass coincides with the origin of the frame. I_r signifies the moment of inertia of the rotor blades, and is roughly $I_r = 3.4 \times 10^{-5}\ kg \cdot m^2$, while ω_i is the speed of the rotors 1,2,3, and 4. G_a is the gyroscopic torque due to the combination of the rotation

of the airframe and the four rotors.

$$G_a = \sum_{i=1}^{4} I_r(\Omega \times e_z)(-1)^{i+1}\omega_i. \tag{2.9}$$

τ_a is the airframe torque generated by the rotor and given by $\tau_a = (\tau_a^1, \tau_a^2, \tau_a^3)^T$, with

$$\tau_a^1 = L \cdot b(\omega_2^2 - \omega_4^2), \tag{2.10}$$

$$\tau_a^2 = L \cdot b(\omega_1^2 - \omega_3^2), \tag{2.11}$$

$$\tau_a^3 = k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2), \tag{2.12}$$

where L is the distance from the rotors to the center of the aircraft. τ_i is contrasted from τ_a, with τ_i the four control inputs to the system, in the form of motor torques, $\tau_i,\ i \in 1, 2, 3, 4$.

With this model defining the dynamics of the quadrotor, the physical system architecture can be developed accordingly.

2.2 Physical System Overview

The complete quadrotor system was developed from scratch, built with low-cost constraints and thus custom-made from available consumer components. Figure 2.6 gives a breakdown of the individual critical parts of the system and how they communicate with each other. The system uses the following major components:

- firmware modified Turnigy Plush electronic speed controllers (ESCs) [21];
- KDA20-22L brushless direct current (BLDC) motors from HobbyKing [22];
- inertial measurement unit (IMU) consisting of gyroscopes and accelerometers:
  - InvenSense ITG3200 MEMS digital output 3-axis gyroscope [23];
  - Bosch Sensortec BMA180 digital triaxial accelerometer [24];
- MaxBotix EZ2 SOund Navigation And Ranging (SONAR) module [25];

- Linux-based Gumstix Verdex Pro XL6P computer on module (COM) with a Wi-Fi network card [26–28];
- two Robostix Atmega128 microcontrollers [29, 30];
- monocular camera equipped with a wide angle lens;
- infrared (IR) blob detecting camera;
- mini servo.

The Gumstix, two Robostix, IR camera, and IMU sensors communicate over inter-integrated circuit (I^2C) protocol. One of the Robostix controllers reads in the sonar on its analog digital converter (ADC) and also outputs pulse width modulated (PWM) signals to the ESCs to control the motors. The other Robostix operates the mini servo. The monocular camera transmits images through a USB cable linked to the ground station. The ground station consists of an Intel Core 2 CPU 2x2.4 GHz processor running Linux.

The complete system weighs 1.4 kg and measures 50 cm from end to end of the frame. Total system is very low cost, at a rapid-prototype price of $1,000, with the IMU sensor chips themselves accounting for only $18. All sensors combined - including the IMU sensors, cameras and sonar - only cost a total of $150.

2.3 Mechanical Architecture

The basic structure of the quadrotor was determined from an examination of the current quadrotor market, including both research based systems as well as hobbyist systems, as mentioned in Chapter 1. Given the project goals of making the quadrotor small, cheap and indoor capable, the frame structure was chosen to be primarily aluminum and carbon fiber materials, with an overall size of about 50 cm from motor-to-motor on a given axis for controllability at the expected quadrotor weight [31]. The Mikrokopter MK-50 quadrotor frame is a perfect match for the project's requirements [7]. It is available separately from the complete quadrotor made by Mikrokopter, and many of the parts are standard sizes. In addition, the frame by itself, including four frame rods, fiber glass center plates, screws,

standoffs and dampers, is a very reasonable price. Mikrokopter had just opened a US shop, and so shipping would also be inexpensive. The order was placed, unfortunately, due to the high demand of these quadrotors, delivery of the parts was to be over a month away.

2.4 Hand-Made Platform

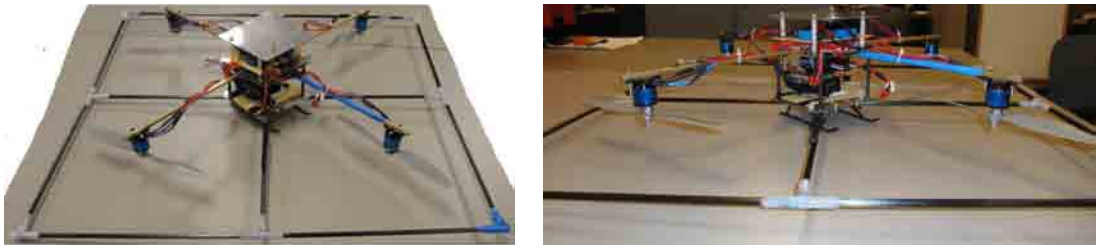
In the time between waiting for the MK-50 frame, all the other needed parts had arrived. It was decided that some progress could be made using a custom frame, based upon the MK-50.

2.4.1 Description

Aluminum rods were ordered, with aluminum sheets cut into squares for the supporting structure. All screw holes were hand drilled using a drill press. Two aluminum 5 in. x 5 in. sheets sandwiched the brass $\frac{1}{4}$-in. rods holding the motors, with aluminum standoffs going up and down from the center for holding the electronic components. A large carbon fiber shroud was made and attached to the bottom, which extended out beyond the frame to protect the propellers. The final product is shown in Figures 2.2(a) and 2.2(b).

2.4.2 Analysis

Although the hand-made platform was able to achieve flight, and in some cases stay airborne in a 50 meter square space for up to 20 seconds, it was fraught with stabilization problems. Uncontrollable yaw lead to constant spinning, which is the only reason the



(a) Top view. (b) Side view.

Fig. 2.2: Custom hand-made platform.

quadrotor was able to maintain being airborne for any reasonable amount of time, as the attitude was often seeing large angles going uncorrected for most of the flight.

Two primary reasons for the difficulties faced with this frame are:

- Errors in the frame structure construction. As will be discussed later in this chapter regarding inertia and center of gravity - as well as will be mentioned in Chapter 3 regarding attitude control methods using the quadrotor dynamics - accuracy and consistency in the frame weight and balance and thrust direction and differentials of the motor and prop system have a huge impact on stability [31].

- Inaccuracies in and noise coupling onto the inertial measurement unit (IMU). The problems with the initial IMU and vibration and noise coupling in general, are discussed in Chapter 3.

2.4.3 Precision Frame from Mikrokopter

By the time the MK-50 frame arrived, it was understood that the hand-made frame contained several issues. In addition to being ungainly and poorly constructed, many of the materials used were unable to withstand the forces imposed on them, even discounting crash damages. As shown in Figure 2.3, the MK-50 frame is precision made, with good rigidity and strength.

Fig. 2.3: Mikrokopter MK-50 frame.

2.4.4 Landing Gear

Originally the Mikrokopter tall landing gear was used, but its bendy plastic construction was not sturdy enough for the heavy quadrotor, especially on landing or crashes and eventually became tilted in one direction. The landing gear currently used is a cheap and readily available radio controlled (RC) helicopter landing gear built for the TRex-600, made by Align [32]. Although of a heavier weight than desired (about 180g), its sturdiness and ease of attachment made it a good fit. The only concern other than the weight was for the landing gear to be high enough to accommodate components such as the camera in the belly of the quadrotor. Extensive searching for taller, but still sturdy landing gears yielded nothing that was within a reasonable price range, so much of the mounting style of the components described in this chapter is a consequence of the room allowable by the relatively short landing gear.

2.5 Motors and Thrust

The quadrotor is equipped with KDA20-22L motors [22]. These were picked based on their price and operational speed range. They operate at a voltage level of 11.1 V, which is one of the most common and widely used by both hobbyists and other quadrotor platforms. These motors fall into the necessary category of slow flying motors, having a reduced revolutions per minute (RPM) limit in exchange for thrust variation in the middle. Higher RPM motors are used for fast fixed-wing vehicles. With these basic requirements, plus cost and availability as well as general quality, the KDA motors were chosen, utilizing a hobby motor comparison table found on a forum that is based on product and test information [33].

Given that the motors are fairly inexpensive imitations of a brand name motor, there have been quality issues with them. Often one of the inner bearings will become worn or will start catching, causing the motor to spin up later than the others and thus preventing a proper liftoff. This is sometimes noticed in flight as well in the manner of poor responsiveness. Changing the bearings has been a way to keep motors going, although it depends on having other motors that, due to crashes, are completely irreparable. The primary effects

of the lower quality is the consistently changing characteristics of the motor, even in flight, which affects the amount of thrust being produced at a given input command. This is likely due to heating and friction affects that are not mitigated well by the motor materials and construction.

2.5.1 Propellers

To fit the size of the motor and the quadrotor frame, a 10x4.7 in propellor is used. The brand is an APC slow flyer, and it was chosen because of its use on the hobbyist platform, Aeroquad, and from comparing it to other props based on web user feedback [34]. The availability and low cost were also factors. The slow flyer configuration is a requirement for the quadrotor due to the necessity for high thrust at low RPMs; the slow flyers have a blade width and angle that create thrust at lower speeds equivalent to fixed-wing propellors that generate thrust at high speeds with a different configuration. Props in general must be matched to static thrust conditions, so thin and flexible rather than chunky with high pitch angles are desirable for quadrotors [35]. These APC propellors are made with composite materials, which is good for sturdiness, although it is breakable on landings. Plastic props were found to be too flimsy and prone to thrust problems due to bending, although crash survivability was higher. Propellors rotate clockwise for pitch, vice versa for roll.

Propellor Attachement

Properly attaching the propellor to the motor is obviously an important task. There are different kinds of propellor attachments for different situations. Originally the propellors were attached using a hard mount fixture which mounted to the top of the motor. This lead to a good connection, but the motor and propellor system ended up being very tall and so it became quite difficult to adjust the rotor plane according to the center of gravity. Propellor attachments known as prop savers attach the prop to the motor using the motor shaft, and can often lead to the benefit of the propellor coming off the shaft during a crash instead of just breaking. One prop saver that was tried uses a rubber band type attachment that wraps over the prop to a piece that attaches to the shaft. This proved unworkable since

the set screw attachment was never tight enough to prevent the attachment from coming off, plus the propellor was able to move too freely, dramatically reducing downward thrust. The final choice is a collet style prop saver, which is able to hold quite adequately to the roughly 1 cm of available shaft length, while keeping the propellor stable, secured to the shaft and centered properly. Although propellors still break on crashing like with the hard mount attachment, some of the time the collet will just pop off the shaft with no injury to the propellor. An occasional detrimental effect due to the strong attachment to the shaft, is that on some crashes, the actual shaft will break off with the collet still attached, requiring disassembling of the motor for a shaft replacement.

2.5.2 Motor Characteristic Identification

Knowing the time constant of the motor is necessary for performing proper model based control. The motor with electronic speed control (ESC) system was tested using LabView to measure the signals of the controller output and the speed of the motor [36]. The speed of the motor was measured using an infrared (IR) light emitting diode (LED) and photodiode that were set up to allow the propellor to spin between them, which is accurate and simpler to set up than measuring the back-EMF (electro-magnetic feedback) of the ESC [35]. The time constant was measured over a variety of thrust ranges, but primarily around the actual flight envelope, as that is where the system dynamics are taking place. The time constant is 80 ms, which is faster than expected for such a large motor, although not as fast as the 50 ms time constants measured by some of the smaller motor commercial quadrotors. The adjusted ESC firmware, described in section 2.5.4, is likely responsible for the reasonably low time constant compared to expected, as feedback control has been shown to improve motor-ESC-prop response time [18]. No issues with high current draw during the step response measurements were noticed [35].

2.5.3 Thrust

The motors are rated at 924 KV and an estimated 940 grams of thrust each [22]. This would yield a net total quadrotor thrust of 3,760 grams. Based on previous research on

quadrotors and other hovering vehicles, a necessary thrust margin is required in order to maintain attitude stabilization, as one motor needs to thrust up in order to adjust the appropriate axis to level flight [31, 37, 38]. This thrust margin is estimated at 70%, giving a net flying thrust of about 2,600 grams. Measurements of the RPM at various inputs gives an estimated max RPM, which when calculated for actual thrust for the type of prop used, indicate that actual total thrust is much lower for sustained flight, on the order of 3,000 grams at an altitude of 4,500 feet. A conservative estimate of 2,100 grams of thrust is then available for flying.

From measuring the RPM vs input data, the input resolution to the ESC of 1 μsec is equivalent to 10 RPM of the propellor, or a minimum resolution of 2.5 g of thrust at hover velocities for an elevation of 4,500 feet.

2.5.4 ESC

Unlike common brushed DC motors, BLDC motors require an ESC to appropriately command the motor to spin, since BLDC motors are commutated electronically instead of electro-mechanically. The BLDC motor contains a fixed number of poles, or coils. The motors used in this project are outrunner motors and so the coils are located on the inner part of the motor, while the outer section contains the magnets. The poles on the motor are divided into three sections and the ESCs control the timing for energizing each third section, as a three-phase system, so that the outrunner will spin in the appropriate direction. Current runs through only two of the phases at a time, while the third is floating, and is used to measure back-EMF for speed detection and regulation. A three-phase H-bridge controls the commutation electronically. The ESC activates the H-bridges by interpreting a received pulse width modulated (PWM) signal high voltage of a time width value between the specified minimum and maximum widths.

ESC Model and Control

Brushless motor speed dynamics are composed of a single-poled dynamic system. A proportional feedback speed controller is modeled

$$\dot{w}_i = k_m \cdot (w_i^{desired} - w_i), \tag{2.13}$$

where w_i is the speed and k_m is the proportional motor gain. A proportional controller is determined to be acceptable for a good response, given the motor system dynamics [35]. Standard ESCs utilize internal speed control with a proportional feedback controller based on the sensed back-EMF. For typical BLDC motor applications, specifically in flying vehicles, this leads to quite acceptable results, however back-EMF is highly inaccurate for slow speeds.

ESC Update Rate

Turnigy Plush 30 Amp ESCs are used and require a minimum of 1 ms for lowest throttle, or off, and 2 ms for maximum [21]. Since the ESC requires the waveform sent to be at least 2 ms long, the theoretical maximum update is 500 Hz. Many ESCs used to have only a 50 Hz update [38]. This was due to the restrictions of hobby RC hand-held receivers, which operate at 50 Hz, and so faster update ESCs were not required. Lately however, commercially available and cheap ESCs are produced that use the full update frequency potential available. Although other users have used similar ESCs at a 500 Hz rate, the Turnigy ESC was never attempted at higher than 400 Hz, which was quite acceptable given the relatively much slower time constant of the motor itself [31].

ESC Command Resolution

The ESC detects input throttle commands in a range of 1-2 ms and is theoretically capable of detecting 1 μsec increments. However, the ESC output command to the motor utilizes an 8-bit number, thus drastically limiting the actual accuracy available for speed commands to the motor. Using a finer resolution than 1 μsec for processor output commands

to the ESC will not realize any improvement for the extra computation.

ESC Software Filtering

Although the ESC motor driving requirements are fairly simple and straightforward, there is a key aspect that is common for ESCs that are purchased through the hobbyist line, as opposed to specifically for quadrotors or designed internally. This crucial piece of information is how the ESC deals with constantly changing inputs. In a quadrotor, where a small thrust differential on one axis can cause a definite movement, fast responses to the actuator are required for stabilization. Thus, motor inputs need to inherently change constantly during flight based on commands. Hobbyist ESCs are mostly built for fixed-wing aircraft, in which the motor is only giving thrust, while other actuators are controlling the attitude stabilization. This means an ESC that operates to change its output directly with changing inputs will do nothing but incur power consumption of trying to change a motor's rotational speed without a specific requirement. Thus, ESC manufacturers include a software low pass filter to smooth out the input commands for improved energy use. As can be inferred, such filtering is highly problematic for a quadrotor, as the affect is an increased actuator time constant, and thus a much reduced stability area of attraction.

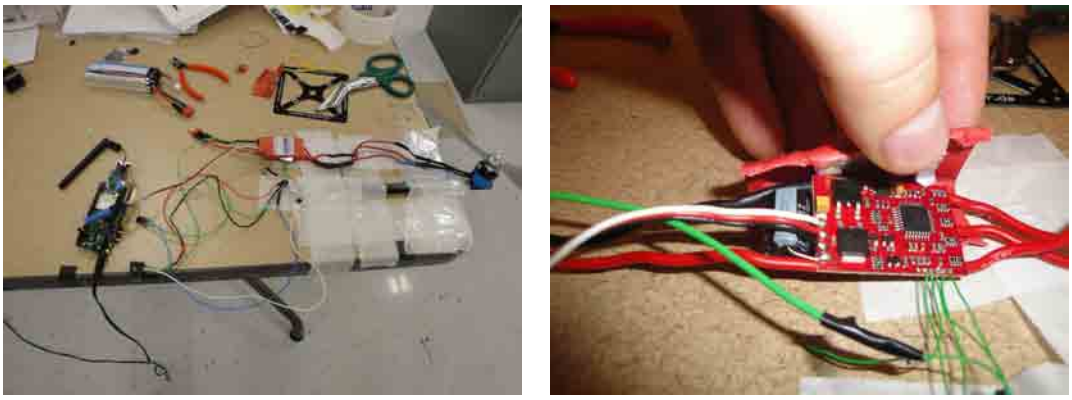
A built-in ESC tuning feature supposedly is used to mitigate some of the output averaging, called fixed throttle mode versus auto-calibrating mode. Most ESCs for airplanes assume the application of full throttle on take off, so the ESCs measure the total range of the input PWM signal and use that as the throttle range, thus limiting fast changes to around the initial application level. If, as in the application of the quadrotor, the ESC is constantly adapting to the signal it sees due to larger than expected variations, it will cause reduced response time in adjusting the signal as the calibration is constantly changing. However, utilizing the fixed throttle mode for the full range of the flight envelope of the quadrotor did not produce a noticeable change in response time, as the in-built low pass filter still dominated the net change to the motor.

2.5.5 ESC Firmware Reflash

The discovery of the low-pass filter on the ESC included both empirical testing as well as a search of the forums on hobbyist created quadrotors. During testing, the quadrotor would often attain very level flight for significant periods of time, only to fall to the floor completely when slightly disturbed or when an oscillation started. At high gains, this was readily apparent, though the cause was not obvious. During testing with the quadrotor held in the hand, fast movements would sometimes completely stop one of the motors.

Fortunately, the hobbyist community had seen the same issue and had developed a solution. An ESC firmware assembly code had been written that could be tweaked according to needs and installed onto most ESCs [39]. Due to the inherent slight differences between each ESC manufacturer, this was sometimes a non-trivial task. Using and modifying this code according to the Turnigy Plush ESC with the requirements of at least 400 Hz update frequency and minimal low pass filtering, the modified firmware was flashed onto the Atmega8 ESC processor [40]. The test stand is shown in Figure 2.4(a) and the programming hardware setup is shown in Figure 2.4(b). Code compilation, debugging and simulation as necessary were performed using the Atmel AVR Studio 4 [41].

Some curious observations seen from various modifications to the firmware were that the internal clock was not very accurate, and no external crystal is available, and so specifying correct PWM high width values for low and high throttle, as well as operational frequency,



(a) Test stand. (b) Wiring.

Fig. 2.4: ESC programming setup.

required giving a margin of error in the code. Setting the calibration bit for clock regulation prevented the ESC from ever initializing.

Input speed change testing was performed on the ESC before and after the new firmware and a much faster response was verified, as the motor was much more directly responding to the required changes that the attitude controller was sending, and there were no drop-offs in speed with high frequency changing inputs. The fully reflashed ESCs for quadrotor use are shown in Figure 2.5.

Another possible reason for a reduced response time, or averaging effect, may be due to a built-in-slew limit for reducing current inrush draws during step changes [18]. No evidence of power issues from a high current effect was noticed with the reflashed ESCs, however these ESCs are over-designed for the required standard currents of the motors used.

Obvious consequences of this reflash include a motor action much more directly in line with actual commanded changes. Less obvious is getting a much faster response time from the system, as the ESC attempts to respond as fast as possible to a changing input, rather than ramping up due to averaging. As such power requirements do increase.

I^2C Modification

This ESC and many others are capable of being modified to run on I^2C input. The modification has been successfully done by many hobbyists and has instructions similar to the reflashing for software filter removal. However, it is still a delicate task. Having the

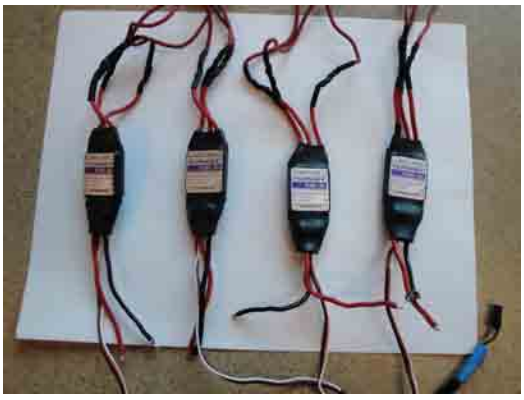


Fig. 2.5: ESCs after programming.

ESCs commanded over I^2C does not yield any real loop timing improvements as all hardware and processing components would remain the same. However, it does have a possible simultaneous command issue, as the I^2C commands to each ESC would be sent sequentially, while the PWM commands essentially all update together. The update frequency would also remain roughly the same due to the internal control loop limitations, so this modification was not performed.

2.6 Computation and Communication Hardware

An autonomous vehicle requires the use of communication and controlling hardware in order to run the control loops and interface with the sensors. For this project, a Linux based single board computer (SBC) with a separate low level processor for ADC and PWM output are used for the attitude control, with a Wi-Fi module for uploading the program and downloading post-flight data. A wireless device is used for communication between the host computer, which performs the manual control and navigation processing. A full electronic component and communication diagram is shown in Figure 2.6.

2.6.1 Processor System

The onboard computing system consists of a Verdex Pro XL6P Gumstix computer, with a Robostix port expansion board and Netpro-Wifi module [26–29]. The Gumstix is a small embedded 600MHz Linux-based SBC. It runs on a Marvell PXA270 (with XScale) processor based on the ARM architecture and comes with 64 MB DDR RAM and 32 MB NAND Flash memory. Its dimensions are 80mm x 20mm x 6.3mm and weighs 8g. It has expansion connectors on each sides for attaching to expansion boards. The Gumstix was chosen due to it being already available in the lab and having the requisite abilities, with several other research groups using it successfully, albeit with different setups and requirements. Drawbacks are of course the non-dedicated nature of the control, being on a full Linux hosted processor, as well as the inability to directly access the ports or control at the microprocessor level. For this requirement, a Robostix was needed for much of the low-level action. An additional drawback of the Gumstix, is its lack of a Floating Point

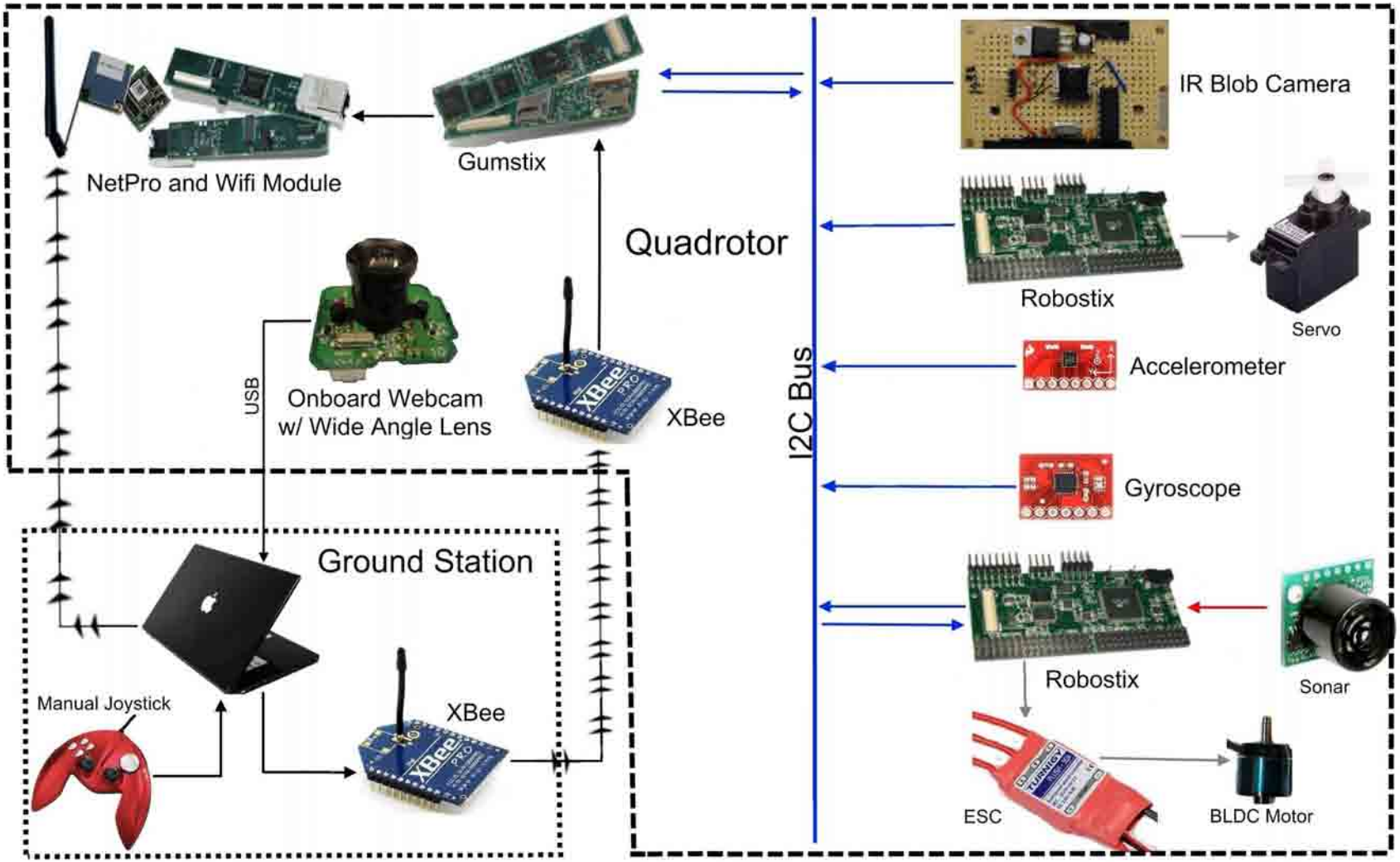


Fig. 2.6: Component level breakdown of the quadrotor system, including communication lines. Blue lines are I^2C communication, the red line is analog input, gray lines are PWM outputs, and black lines are serial connections.

Unit (FPU), requiring floating point computations to be implemented in software and thus increasing computational requirements.

The 600MHz Gumstix was chosen after the 400Mhz version had been used, hoping to achieve a little better performance, but no measurable difference in loop frequency was realized.

2.6.2 Zigbee

A Zigbee module is used to send the packet data of the manual controller and navigation system to the quadrotor [42]. This system was chosen due to its high reliability, low power, robustness, and ease of interfacing. Individual registers are set and the firmware was slightly modified in order to obtain: communication rate operation at 115200 baud, interfacing over a serial line and minimal delay.

2.6.3 Robostix

The Robostix is an expansion board for the Gumstix equipped with a suite of input/output (I/O) ports; it has an Atmel ATMega128 processor and communicates with the Gusmstix via the I^2C serial protocol [30]. The Robostix is required as an ADC and for being able to get I/O from the Gumstix. The Robostix takes in the sonar sensor to the ADC and outputs the four ESC PWM signals. Power to the Gumstix is also routed through the Robostix. A second Robostix is also on the quadrotor, and is used to control a servo, which is used as part of a separate thesis [43]. The way the Gumstix and Robostix processor combination is installed on the quadrotor platform is shown in Figure 2.7.

2.7 Sensing Hardware

A full suite of sensors are needed in order to give enough information to the quadrotor for attitude stabilization, heading or yaw control, altitude control and positioning detection. This system has attitude sensors in the form of a 3-axis accelerometer and 3-axis gyro for attitude and yaw stabilization, a sonar for maintaining height, and two cameras for detecting position. These sensing components are shown in Figure 2.6.

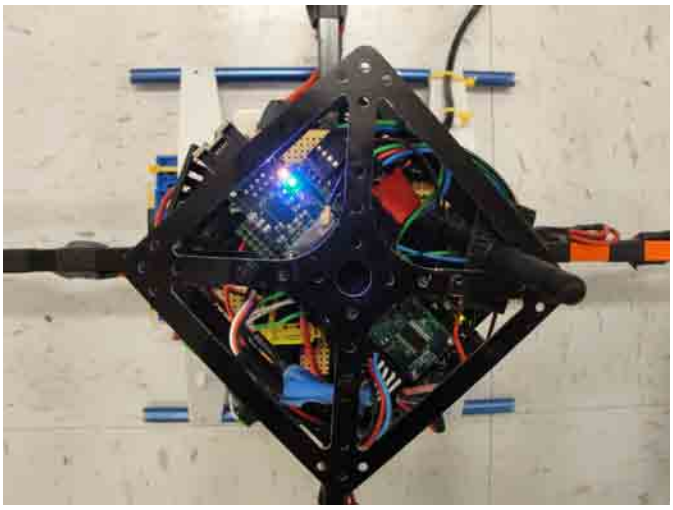


Fig. 2.7: View of processors installed on quadrotor.

2.7.1 Attitude Sensors

The quadrotor has two primary sensors for attitude control: a 3-axis gyroscope for detecting angular rates and a 3-axis accelerometer for determining accelerations and consequently angles. The ITG-3200 is a gyroscope specifically made for roll, pitch, and yaw angular rate detection [23]. The BMA180 is the 3-axis accelerometer [24]. The ITG-3200 and the BMA180 were pre-packaged on a small circuit board by Sparkfun for simplicity of rapid-prototyping use, and are shown mounted to the component board in Figure 2.8 [44].

These sensors were chosen due to their low cost, I^2C interface, and onboard filtering capability. With the processor loop frequency limitations, using only onboard filtering would yield large delays; reducing the communication and filtering requirements by eliminating the need for onboard ADC and filtering is needed.

Sonar

The quadrotor has a MaxBotix EZ2 as the sonar module for detecting height [25]. This sonar was chosen due to its low cost, availability, and ready to use package. It gives analog readings at 20 Hz at a resolution of 1 inch (2.54 cm), although when processed

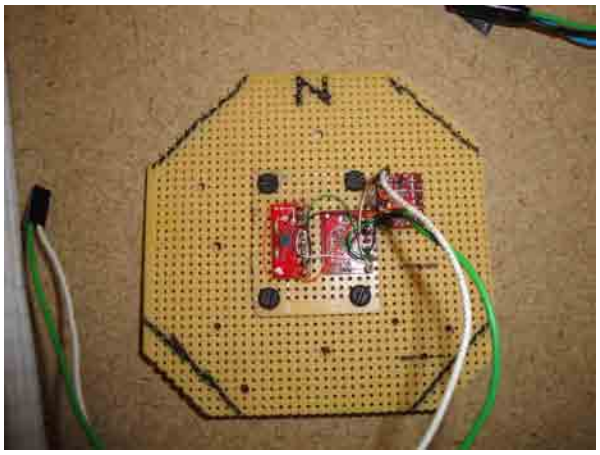


Fig. 2.8: Gyro and accelerometer sensors mounted to the component board.

through the 10 bit ADC on the Robostix, the quantization will give net measurements of 0.5 inches (1.27 cm) resolution. One important characteristic of the sonar is that it gives a constant 6 inches (15.24 cm) reading when between 0 - 6 inches (15.24 cm), which is a crucial feature for takeoff and landing control. The EZ2 sonar has a fairly narrow beam width characteristic, which is important for measuring height above a large plane, so that occasional objects nearby do not interfere with the reading.

2.7.2 Cameras

The quadrotor is equipped with a wired web camera used for navigation and is discussed in detail in Chapter 4. There is also an IR blob detecting camera that is discussed in Chapter 6. The way the cameras and sonar are mounted onto the quadrotor is shown in Figure 2.9.

2.8 I^2C - Communication Backbone

I^2C is a commonly available multi-master serial protocol, originally developed by Phillips. Only the Gumstix is the master on the quadrotor since it is the primary processor. The Gumstix polls the Robostix for sonar sensor data via the I^2C bus, and sends updates of the ESC PWM values. The I^2C bus on the Gumstix is configured to work in Fast I^2C mode, yielding a theoretical peak data rate of 400 Kbps, which, with overhead of starting and stopping communication and the system calls required, the estimated data rate

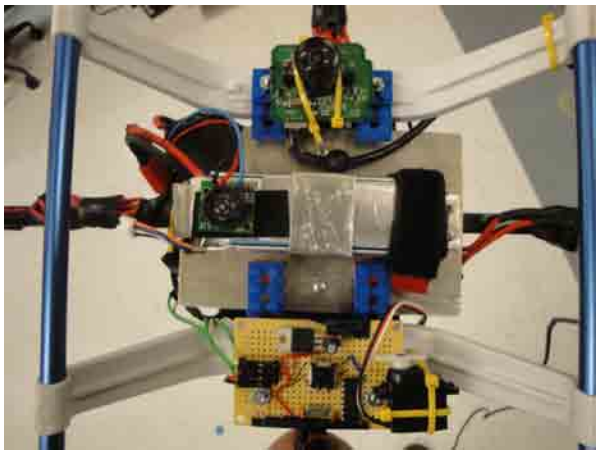


Fig. 2.9: Camera and sonar sensors mounted on the quadrotor.

drops to 150-200 Kbps. Although I^2C is capable of operating at faster speeds, the Gumstix and most currently available hobby components do not support it.

Figure 2.6 shows the setup of the I^2C bus for the quadrotor components. The only missing component is a 3.3 V logic converter that is used to step down the I^2C 5 V signals from the Robostix to 3.3 V for use by the IMU and the IR Camera [44]. The converter hooked up to the IMU is shown in Figures 2.8 and 2.11(a). A side view of the quadrotor showing the power, communication and mounting setup of the built system is in Figure 2.10.

2.9 Ground Station

Any autonomous vehicle is not complete without all of the interfaces and programming setup required to bring the vehicle to a safe and operational status. The ground station for the quadrotor consists of a host computer, which is used for the manual controlling interface, navigation software processing and for setting up, compiling and installing the on-board software.

2.9.1 Manual Control - Operator Interference

The quadrotor can be controlled manually using a standard gamepad with joystick buttons. This is primarily for safety, but is also utilized for specific actions which are

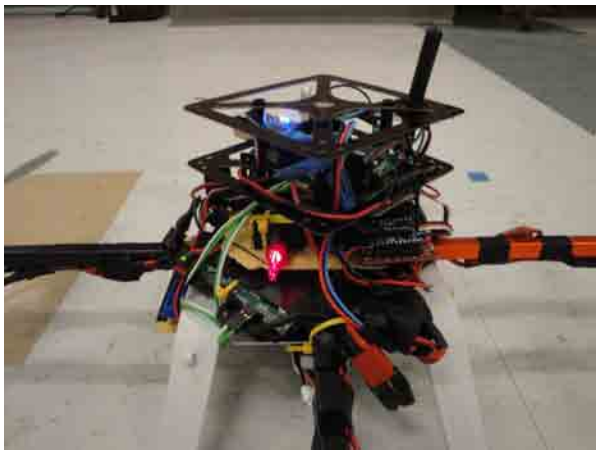


Fig. 2.10: Side view of the completed quadrotor.

discussed in Chapter 3. The gamepad has a common USB interface, with both analog control sticks as well as buttons. A typical joystick reading program is implemented on the host computer which simply parses the button details over USB and then sends the values over serial to the Zigbee so it can be received at the quadrotor and utilized within the control loop. This information is sent in four byte packet format common with the navigation system, containing the button type and value whenever one is changed from the previous value.

2.9.2 Host Computer Setup

The ground station computer runs native Linux, using Ubuntu 10.1. This allows a common base between the host computer and the Gumstix, and simplifies the use of the navigation software.

2.9.3 Operational Setup

In order to use the Gumstix, a special program called OpenEmbedded Build system that uses Bitbake is required. The OpenEmbedded build is used to create standard images for a root file system and Linux kernel to be used on the Gumstix. It additionally sets up a cross compilation tool chain, which is a set of utilities needed to compile C code on the desktop computer that will then be installed on the Robostix (ATMega128) micro-

controller. The OpenEmbedded build system is an open source code, so significant uses of peer-based online support through the Gumstix mailing list was utilized to work out the bugs specific to the situation. All programming for the Gumstix is done using Ubuntu Linux and it takes around eight hours to compile the entire OpenEmbedded system. One issue with this initial setup is the build system will refuse to compile on some of the later versions of Ubuntu and apparently is only currently able to compile properly with Ubuntu 8.04 LTS. As such, a virtual machine of Ubuntu 8.04 is on a separate computer used for compiling new Gumstix code.

Once the OpenEmbedded build system is working correctly, the next step is to establish a serial connection to the Gumstix, using a software utility such as Kermit. The onboard flash memory must be updated with the kernel and root file system. The Gumstix does not have a serial connector on it due to its small physical dimensions and so a breakout Tweener board is attached to the Gumstix for accessing the serial port [45]. The OpenEmbedded build system uses the "ipkg" package management tool. In order to be compliant with this package management, a packaging tool called bitbake that comes with the build system is required. Bitbake needs "make" files for compilation. Upon compilation it packages the code into an installer package which can be transferred to the Gumstix over Wi-Fi and installed using the "ipkg" manager. This completes the configuration of the Gumstix with any recently compiled code.

The next step is to configure the Robostix and requires installing an I^2C based ATMega programmer on the Gumstix for programming the Robostix over the I^2C bus. A small hardware modification needs to be made to the Robostix board in order to achieve this: the installation of jumpers to enable the Gumstix and Robostix boards to talk to each other. With this completed, sending programs to the Robostix is a straight-forward process.

Compiling C code for the Robostix requires another cross compiler so that the code can run on the ATMega processor. The AVR-GCC compiler is used to compile code for the ATMega Controller.

Even though, the Gumstix COMs come with a Linux kernel and a file system, the entire

build system setup is required in order to include some utilities specific to the Robostix in the kernel image.

2.10 System Aspects

A quadrotor is ultimately quite simple in terms of parts required. Many parts are widely available in the RC community and simple in design. This quadrotor, in addition to the basic MK-50 frame, utilized inexpensive BLDC motors with their corresponding ESCs and appropriate propellers, an IMU, and a controller. Figure 2.6 shows the complete component breakdown with applicable communication lines. A web camera and IR camera are also indicated, which will be discussed in greater detail in Chapter 4 and Chapter 6.

2.10.1 Cost

In keeping with the system requirements, the parts that make up the quadrotor are inexpensive. Per the Appendix, a complete breakdown of component description, part number, and cost, yields a total net cost of $1,000 dollars. As a prototype system, this is a very low price, as components bought in single quantity and pre-made for ready use are always much more expensive. Features that are not used, such as the extensive computer capabilities of the Gumstix, could be trimmed, and cost reduction actions could be taken by switching to cheaper and readily available surface mount components for the voltage regulators and miscellaneous circuit components. For an example of the cost reductions possible, the two sensor boards bought through Sparkfun were a total of $80, while an examination of readily available electronic component suppliers, such as Digikey, yields a cost of $18 for the sensor chips themselves. A minimal cost for the additional materials, plus the PCB needed which would include all the other components, would not contribute a significant amount.

2.10.2 Weight and Center of Gravity

Total system weight is 1345 g, but the attitude and navigation systems only have a weight of 1215 g by discounting the components required for the extra capabilities described

in Chapter 6. The conservative estimated max flying weight is 2100 g, yielding a non-essential payload capacity of over 800 g at high altitude. Center of gravity (CG) was measured by weighing each component and measuring as accurately as possible the distances to the center of gravity of each component. The measurements are shown in Equation (2.14), measured in cm, relative to the very center of the frame, with the z-axis center between the rods. The rotor plane sits 1.7 cm above the vertical center of the rods. The weight of the USB wire that hangs from the quadrotor to the ground is included, and so brings the CG much lower.

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} -0.182 \\ -0.265 \\ -1.873 \end{bmatrix} \tag{2.14}$$

2.10.3 Power Routing and Consumption

A standard Lithium-Polymer (LiPo) battery is used due to its high current discharge rates and good energy per unit weight. A MS Pro-Lite model battery manufactured by Thunderpower RC, Inc. is used in the 3-cell, 11.1 V version of 2600 mAh capacity [46]. This battery has a continuous discharge rate of 20 C, giving it the capability to discharge a continuous current of 52 A, which is much higher than needed. It weighs 180 g.

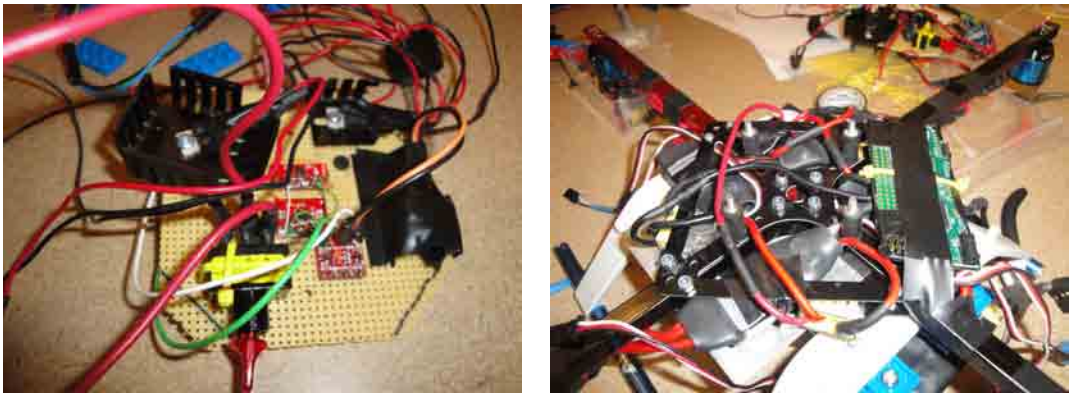
Component Power

Only the ESCs, and consequently the motors, run off of 12 volts. The Robostix, which powers the Gumstix (through an internal 3.3 V converter) and Netpro boards, runs off 5 V, which is powered by an LM1084 voltage regulator [47]. There is a second Robostix that is used to control a small servo, which is powered through a second LM1084 regulator. The Zigbee and IR camera board are also fed 5 V from this regulator. The sonar is powered via a 5 V output directly from the Robostix. The accelerometer and gyros require 3.3 V, and so an LD1117 regulator is used, taking its 5 V input from the Robostix [48]. Another LD1117 is used to power the IR camera which is dedicated to that board for installation

ease. All regulators have the standard capacitors outlined in their individual specifications for filtering the power supplies.

The voltage regulator powering the Robostix and flow down sensors was found to be operating near its capacity. Although based on the calculated current draw and the temperature of the regulator, there should not be issues, in practice it would occasionally cause the Gumstix to reset. With an extra large heat sink, plus a smaller one attached, the reset problem does not recur, although the heat sink is still quite warm to the touch.

After long periods of constantly powered use, the Robostix I^2C would occasionally hang by not releasing its hold on the active low switch. This was determined to be caused by poor power inputs due to heating, and adding yet another heat sink as well as heat flux, mostly prevented this effect. This was also caused by failing regulators onboard the Robostix, which prevented the processor from operating correctly. It is apparent from this, that switching to a surface mount more efficient switching regulator would save many problems and a significant amount of space and weight. Figure 2.11(a) shows the power switch and the first two regulators with respective heat sinks; Figure 2.11(b) shows the wiring method for the main power and ESCs, using pushpin wire connectors.



(a) Main component board with IMU and power components.

(b) Wiring setup for main power and ESCs.

Fig. 2.11: Component power wiring and setup.

Power Consumption

The total system utilizes 0.9 amps when no motors are running. A breakdown of the major components and the consumption of the motors is shown in Table 2.1. At hover thrust, estimated steady-state power consumption is 11 amps. This does not account for the constant thrust changes occurring for attitude stabilization, which would yield more frequent peak currents.

2.10.4 Running Time

Based on both the estimated power consumption during flight as well as empirical measurements of lengths of flights for a single battery, estimated flying time for a 2600mAh battery is 5 minutes, for only roughly 50% discharge of the battery for safety.

2.11 Issues

Building a physical system often includes many iterations of architecture setups as components get changed, become broken or pieces are moved around to optimize the dynamics of the system. Aside from the hand-made frame, the quadrotor went through several significant changes. Motor and propellor orientations were changed frequently and using different attachments as the effects of CG were examined. Similarly, the battery and processor components were moved. Sensor placement and mounting were changed several times, with the entire IMU being swapped out, as discussed in Chapter 3. In general, it took many flights and opportunities for data analysis in order to understand what effect each change

Table 2.1: Power consumption.

Component	Current Consumption
Gumstix w/ Robostix, IMU and Sonar	0.64 A
Four ESCs	0.09 A
Zigbee	0.05 A
Observed Total	0.9 A

RPM	Current Drawn	Peak Current
4250	2.5 A (One Motor)	3 A
4250	6.5 A (All Motors)	10.55 A
6500	7 A (One Motor)	10.5 A

had and what needed further optimization. Figure 2.12 shows the frame modified with a structural bracing on the roll axis as well as a cage-based battery mount.

2.12 Comparison to Commercial Quadrotors

By cost, the only quadrotors that compare are the AR Drone Parrot and the hobbyist-based Arducopter [12, 49]. Both are much lighter, on the order of a few hundred grams and offer very little payload capacity. The Mikrokopter quadrotor is about twice the price, and also weighs less, with reduced payload and sensing capabilities; similarly with the popular research platform, the Ascending Technologies Hummingbird [6, 7]. Quadrotors in the roughly similar weight range are the Ascending Technologies Pelican, the Microdrone MD4-200, and the Quansar QBall, all of which have a cost of well over a magnitude in difference from the quadrotor presented here [8, 9, 50].

2.13 Related Work

The development of quadrotors has been very active in the research community for the past decade. Several different kinds of quadrotors have been used, with varying degrees of success. Quite a few groups started with or modified one of the first commercially successful RC quadrotors, the Draganfly, but many later moved on to more robust and customized platforms [10,20,51–54]. Some of the custom quadrotors built or used by individual research

Fig. 2.12: Quadrotor frame with structural bracing and battery mount.

groups include the STARMAC, X-4, the Ascending Technologies Hummingbird and the OS4 [17,31,38,55]. The successful STARMAC quadrotor is of similar size, weight, and system to the quadrotor presented here. The Hummingbird is much lighter, roughly 300g, and is much more streamlined in construction. The X-4 flyer is quite different, in that it weighs over 4kg, designed with an eye towards robust construction and high power considerations [37].

Although specific numbers are not available, in general the construction of the quadrotor here is roughly similar to others in weight, size, sensing, and processing, with the significant difference of utilizing very low cost and widely available or easily customizable components.

2.14 Chapter Summary

This chapter presented a simplified mathematical system model and the full build up and specifics, both mechanical and electrical, of a customized quadrotor platform. Analyses of specific components and changes required to make flight possible were presented. Finally, the quadrotor presented was shown to utilize low cost components when compared with other available quadrotors, while still offering similar, if not improved, capabilities. The final quadrotor product is shown in Figure 2.13.

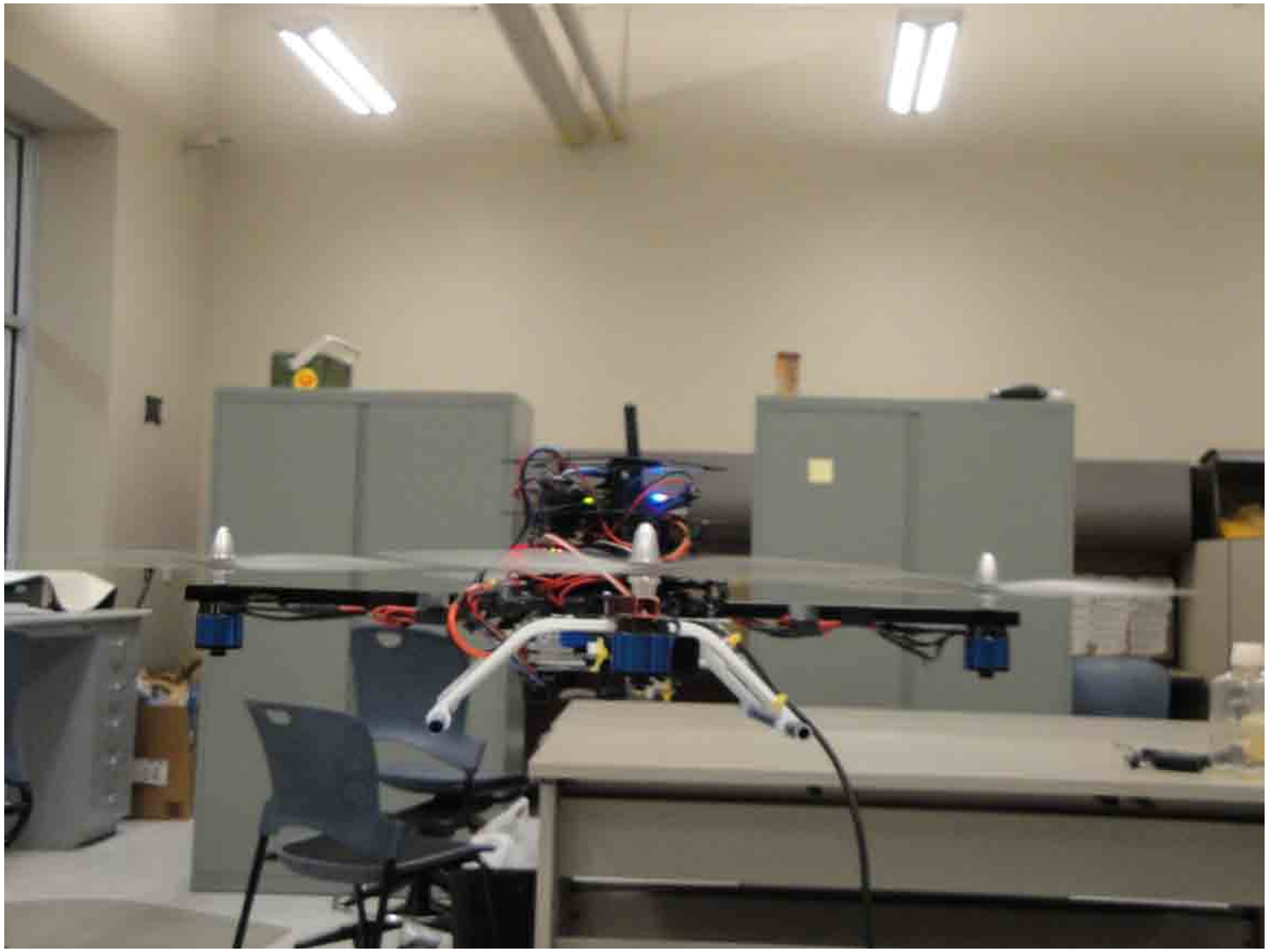


Fig. 2.13: The quadrotor.

Chapter 3

Attitude Control

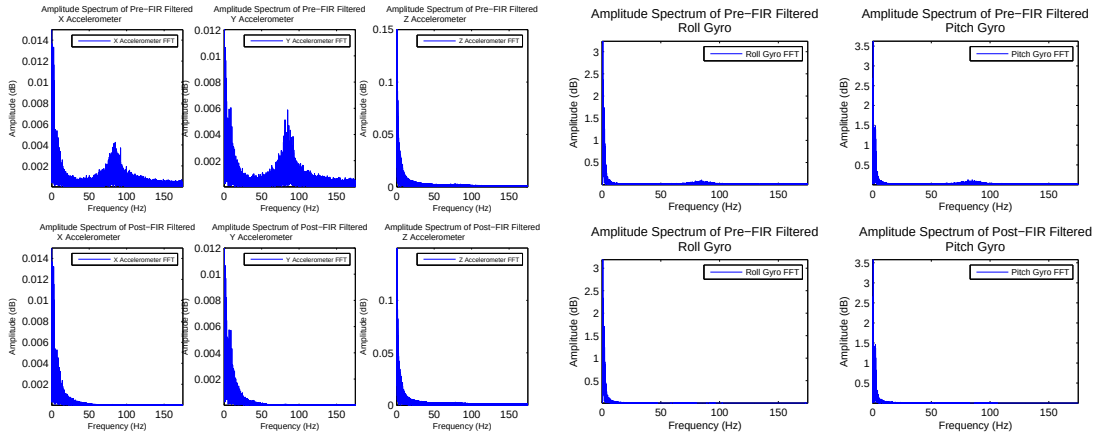
As a dynamically unstable system, the quadrotor is unusable as an UAV without minimal attitude control for maintaining level flight, so that some type of directional input can be given for navigation. This chapter covers details of the types of sensor conditioning and attitude estimation in use, the software attitude control algorithm, analysis of empirical testing, and references to related work.

3.1 Sensor Conditioning

In ideal systems, noise effects are disregarded, and in many cases so are sensing errors and quantization effects. In physical systems, however, attempting to control the system while ignoring these aspects will lead to debilitating results, often preventing any sort of understanding of system response. This section indicates the extensive filtering and sensor fusion in use on the system, as well as some of the issues and difficulties involved, and analyzes the problems faced with a previous sensor for which these noise and error effects were extensive and ultimately unworkable.

3.1.1 Noise Sources

The primary noise source for the quadrotor comes from the rotational effects of the motors and props. This noise can easily couple through the frame to the sensors [52]. As indicated in Figures 3.1(a) and 3.1(b) for the magnitude vs frequency spectrum, there is significant noise levels at around 60-80 Hz. During flight, operating speeds of the motor are in the 4500-5500 RPM range, which when converted, falls directly within the range where there are high levels of signal frequencies. The fact that the noise is not as apparent on the z-axis accelerometer further indicates that the noise is from the motor, which will be



(a) Accelerometers. (b) Gyros.

Fig. 3.1: Magnitude vs frequency spectrum of accelerometers and gyros during flight.

primarily rotational noise in the horizontal plane (back and forth) and not so much rapid up and down movement. In addition, the frequency of the noise changes depending on the speed of the motor - the noise is only 40-55 Hz at very low thrusts, which fits with the estimated 2700 RPM. This noise due to vibration is already reduced mechanically using rubber damper standoffs, which sit between the frame and the board with the sensors. Further mechanical damping was attempted as discussed in Section 3.3, but feasible solutions were not discovered.

3.1.2 Accelerometer

The BMA180 has on-chip digital filtering capability which is utilized to perform finite impulse reponse (FIR) filtering at 1 KHz, with a cutoff frequency setting of 10 Hz. This setting was determined based on comparing sensor data when: stationary with and without motors running, moved by hand with and without motors, and in flight. The cutoff is based on removing all data that is not actually due to a response from physical movement while not eliminating any data that is due to movement. Since the quadrotor is ultimately limited in high frequency response characteristics due to physical system effects, it exhibits a rather low frequency movement, so high frequency data can be eliminated as noise.

IIR vs FIR Filtering

FIR filtering is used instead of infinite impulse response (IIR) filtering due to the advantage of not having to worry about stability concerns. The increased delay and computation time are considered negligible due to such short length filters and the performance is nearly the same.

FIR Filter Details

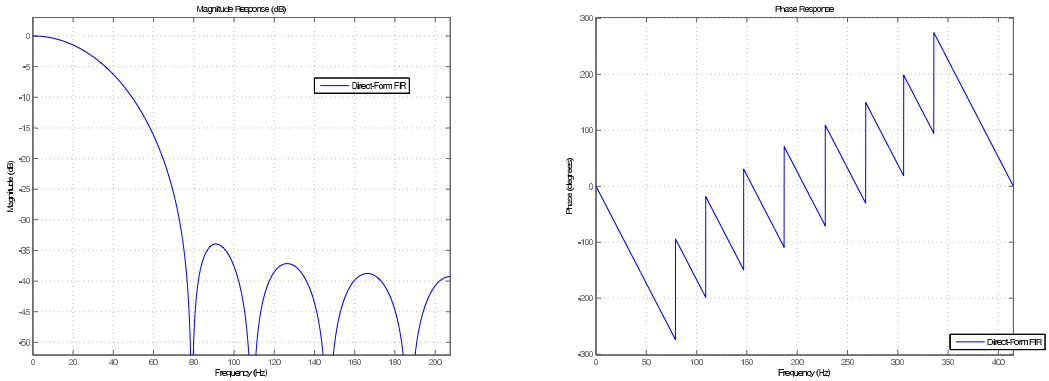
Figure 3.1(a) indicates raw hover frequency spectrum flight data from the chip, both after just on-chip FIR filtering as well as after final software filtering. Even after mechanical damping and on-chip FIR filtering, there is still extensive high frequency noise. If no onboard filter is used, signal variations are up to 30 times greater than the raw data shown for a 10 Hz cutoff. Due to this strong high frequency noise, further FIR filtering is performed in software. Since the attitude control loop only runs at roughly 400 Hz, it is important to keep the length of the filter as low as possible in order to reduce delay effects [38]. As such, an 8th order filter is used, with

$$\begin{aligned} y^a[n] = a[n]\cdot \quad & (b_0^a + b_1^a z^{-1} + b_2^a z^{-2} + b_3^a z^{-3} + b_4^a z^{-4} \\ & + b_5^a z^{-5} + b_6^a z^{-6} + b_7^a z^{-7} + b_8^a z^{-8}), \end{aligned} \tag{3.1}$$

where $y^a[n]$ is the filtered accelerometer, $a[n]$ is the input measurement from the accelerometer chip, and b_i^a are the filter coefficients, with,

$$\begin{aligned} b_0^a \ldots b_8^a \quad = \quad & [0.02906, 0.07412, 0.12850, 0.17347, 0.19094, \\ & 0.17347, 0.12850, 0.07412, 0.02906]. \end{aligned} \tag{3.2}$$

The filter was designed using a weighted least-squares method. The magnitude and phase response of the filter are shown below in Figures 3.2(a) and 3.2(b). The filter valleys are designed so that they center around the frequency of the noise from the motors at flight speeds. Figure 3.1(a) indicates the reduction in magnitudes of the noise after running



(a) Magnitude.

(b) Phase.

Fig. 3.2: Accelerometer FIR filter plots.

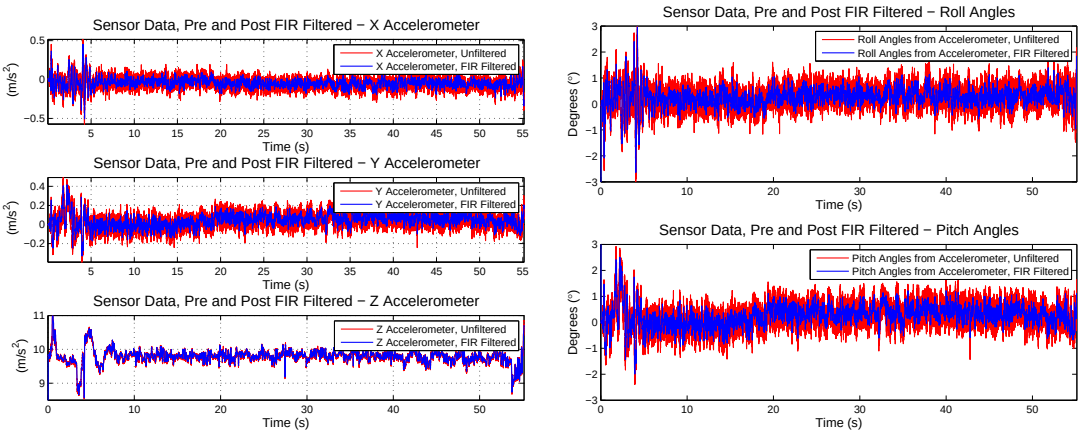
through the FIR software filter.

Care has to be taken in the filter design to ensure that it will work accurately for all three axes. The z-axis acceleration is different from the other two in that it is constantly measuring gravity. If a filter is chosen that starts out above or below 0 decibels (DB), then the filtered z-axis accelerations will be offset. When using the original sensor, placing capacitors on the z-axis also caused this effect.

Figures 3.3(a) and 3.3(b) indicate how the accelerometer data, and angles calculated purely from them per Section 3.1.3, perform during a typical hover flight before and after filtering.

Sensor Settings and Specifications

The BMA180 comes with a few sensor setting options. One of these is a noise mode setting, which will trade off power for resolution. For dealing with the extensive noise indicated above, the sensor is set to ultra low noise mode. The sensor is capable of offset calibration on-chip, however its use was not intuitive, and since the software was already set up to perform calibration in the main routine, on-chip calibration was considered unnecessary. The sensor also has a variable acceleration range setting for $\pm 1g$ to $\pm 16g$. Given the stability and physical requirements of the quadrotor system and with examination of



(a) Accelerometer measurements. (b) Angles calculated from accelerometers.

Fig. 3.3: Accelerometer hover measurements - unfiltered vs filtered.

several data from flights, a range of $\pm 2g$ was chosen, allowing better resolution. The 14-bit ADC on-board the sensor is used, allowing for a net resolution of 0.25 mg/least significant bit (LSB) with the range chosen.

Usage

The sensor was double-checked for accuracy against a digital inclinometer, for the reasons indicated in Section 3.3 below. Although the inclinometer was not calibrated and its accuracy was unknown, it did verify that the angles estimated using the accelerometer were reasonable, except possibly for very large angles (greater than 45 degrees).

Utilizing the sensor required writing specific register values according to the desired settings indicated above. Since it takes time to confirm registers were written correctly, sleep statements are used to give time for the registers to return with the newly-written value after being changed. The sensor is reset initially due to poor power startup from the Robostix, which hovers around 3 V initially and possibly causing initialization problems with the sensor. Wait times after reset allow the sensor to come up completely before writing, and then written values are read back to make sure that they have been written properly as a safety feature. Wait times had to be tweaked a bit in order to allow all registers to be written properly.

3.1.3 Calculating Angles Using Accelerometers

The attitude of the quadrotor can be obtained using the accelerometers, based upon taking the relationship between the horizontal and vertical measured accelerations so that

$$\phi = \arctan\left(\frac{\ddot{z}}{\ddot{y}}\right) + \frac{\pi}{2}, \tag{3.3}$$

$$\theta = -\arctan\left(\frac{\ddot{z}}{\ddot{x}}\right) - \frac{\pi}{2}, \tag{3.4}$$

where the angles are measured in radians and ϕ and θ are the roll and pitch angles, respectively. The accelerations, measured in m/s^2 are $\ddot{x}$, $\ddot{y}$, and $\ddot{z}$ for the x, y, and z measured accelerations. Note that in practice, the atan2 function is used for robustness. Angles exceeding π and below $-\pi$ must be explicitly taken care of.

3.1.4 Gyroscope

The ITG-3200 similarly has digital on-chip filtering capability and is performed at 1 KHz. The on-chip cutoff frequency is set to 25 Hz due to the quadrotor angular rate dynamics being of a higher frequency than for acceleration measurements. Due to the excessive noise, a similar length FIR software filter as for the accelerometer is used

$$\begin{aligned} y^g[n] = g[n]\cdot \quad & (b_0^g + b_1^g z^{-1} + b_2^g z^{-2} + b_3^g z^{-3} + b_4^g z^{-4} \\ & + b_5^g z^{-5} + b_6^g z^{-6} + b_7^g z^{-7} + b_8^g z^{-8}), \end{aligned} \tag{3.5}$$

where $y^g[n]$ is the filtered gyro output, $g[n]$ is the input measurement from the gyro chip, and b_i^g are the filter coefficients, with

$$\begin{aligned} b_0^g \ldots b_8^g = \; & [0.05315, 0.08955, 0.12388, 0.14842, 0.15732, \\ & 0.14842, 0.12388, 0.08955, 0.05315], \end{aligned} \tag{3.6}$$

designed using a weighted least-squares method and a cutoff of 25 Hz. Similar high cut-off frequencies for the gyro and accelerometers have been used before without issue [20]. Net

resolution of the gyro is 0.696 ° /sec/LSB. Figure 3.4 shows how the gyro measurements look before and after filtering during a typical hover flight.

3.1.5 Sonar Sensor for Height

The sonar is run through a length three median filter in order to account for extraneous values that sometimes occur with sonar readings, using

$$z^{alt} \quad = \quad Median[(z^{alt}_{raw}(t-2)) \;\; (z^{alt}_{raw}(t-1)) \;\; z^{alt}_{raw}(t)], \tag{3.7}$$

where $Median$ sorts the three values in numerical order and returns the middle measured value. The anomalies being removed typically occur several times during a two-minute flight. The sonar is sampled at 20 Hz so that no values are lost, no extra loop time is taken to sample duplicate values, and repetitions of the same sampled value are not acquired.

3.1.6 Sensor Coordinate Frame

The axes definitions of the prototyped quadrotor differ from traditional aero convention discussed in the modeling of the quadrotor in Chapter 2 due to the mounting of the sensors. Although this could be adjusted in software, development of the control system was completed before adapting the proper convention, and so for consistency, all implementation in this thesis uses the axes definitions shown in Figure 3.5. The rotors, numbered $i \in 1, 2, 3, 4$ per Equation (3.27), are mounted outboard on the $-yB$, xB, yB, and $-xB$ axes, respectively.

3.1.7 Attitude Sensor Issues

Individually by themselves, accelerometers and gyroscopes cannot obtain level flight. The accelerometers are too noisy and prone to small errors to accurately estimate angular rates by differentiating the estimated angles [56]. Angular rates are needed for stabilization since the quadrotor is an underdamped system; they are needed due to the important measurement of how fast the quadrotor is rotating so it can be brought back from on its

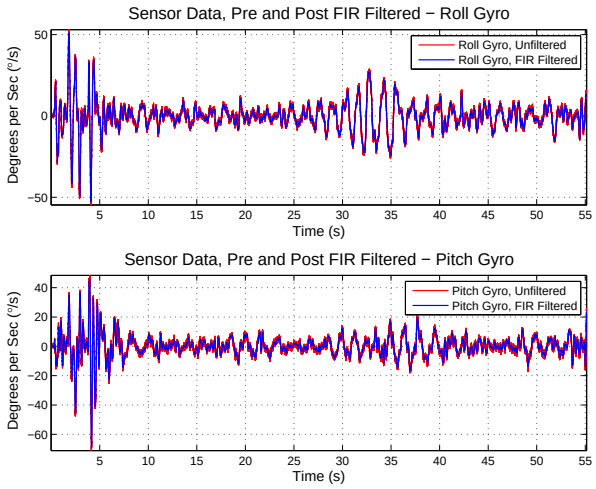


Fig. 3.4: Gyro hover measurements - unfiltered vs filtered.

way to large angles rather than just depending on the immediate angle from which the quadrotor is sitting. Gyroscopes by themselves are also limited, as they cannot detect a constant rotational (angular) offset, which would cause the quadrotor to fly off in one direction, known as translational drifting. In addition, gyroscopes are prone to bias effects, which build up over time and cause inaccuracies in the angular rate estimate if not accounted for.

Software Sensor Calibration

Due to possible offsets from ground truth that can occur in the sensor readings on initialization or between flights, as well as gyro bias buildup, a software calibration process is run, simply taking 500 readings and averaging them to determine the offset from the known truth (zero degree roll and pitch angles and zero angular rates). This offset calculation is then used throughout the flight.

Gyro Bias

The calibration is one step towards removing unwanted bias effects from the gyros, by determining the current bias from zero. Since the bias is changing over time, however, an estimate of the bias is needed during flight. This action is performed by fusing the

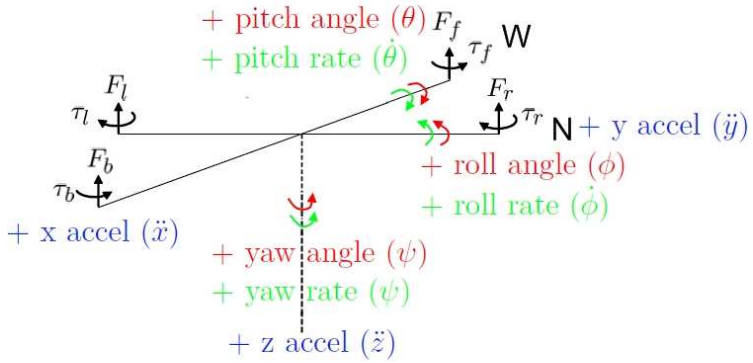


Fig. 3.5: Attitude system axes definition. Note that this differs from the standard aero convention discussed in Section 2.1 due to application specifics, so readers will be required to associate the system aspects in this thesis using this frame setup.

accelerometer and gyro measurements. An alternative to this is treating the bias as a very low frequency effect and filtering it out [20]. This approach does not yield very good results in practice due to noise, true signal effects from actual movement, and extensive filtering requirements.

3.2 Sensor Fusion

Estimating the true attitude is not an easy task when noise is present in the system. Accelerometer data with low cost sensors are noisy and have poor vibration resistance, as can be seen from the hover figures of just the accelerometers, Figure 3.3(a), and the angles calculated from them, Figure 3.3(b). Using purely this data to estimate the attitude of the quadrotor yields an inaccurate estimation of the state of the system. Utilizing the much less noisy gyro measurements is also problematic, as there are two measurement drifts incurred when calculating angles from the gyros: the gyros' inherent characteristics cause a very slow changing drift; and numerically integrating will additionally yield to a net large accumulation of many small errors.

Utilized together, the gyros and the accelerometer can diminish the problems with each of them. The gyros can be integrated and combined with the accelerometers to get a better

angle estimate, and the accelerometers, which do not have bias issues, can be used to adjust for the drift in the gyro angle calculations. Typically, a standard or modified Kalman Filter is used for this requirement [20, 57, 58].

The Kalman filter did not give the most desirable results when applied to the quadrotor outlined here. This is likely due to the nonlinear system dynamics and measurement data and the non-Gaussian noise in the low-cost sensors, neither of which are effectively taken care of with the Kalman filter, since it assumes linear systems with Gaussian noise. Thus, a different fusing filter, the Nonlinear Passive Complementary Filter is used [59]. Classical or linear complementary filters have also been used for quadrotor angle and gyro bias estimation, and complementary filters themselves have been around for some time [58,60,61]. The linear complementary filter was not implemented due to the problems seen with the Kalman filter. The linear complementary filter exploits the differing spectral characteristics of the measurement sources and fuses these using a fixed gain. However, the filter relies on the measurement system being able to be accurately linearized.

3.2.1 Nonlinear Complementary Filter

The nonlinear complementary filter provides a superior method to fuse the accelerometer and gyro measurements, exploiting the known properties of the underlying system. A block diagram of the nonlinear filter is shown in Figure 3.6 [59]. Notice the structural similarity with the linear complementary filter, including the preset gain, and hence a similar name is used.

3.2.2 Description of Structure and Operation

The rotation of the quadrotor can be described using a rotation matrix, and all possible orientations of the vehicle live in the space of $\mathbb{R}^{3x3}$ rotational matrices known as the Special Orthogonal Group SO(3). With the rotational kinematics of the quadrotor given as

$$\dot{R} = R \cdot sk(\Omega) = sk(R\Omega)R, \tag{3.8}$$

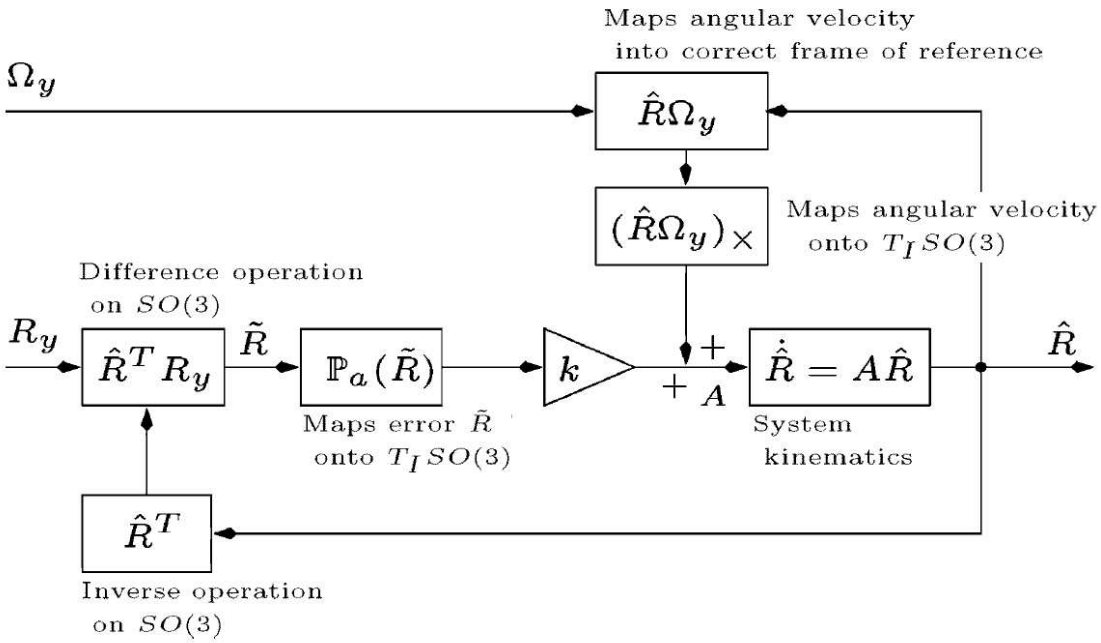


Fig. 3.6: Block diagram of the nonlinear passive complementary filter.

per Equation (2.6), where R is the rotation matrix formed from the orientation of the quadrotor in the inertial frame as described in Equation (2.3). $sk()$ denotes the creation of a skew-symmetric matrix generated using the vector inside the parenthesis and is equivalent to the Figure 3.6 block diagram notation of $()_X$. $sk(\Omega)$ is then the skew-symmetric matrix formed from the angular velocity, Ω, in the body frame. The pre-multiplication of Ω by the rotation matrix R, is to ensure that the velocity is in the correct frame of reference, since the measured angular velocity lies in the body-fixed frame, while the same frame is required for combination of the two measurements, as the accelerometers are measured in the inertial frame. This filter exploits the fact that the system can be described using DCMs, and so is specifically designed on the SO(3) group. This makes it explicitly useful for airborne vehicles.

An observer is proposed for obtaining the estimated orientation of the quadrotor, which is posed as a kinematic system to give an estimate of the attitude of the system, $\hat{R}$, with

$$\dot{\hat{R}} = sk(R\Omega + k_{\text{NLF}}\hat{R}\omega)\hat{R}, \qquad k_{\text{NLF}} > 0, \tag{3.9}$$

where, k_{NLF} is a non-zero positive fixed gain, with the minimum value determined using the Lyapunov Argument, although the calculated minimum value for stability is much lower than the practical value used. Then

$$\omega = vex(\mathbb{P}_a(\tilde{R})), \qquad \tilde{R} = \hat{R}^T R, \tag{3.10}$$

is the correction term and is a function of the error, $\tilde{R}$ given by $\hat{R}^T R$ where the $\hat{R}^T$ operation is an inverse operation on SO(3) and is equivalent to a "-" operation for a linear complementary filter. The $\hat{R}^T R$ operation is then equivalent to generating the error term $y - \hat{x}$. vex returns the generating vector of a given skew-symmetric matrix. ω then maps this error into the tangent space of SO(3) for combining with the attitude from the gyros, Ω. ω can be seen as a nonlinear approximation of the error between R and $\hat{R}$.

The aim of the observer, Equation (3.9), is to drive the error term $\hat{R}^T R$ to the identity matrix, I_{3x3}, and thereby estimate the correct value of $\hat{R}$. In this implementation, this is the error between the estimated attitude R_y given by the rotation matrix formed from angles obtained from the accelerometers, calculated according to Equation (3.3), and the attitude estimated by the filter $\hat{R}$, with $\Omega_y = \Omega + b$ being the vector derived from the angular velocity data given by the gyroscopes, where b is the bias on the gyroscopes. Through Lyapunov analysis, the method of combining the error using R_y and $\hat{R}$ and the attitude from the gyros, $\hat{R}\Omega_y$, is found to be the mapping of these two into the tangent space of SO(3), which is the space of skew-symmetric matrices.

The two operations $\mathbb{P}_a(\tilde{R})$ and $sk(\hat{R}\Omega_y)$ are maps from the error space and velocity space into the tangent space of SO(3), which are specifically required by the properties of SO(3). The two skew-symmetric matrices thus generated are added using the positive gain, k_{NLF}. Finally, taking the exponential map for the system kinematics of Equation (3.8), where R is replaced with $\hat{R}$, brings the output back to the Lie-group SO(3). The kinematic model, $\dot{\hat{R}} = A\hat{R}$, is the Lie-group equivalent of a first order integrator.

The final form of the mathematic model for the filter, along with the bias estimator, is then given by

$$\dot{\hat{R}} = \hat{R}A \qquad A = sk(\Omega_y - \hat{b} + k_{\text{NLF}}\omega), \tag{3.11}$$

$$\dot{\hat{b}} = -k_b \cdot \omega, \qquad k_b > 0, \tag{3.12}$$

where $\hat{b}$ is the estimated bias and k_b is a bias gain which needs to be greater than 0. The gains, k_{NLF} and k_b, are set to 1 and 0.3, determined after extensive examination of the sensor data in flight and ends up being the same experimental values previously used [59].

3.2.3 Motivation

Like most filters, this filter allows a tunable emphasis on gyroscope angle estimates, $R\Omega_y$, which have been found to be a better estimate of the angular rotation of the quadrotor than those obtained from the accelerometers, R_y, due to the noise and inaccuracies in acceleration measurements. As a corollary to this, the second feedback loop in the filter makes use of the filtered attitude, $\hat{R}$, in the predictive angular velocity term, $\hat{R}\Omega_y$, giving the advantage of avoiding corrupting the predictive angular velocity term with the noise and errors in the reconstructed pose from the accelerometers, R_y.

Also, a crucial factor of this filter for use on an embedded system is that a closed form solution can be obtained, since solutions to matrix exponentials, for solving the first order differential equation, $\dot{\hat{R}} = A\hat{R}$, are computationally expensive. A closed-form solution can be obtained because the matrix exponential being calculated, A, is a skew-symmetric matrix, and so there is an analytical solution, Rodrigue's Formula, which can be used in place of matrix exponentiation for this case.

3.2.4 Gyro Bias

As noted in Equation (3.11) and similar to the function of the Kalman filter implementation, this filter calculates the gyro biases, then uses the bias adjusted gyros to estimate the angles through integration, and then fuses them with the accelerometer data according

to the filter gains. The yaw gyro bias is also estimated before finding the angles, but no other fusion is possible since no sensor, such as a magnetometer, is available for fusion.

Actual gyro bias has been measured to be very small for the sensors used. In 3 minutes of data taken while flat, a drift of only 0.5°/sec was measured, and this can easily be corrected with the nonlinear filter. An example of the gyro bias measurement and how it affects the bias-adjusted rate compared to the raw measurement in flight is shown in Figure 3.7.

3.2.5 Results

The overall performance of the nonlinear complementary filter can be shown by examining the output of the attitude estimation compared to the FIR filtered raw data. Figure 3.8(a) shows the roll angles estimated by the filter compared to those derived purely from the accelerometers; Figure 3.8(b) shows the same for the pitch angles.

3.2.6 Challenges in Implementation

Using this filter with the components available on the platform is problematic due to the restrictions of the processor. The lack of a FPU and limited memory prohibited the use

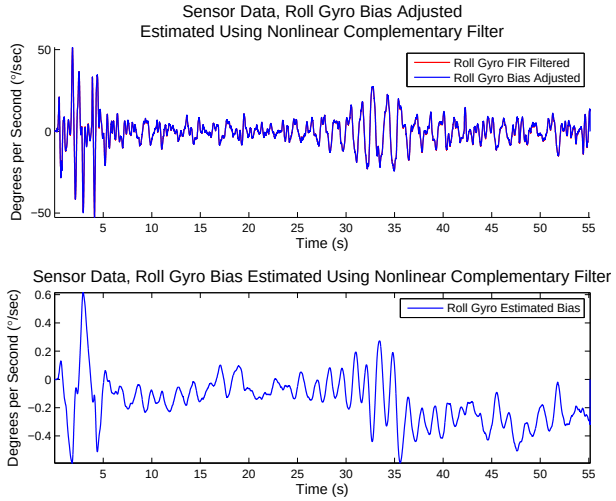


Fig. 3.7: Gyro bias estimated by the filter, and bias adjusted gyro compared to unbiased data.

of standard matrix libraries with efficient routines for floating point matrix computations. Specific needed matrix routines had to be implemented.

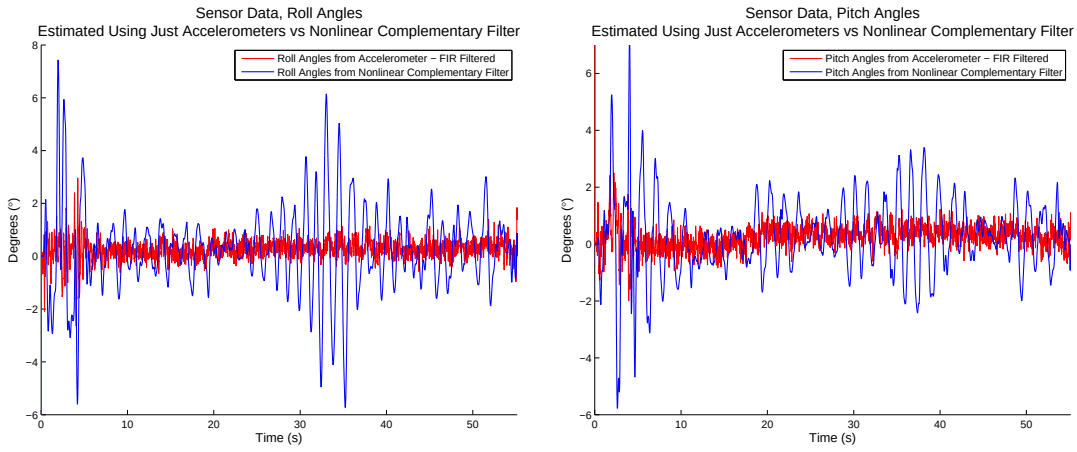
A concern while implementing the filter was the loss of orthonormality of the DCM due to approximation factors, as this would require specifically normalizing the matrix. However, in practice, this was not found to be a problem.

3.2.7 Kalman Filter vs Nonlinear Complementary Filter

Figure 3.9 below gives a comparison between the Kalman filter and the nonlinear complementary filter. The advantage of the nonlinear complementary filter in obtaining accurate and smooth attitude data is readily apparent. Since this data is from an actual hover flight, it is obvious that the estimated attitude from the Kalman filter does not represent the true orientation of the quadrotor.

3.3 Original Sensor

Initially a different IMU setup was used. The IMU was a single package 2-axis gyro and 3-axis accelerometer from Sparkfun, containing an ITG500 gyro chip and ADXL335 accelerometer chip, and is shown in Figure 3.10 [44, 62, 63]. Since there is a missing gyro



(a) Roll. (b) Pitch.

Fig. 3.8: Angles comparison between attitude estimator and purely from accelerometer during hover.

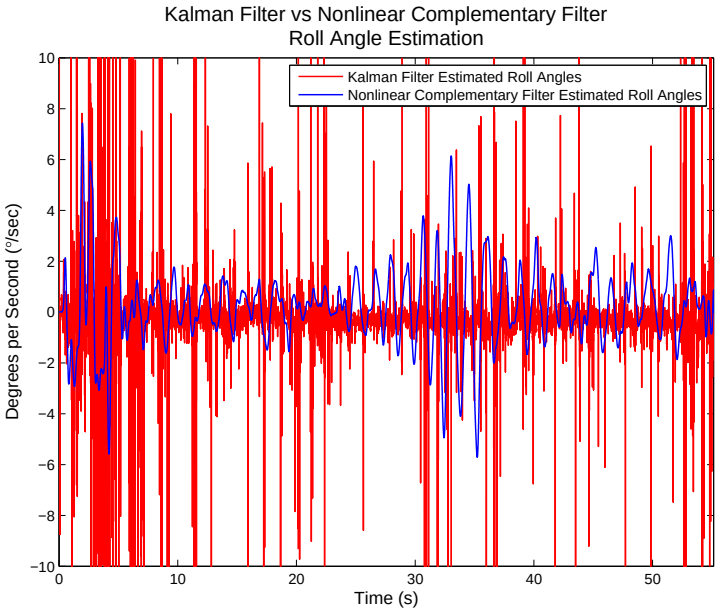


Fig. 3.9: Angles estimated: comparison between Kalman filter and nonlinear filter.

measurement around the z-axis for yaw, a second IMU of the same type was used, mounted at a vertical angle to measure the angular rate around the vertical axis.

3.3.1 IMU Specification

This IMU package gives only analog outputs, so its use required first going through ADC sampling on the Robostix and then transmission over I^2C to the Gumstix attitude process for stabilization control [26]. This extra routing placed extra delays in the system, plus the analog sampling prevented the implementation of a software filter that could run at a high frequency for a sharp noise cutoff since floating point filtering is too computationally intensive for the Robostix, and the Gumstix control loop only ran at 350 Hz. Thus, software filtering was only minimally effective compared to the current sensor suite. In addition to the filtering limitations, the Robostix ADC is only 10-bit, plus the IMU sensors themselves are fairly low resolution, yielding rather large error margins, even without noise considerations. With noise considered, the sensors were even worse as they seemed highly vulnerable to vibration noise. A direct comparison of the noise vulnerability can be seen between these sensors and the new ones by an examination of the z-axis accelerometer. The original

sensor shows large amounts of noise on all three accelerometer axes, while the new sensors have very little noise on the z-axis, indicating that the original sensors were picking up the relatively smaller noise coupling onto the z-axis in addition to a high sensitivity to the horizontal vibrations.

The IMU has onboard hardware filters placed as part of the Sparkfun package as well as part of the individual chips. A net 140 Hz low pass cutoff is in place for the gyros while the accelerometers have a net 50 Hz low pass cutoff in place.

Under no noise interferences, the accuracy on the sensors is fairly low. With a 10-bit ADC, the uncertainty for the rate measurement is $\pm$2.5deg/sec for the gyro rates and $\pm 0.16\ g$ for the accelerometers. When converted to angles as described in Section 3.1.3, this yields a no noise uncertainty of $\pm$0.95 degrees. Measurements from the sensor were taken with motors off to show its minimum accuracy, before and after filtering, as seen in Figures 3.11(a), 3.11(b), 3.11(c) for measurements of accelerometers, gyros, and angles estimated from just the accelerometers.

To show the problems the sensor has in the presence of noise even after filtering, measurements were taken at low speeds while flat on the ground, shown in Figures 3.12(a), 3.12(b), 3.12(c). For this case, the quadrotor was moved right, left, up and down, in that order, although it should be noted that this IMU's axes definition are different than for the new IMU outlined in Figure 3.5, however this information is not important to the situation.

Angle data was compared between the complementary filter fused angles and an off

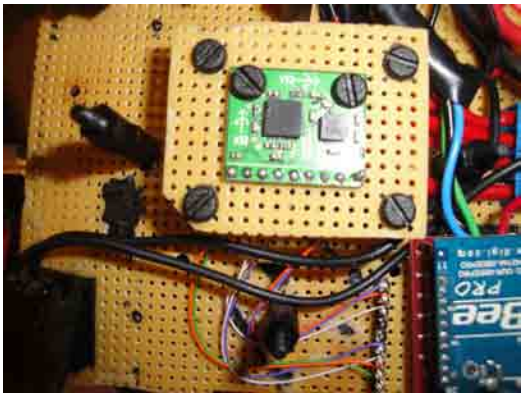
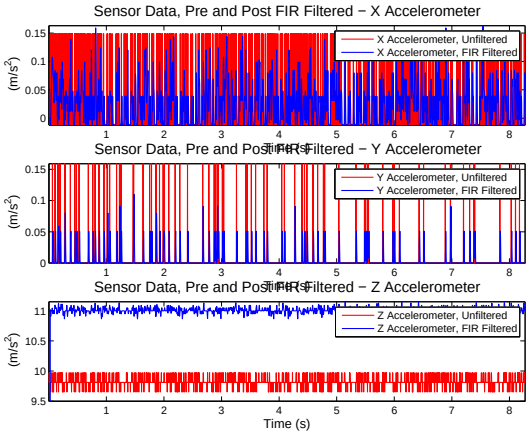
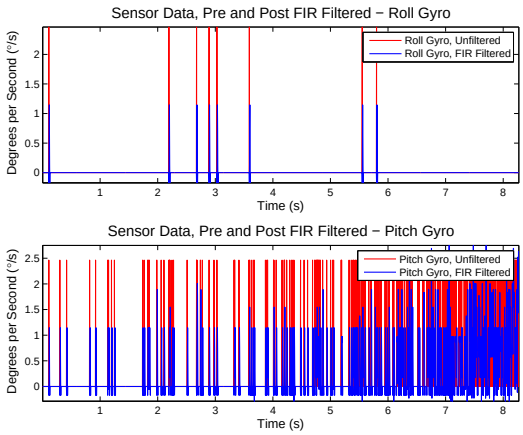


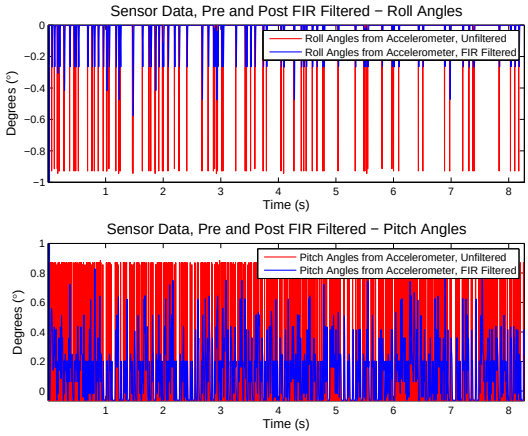
Fig. 3.10: Picture of original IMU sensor.



(a) Accelerometers.



(b) Gyros.



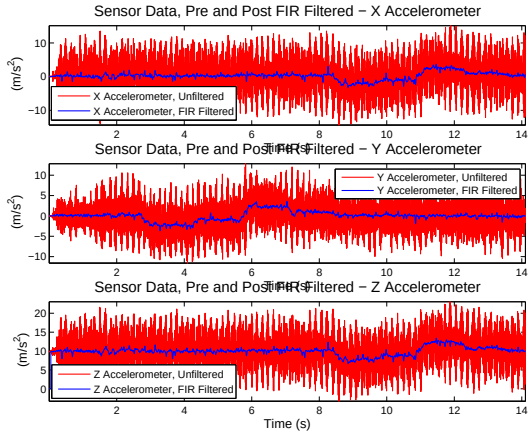
(c) Angles from accelerometers.

Fig. 3.11: Filtered vs unfiltered measured values of original sensor, flat on the ground, with motors off.

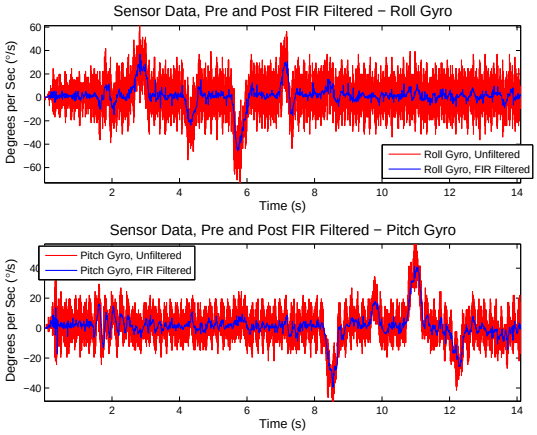
the shelf digital inclinometer. Angles measured were fairly close (within a degree) for small angle deviations, however larger angles were over or under reported by up to 2-3 degrees. This information is of limited use since the inclinometer was not calibrated, and its accuracy is unknown. Even still, used as a consistent unknown reference, checking without motors running and with them running, angles had a discrepancy in both cases of about the same.

3.3.2 Noise, Filtering, and Fusing

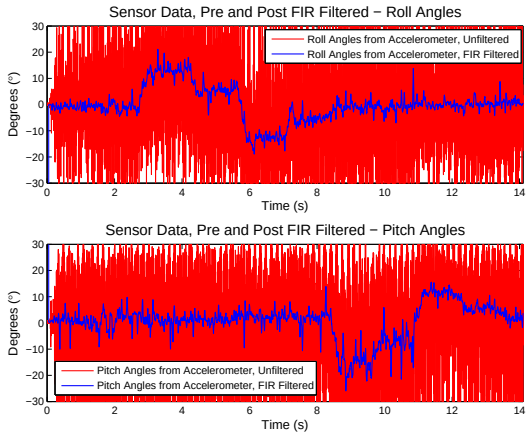
To further reduce the vibration noise, additional hardware filters were added. For the gyro, 2.2 μF capacitors were added in parallel to those provided on the chip, yielding a net



(a) Accelerometers.



(b) Gyros.

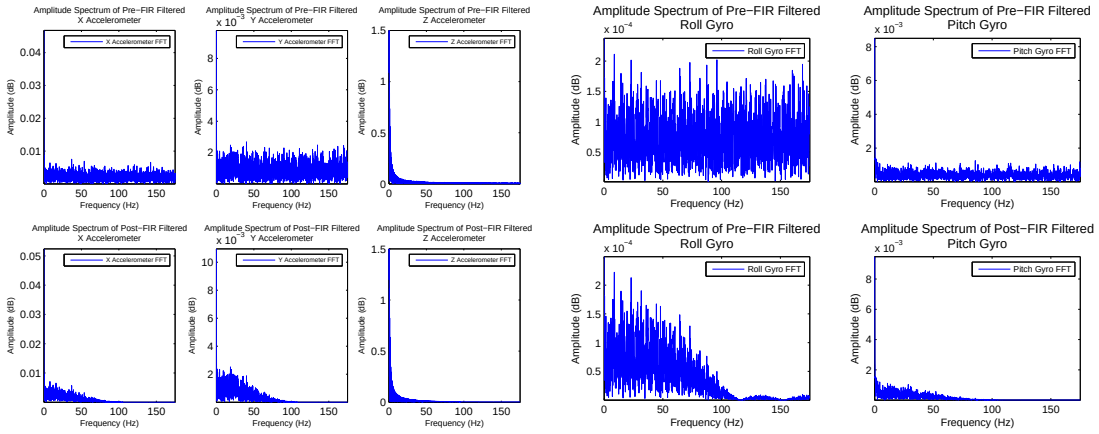


(c) Angles from accelerometers.

Fig. 3.12: Filtered vs unfiltered measured values of original sensor, right, left, up, down in hand movement, with motors on.

cutoff of 92 Hz. Aside from vibration noise, electro-magnetic interference (EMI) is always a concern with electronics, when high frequency power signals for the motor are nearby, in addition to communication noise. Such interference was noticeable from the measurements when an RC hand-held transmitter was brought nearby, but interference from other sources on the quadrotor could not be confirmed. An example of how the vibration noises, as well as the quantization and accuracy, affect the accelerometer and gyro, with motors both off and on at slow speeds when the quadrotor is completely flat on the ground are shown in Figures 3.13(a), 3.13(b) for motors off, and Figures 3.14(a), 3.14(b) for motors on.

Although fusing the accelerometers and gyros using the complementary filter yielded



(a) Accelerometers. (b) Gyros.

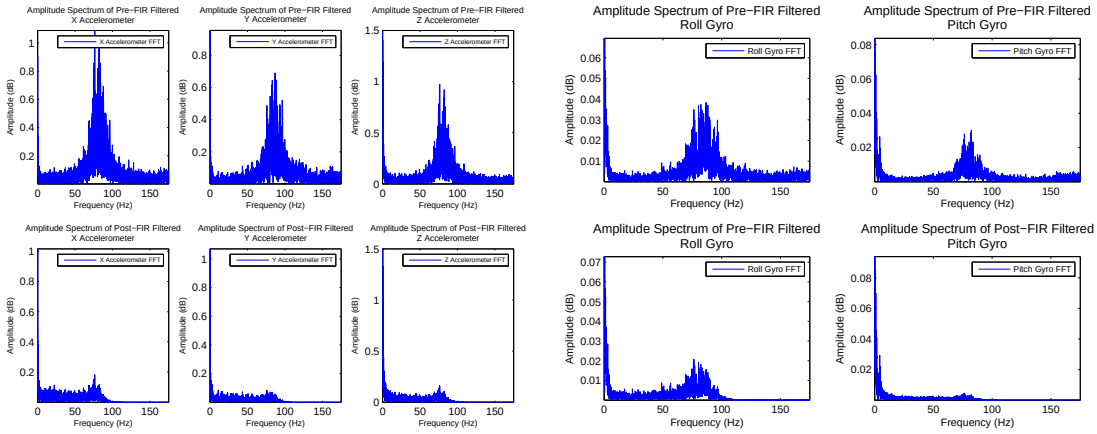
Fig. 3.13: Magnitude vs frequency spectrum of original sensors, flat on the ground, with motors off.

more accurate angles than by calculating directly from accelerometers, per the figures noted above, accuracy was still poor due to resolution and noise. Tuning the filter led to no ideal situation: low gains made the quadrotor too slow to get to the correct angles, while higher made the estimated angles quite noisy based on the weighting towards accelerometers.

Mechanical Damping

Extensive effort was placed in mechanically damping the IMU, since software filtering was not filtering out enough noise and too much delay is always a concern. Initially the IMU was simply stacked on top of some paper and foam strips and glued to a Lego piece for obtaining the 90° angle required for the yaw gyro [64]. The unit was then glued to the main power board above the frame. This proved workable, but not ideal.

Many other types of mounting styles were utilized after the first change attempt of mounting the IMU using solder and screws to a small PCB. The board was placed on top of rubber damper standoffs and then attached to the power board which was sitting on simple plastic standoffs, as shown in Figure 3.15. This proved to couple high amounts of noise, likely from the fact that the sensor board did not weigh enough for the dampers to be able to absorb the vibrations coming up from the frame. Switching the rubber dampers to the bottom of the power supply board and the plastic standoffs to underneath the sensor board



(a) Accelerometers. (b) Gyros.

Fig. 3.14: Magnitude vs frequency spectrum of original sensors, right, left, up, down in hand movement, with motors on.

still lead to excessive coupling, determined to be amplified by the standoffs. Damping the motors using grommets between the rod and the motor yielded no significant noise reduction, contrary to expectations [17].

The sensor board was mounted in different ways: on foam strips of various numbers of layers, mounted on small and medium springs, and finally mounted as directly to the power supply board as possible using Lego's to create enough surface area between the sensor board and the power board as shown in Figures 3.16(a), 3.16(b), and 3.16(c). Only the last option yielded beneficial results when root-mean squared (RMS) values were examined, although still not reduced enough ultimately for the sensor to be successfully useful. It seems that the rubber dampers used are best designed for reducing vibration when compressed to some degree, and that mounting the sensors above the board yields extra vibration coupling than if mounted directly to the board. This knowledge was used in the mounting of the new sensors.

Electrical Connection "Noise"

Not all of the extra noise was coming from vibration coupling through another device. Some problematic flight anomalies were due to vibration and movement causing uncovered pins to become shorted to something or for connectors to wiggle enough to create occasional

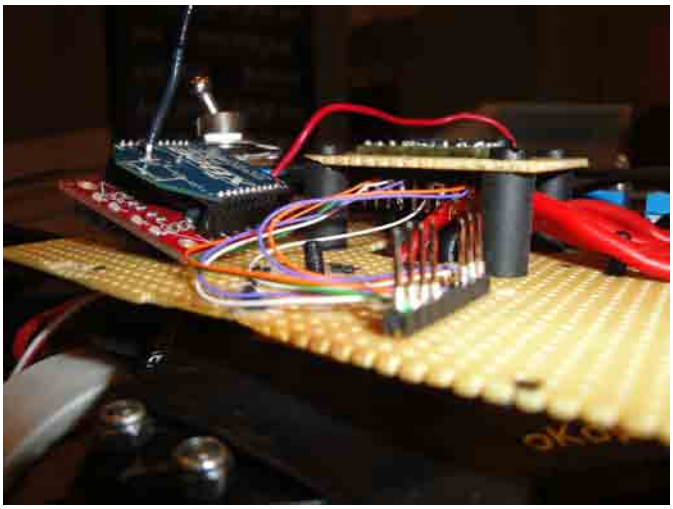


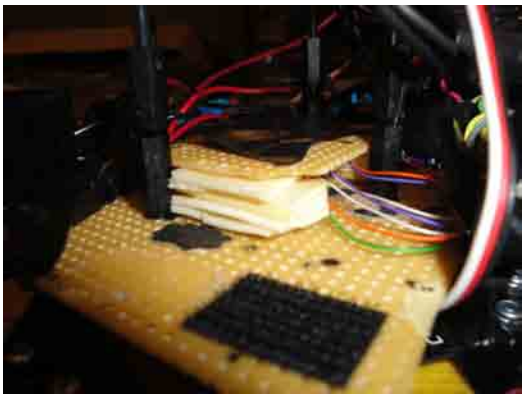
Fig. 3.15: Original IMU sensor mounted on dampers.

open signals. This was first discovered when data was being taken while the quadrotor was lying flat with the motors off. It was accidentally bumped and the graphs indicated an impossible measurement from the sensor. Detailed testing of each of the main components on the quadrotor yielded the conclusion above.

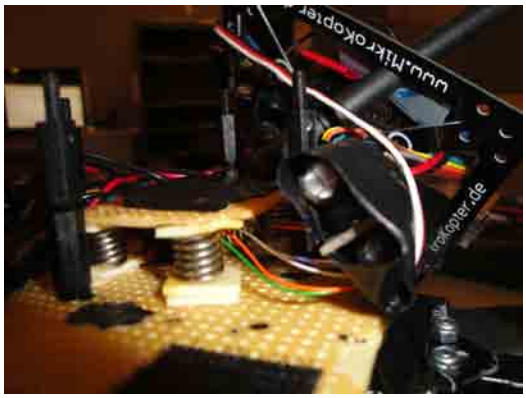
Figures 3.17(a), 3.17(b) contain an experiment where the following items were hand moved and flicked with a finger to see how the IMU readings were affected: battery, ESC, PWM to ESC wires, roll and pitch rods, IMU wires/wire wrap, IMU plug at the board, Gumstix, antenna, and main power switch. It was discovered that the battery, antenna, and Gumstix yielded the most vibrations as well as extraneous data. The connection problems were fixed by strapping the Gumstix down better.

3.3.3 Flight Analysis

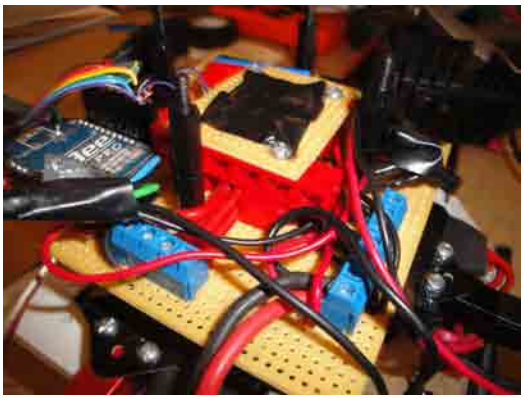
The obvious effect of having such problems with the sensor is poor flight quality. Yaw was nearly completely uncontrollable since the gyro used for yaw was not nearly accurate or sensitive enough to detect the kind of slow yaw typical during a flight, and in the presence of noise, the yaw gyro readings would sometimes cause instability when used in the control algorithm. In general, the quadrotor in flight exhibited poor stability, where high gains caused very large oscillations and consequently poor recovery, while lower gains would cause



(a) Mounted on foam strips.



(b) Mounted on springs.

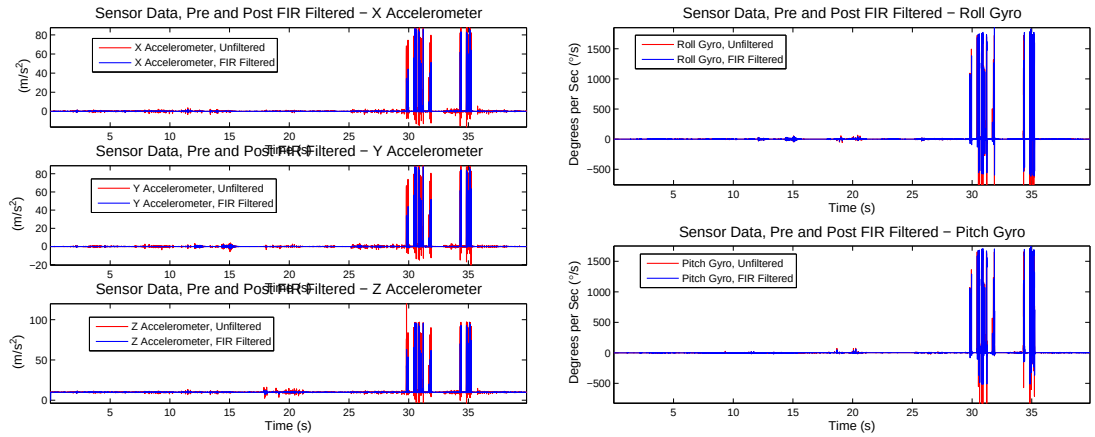


(c) Mounted on legos.

Fig. 3.16: Original IMU sensor mountings.

a smooth flight but would yield slow or no response to recover to a hover condition. Some of this tuning was attempted around one foot from the ground, where air flow through ground effects are problematic, and so testing above two feet from the ground allowed more proper tuning and flight consistency [18, 37].

Consistency in flight characteristics was also an issue, as sometimes there would be several decent flights in a row, and then some flights were just flying off in one direction until changing back again to the former. Figure 3.18 shows the sensor data during actual flight. Although this sensor is widely used by hobbyists, the difference between stable enough for a good pilot to control and a fully autonomous flight is enough to warrant requiring a better sensor. However, the gyro portion of this IMU has been successfully used by another research group, so the true problems with this sensor, or the platform itself, are



(a) Accelerometers. (b) Gyros.

Fig. 3.17: Filtered vs unfiltered measured values of original sensors, component vibration detection.

not known [65].

Aggressive Filtering Results

The final attempt to utilize this IMU for stable flight involved extensive filtering. Hardware filters in the form of capacitors, seen in Figure 3.19, plus aggressive software FIR filters were implemented. Care was taken in choosing the filters, as per the current sensors, in order to not eliminate frequencies that were actually due to the movement of the quadrotor and not just noise. Delay was kept as minimal as possible, on the net order of 10 ms, which should not by itself cause problems. The net result was much smoother sensor readings, but in practice, flights could not exceed 5 seconds due to inability to stabilize to the hover region.

Although the accelerometers were determined to be very noisy, they were not the ultimate culprit, as flights using just the gyros were similarly unsuccessful. A flight with just gyros should yield at least characteristics of flatness and smoothness, even if it is unable to maintain positioning due to accumulating an angle that goes uncorrected. A sensor calibration of 20,000 cycles (over 5 minutes) was also tried, but yielded no improvements. Weight and balance issues were likewise eliminated, as components were shifted around, in one case moving the battery to sit completely on one axis, and yet the flight moved even

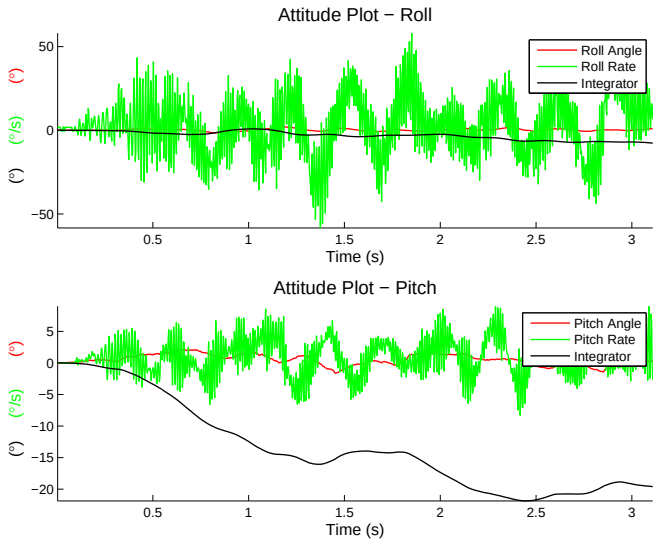


Fig. 3.18: In flight original sensor attitude data.

against this inertia.

Aerodynamic effects were considered, removing the landing gear skids based on the idea that they might be obscuring airflow, but there was no change. Closed loop was then compared to open loop for further elimination of the platform in general, and the open-loop case, although highly unstable, would not be predictable in direction, while this was usually not the case when the loop was closed.

The flight characteristics were highly nonlinear, and when flight actions were closely examined against the sensor readings, it was determined that the sensor did not accurately reflect what was happening, in some cases actually commanding a net thrust in the direction of travel instead of correcting. The resulting conclusion is that the sensor was just not accurate enough, with poor noise coupling issues that could not be filtered out without losing real data. The only reason it was able to obtain some level of stable flight when less extensive filtering was used is that when less filtered, the noise causes the measurement signal to oscillate up and down, which if the average value is actually inaccurate, the oscillations will act like a signum type function, maintaining net control close to the accurate data as opposed to the inaccurate filtered data.

The final condemning factor of the problem being the IMU sensor is that simply using

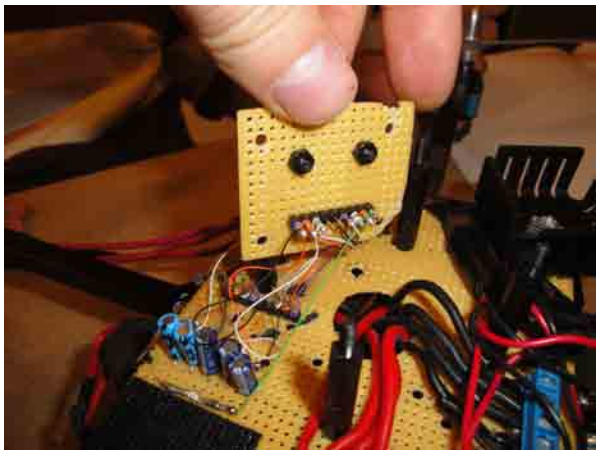


Fig. 3.19: Original IMU sensor with heavy hardware filtering.

the current sensors listed at the beginning of this chapter enabled consistently attitude stabilized flight without any additional changes.

3.4 Attitude Flight Controller

The attitude controller utilizes the accelerometers and gyros to obtain smooth and level flight. The algorithm uses a model independent PD controller. The attitude control loop runs at around 400 Hz where all computation, including filtering, is performed in floating point. Since the Gumstix does not have a FPU, a software implemented floating point is used, which necessarily reduces the speed at which the control loop is completed. Floating point is used to obtain high accuracy at the cost of time; in comparison to a highly successful system that takes the opposite approach of using fixed point numbers in order to obtain a very fast system of 1Khz [38]. A similar approach was deemed unnecessary given that a 400 Hz control loop is significantly faster than most of the initial quadrotors developed, which were on the order of 50 Hz and are described in Section 3.10.

3.4.1 Control System Block Diagram

The attitude controller is a software algorithm with some basic features. The general setup of the control system implementation of the complete attitude flight controller is shown in Figure 3.20.

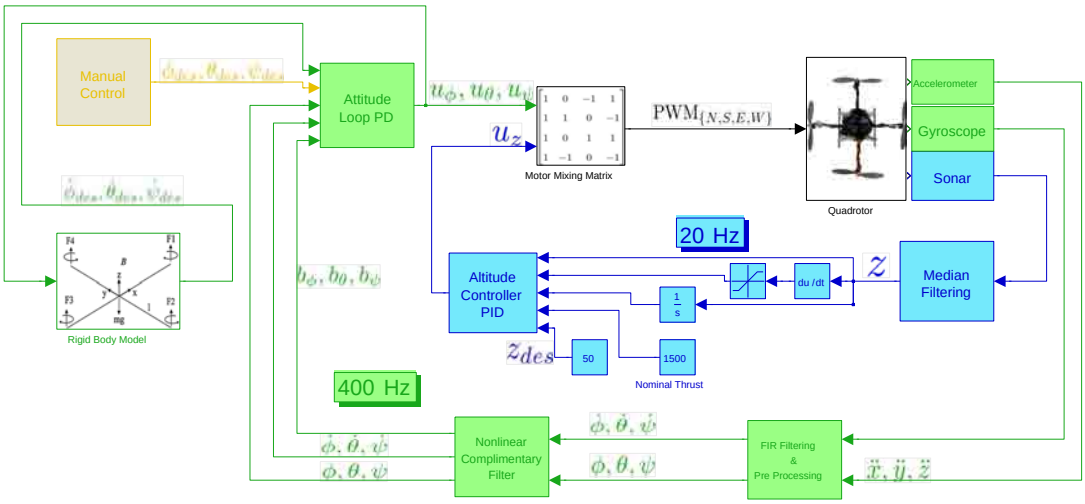


Fig. 3.20: Control system block diagram of attitude controller. The green blocks and wires indicate the attitude control system; the blue blocks and wires are for the sonar altitude controller. A backup manual controller for safety is shown in a tan color.

3.4.2 Control Model

The roll and pitch inputs, ϕ and θ, are governed by

$$\begin{bmatrix} u_\phi \\ u_\theta \end{bmatrix} = \begin{bmatrix} k_{p,\phi} & 0 \\ 0 & k_{p,\theta} \end{bmatrix} \begin{bmatrix} \phi^{des} - \phi \\ \theta^{des} - \theta \end{bmatrix} + \begin{bmatrix} k_{d,\phi} & 0 \\ 0 & k_{d,\theta} \end{bmatrix} \begin{bmatrix} \dot{\phi}^{des} - \dot{\phi} \\ \dot{\theta}^{des} - \dot{\theta} \end{bmatrix}, \tag{3.13}$$

where ϕ^{des}, θ^{des} indicate the desired references.

3.4.3 Control Methods Analysis and Related Work

The attitude control utilizes PD control based on Euler angle representation. Other aspects on top of PD control were attempted with results and analysis described in the following sections. In general, many different controller types have been applied to quadrotors, including PD^2, which was shown to have exponential or asymptotic stability, depending on the model implementation [20]. In simulation, such a controller was even shown to recover from being initially upside down [66]. Traditional PD and PID controllers have been common and successful [18, 31, 38]. Various nonlinear or saturation-based controllers have been implemented [51, 52, 67, 68]. Using measurements of the system response, a linear quadratic

regulator (LQR) has been used with less success than expected [51, 53, 69]. Several types of backstepping controllers have been implemented, yielding good results [19, 54, 55, 70, 71]. Sliding mode controllers have also been implemented, but did not give success on a physical platform [55, 72].

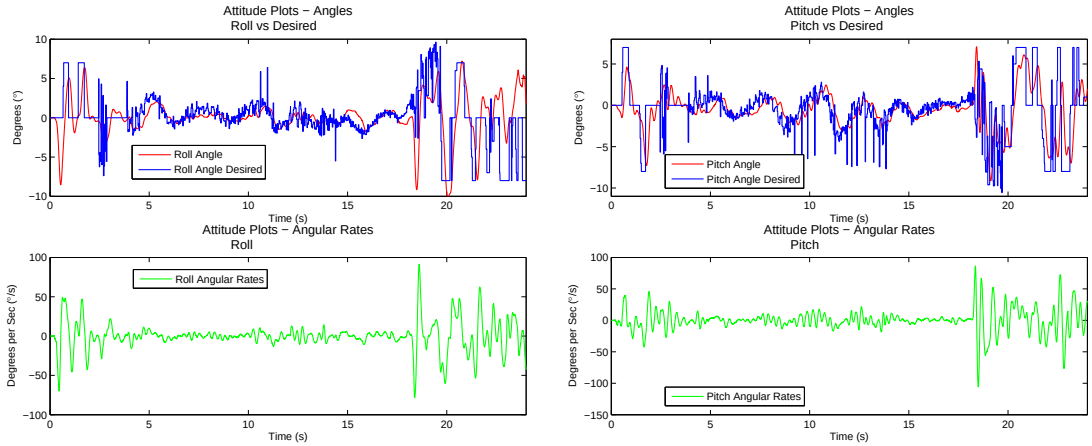
These controllers have delivered various results, usually depending on the type of sensor and platform used, and which nonlinear dynamics were modeled. In some of the cases, the actual application of the controllers to free flight are unknown since the experiments were done with some manner of tethering or restriction of the degrees of freedom. In practice, the most successful quadrotors utilize PD or PID control [7, 31, 38]. As such, PD control is used, given the additional fact that PD is simple to implement and has consistently been shown to give approximately equal, if not better, results than other types of controllers. The fact that a full system model is unavailable also limits the control options, and PD is very effective at controlling a system with un-modeled and nonlinear aspects.

The usage of quaternion representation will avoid singularities, and has been proven to yield stability to PD^2 controllers in both model independent and model-dependent cases [20, 52, 58, 73]. Euler representation is used because it is more intuitive and the computation lost in using it is minimal.

The performance of the attitude controller during a path following flight is indicated by Figures 3.21(a) and 3.21(b), showing the attitude measurements of the roll and pitch axes. During a path following, desired angles are often more extreme because of the movement of the vehicle. The figures of the path roll and pitch behaviors show the attitude tracking capability of the quadrotor, which, other than the time lag due to the system dynamics, the quadrotor is quite capable of following considerable magnitudes in desired references. From this graph, the system response can be estimated at about 160 ms.

Derivative Squared - Feed Forward Term

The use of detected accelerations for stabilizing attitude control is described as being a way to anticipate what the quadrotor will do and to account for the change before it becomes worse [20, 73]. In practice this was found to be problematic since accelerations can



(a) Roll axis. (b) Pitch axis.

Fig. 3.21: Attitude performance during position controlled hover - roll and pitch axes.

occur due to both a change in the angular position of the quadrotor as well from yawing due to the centrifugal force. In addition, accelerometers are somewhat noisy signals, so can lead to jitter when used directly [58]. Utilizing accelerations directly lead to jerkiness and general instability during flight. An example of the acceleration measurements that would be controlled off of is shown in Figure 3.22 where the values have been magnified so that they can be compared to the angle and rate measurements. An alternative to using the accelerometers for such a feed forward anticipation term is to numerically differentiate the smoother and typically more accurate gyro input [74]. This method for improving control could be explored in future work.

Integral Control

An integrator is an important part of general PID control, since it typically provides a stabilizing effect for growing inputs, as well as bringing steady state error to zero. However, an integrator needs time to adapt and only accounts for biases effectively [31]. In a quadrotor, this can be problematic due to at least slight constant motion, making steady state error non-existent. This is coupled with the fact that integrating the angles involves in some part integrating accelerometer data, which leads to poor results because the signal to noise ratio is low. The integrator then acts to accumulate errors and small but slightly inaccurate

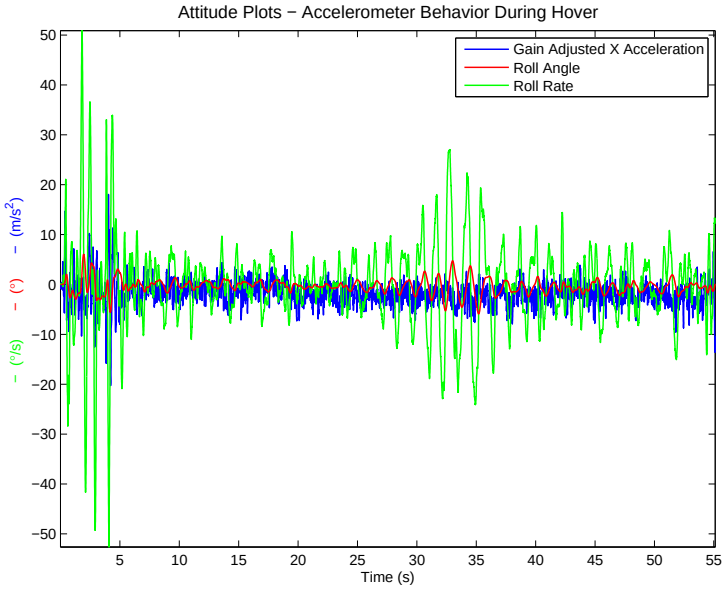


Fig. 3.22: Performance of acceleration signal during position controlled hover - roll axis.

angles, creating a flight that is fine to begin with but will then cause the quadrotor go off in one direction. This is concretized in Figure 3.23 - this data was taken during a position controlled hover, so the quadrotor had to have a net zero angle in order not to drift, yet the integrator would have pushed hard in one direction only.

Signum Term

In many applications, a signum controller has been proven to effectively regulate a system to a desired steady state. Utilizing a signum only controller for a quadrotor application was deemed too risky and ineffective, but attempts were made to add a signum term based on the PD output, adding it to the end PD output, so the net output is shown as

$$\begin{bmatrix} sg_\phi \\ sg_\theta \end{bmatrix} = sgn\left(\begin{bmatrix} k_{p,\phi} & 0 \\ 0 & k_{p,\theta} \end{bmatrix} \begin{bmatrix} \phi^{des} - \phi \\ \theta^{des} - \theta \end{bmatrix} + \begin{bmatrix} k_{d,\phi} & 0 \\ 0 & k_{d,\theta} \end{bmatrix} \begin{bmatrix} \dot{\phi}^{des} - \dot{\phi} \\ \dot{\theta}^{des} - \dot{\theta} \end{bmatrix} \right), \tag{3.14}$$

$$\begin{bmatrix} u_\phi \\ u_\theta \end{bmatrix} = \begin{bmatrix} k_{p,\phi} & 0 \\ 0 & k_{p,\theta} \end{bmatrix} \begin{bmatrix} \phi^{des} - \phi \\ \theta^{des} - \theta \end{bmatrix} + \begin{bmatrix} k_{d,\phi} & 0 \\ 0 & k_{d,\theta} \end{bmatrix} \begin{bmatrix} \dot{\phi}^{des} - \dot{\phi} \\ \dot{\theta}^{des} - \dot{\theta} \end{bmatrix} + \begin{bmatrix} k_{s,\phi} \cdot sg_\phi \\ k_{s,\theta} \cdot sg_\theta \end{bmatrix}, \tag{3.15}$$

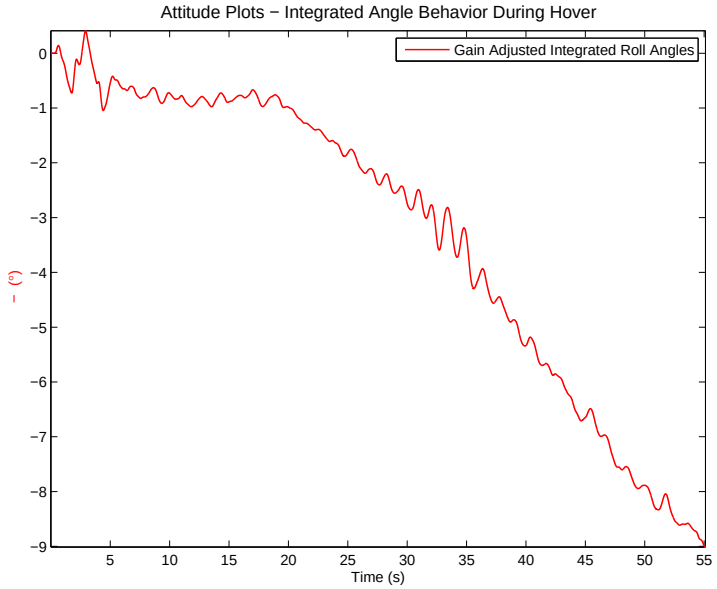


Fig. 3.23: Behavior of integrated angles during position controlled hover - roll axis.

where k_s indicates the signum gains and sgn returns the sign of the function:

$$sgn(x) = \begin{cases} -1 & \text{if } x < 0, \\ 0 & \text{if } x = 0, \\ 1 & \text{if } x > 0. \end{cases} \tag{3.16}$$

This was thought to assist in providing the extra error correction for reducing drift. However, similar to the feed forward term, this just caused jittery-ness and instability.

3.4.4 Gain Tuning and Analysis

Effective PD control requires extensive gain parameter tuning. This tuning was done entirely based on actual flight tests, as all tuning that was tried based on restricting quadrotor freedom or while holding and moving in one's hand was found to give results inapplicable to actual flight. Additionally, tuning was performed over 2 feet above the ground, so that ground effects due to propellor down-wash did not confuse the tuning process for regular altitude flights. The tuning method followed an iterative process whereby gains were raised or lowered based upon the observed behavior. The pitch and roll axes were considered as

decoupled systems for the tuning process. The derivative gains of the gyros were tuned first in order to obtain a smooth attitude, with no jitter detectable by the naked eye in flight or on video. Proportional gains were then added on in order to properly correct accumulated angles. These were kept below the level that a back and forth motion is observed. Finally, the gains were further tuned in response to operator disturbance based interference, where gains, specifically on the derivative, were lowered so that minimal oscillations occur in response to disturbances. A stiffer controller was found to have slightly less drift but would perform poorly with regards to significant external disturbance.

Roll and pitch axes were found to require different derivative gains, likely due to a structural stability difference between the two axes. The roll axis, which is physically felt to have more frame rod movement, requires lower gains.

3.4.5 Yaw Control

Like the primary attitude axes of roll and pitch, the yaw is similarly controlled using a PD controller

$$u_\psi = k_{p,\psi}(\psi^{des} - \psi) + k_{d,\psi}(\dot{\psi}^{des} - \dot{\psi}), \tag{3.17}$$

where ψ^{des} indicates the desired reference. Angles are obtained by integrating the gyro rates. The yaw gyro bias was observed to be very small, but is still nevertheless estimated using the complementary filter. Yaw angles were observed to be under reporting; when visually seeing $30-45^\circ$ angles, the yaw measurement only reported about 20°. These errors were much reduced for small angles, and so the gains were simply increased in order to prevent such large yaw effects from occurring. Compared to the attitude axes of roll and pitch, the yaw rotational force dynamics are much slower acting and so much higher gains are required to yield effective results.

The performance of the yaw control is indicated in Figure 3.24, where an error of only $\pm 2.5^\circ$ is achieved. Note that while this is the yaw measured from the attitude, the actual yaw is controlled using the Kalman filter fused yaw that is described in Section 4.6.9, which yields a tighter true yaw error. In a purely attitude controlled yaw, the yaw error is equally

as small, but during the flight the gyro integrated errors start to accumulate enough that the measured yaw angle will be larger than the true yaw, so that the small error does not reflect the actual yaw of the vehicle.

3.4.6 Model-Independent Control

Given the difficulty of accurate system identification with onboard sensors, PD control is implemented without accounting for the true dynamic effects of the actual quadrotor [75]. Although done using quaternions, PD implemented model independent control has been theoretically proven effective for attitude stabilization [20, 73]. In this implementation, the quadrotor is considered to perform near the hover region, where low amplitude motions occur, yielding small roll and pitch angles. Thus, linearization of Equation (2.3) is employed using small angle assumptions, which also yields the identity $\dot{R} = \Omega$ from Equation (2.6).

3.4.7 Rigid Body Model Control for Roll and Pitch

Utilizing a rigid body dynamic model based approach for controlling roll and pitch, the attitude controller takes the calculated inertia matrix and the roll, pitch, and yaw commands from the PD control loop and outputs a desired angular acceleration for the roll, pitch, and yaw, which are then integrated to produce the desired angular rates for the next loop [76]. This model requires a system identification of the motor for how the thrust changes with RPM for the propellor, the values for which are indicated in Chapter 2. Using the approximation that the rotation matrix from the body frame to the inertial frame for the angular velocities is identity and linearizing about the hover point with small angle approximations, the model in Equation (2.7) takes the form

$$\dot{\phi}^{des} = \frac{4k_F L\omega_h}{I_{xx}}(u_\phi k_{\text{rpm}} + c_{rpm}), \tag{3.18}$$

$$\dot{\theta}^{des} = \frac{4k_F L\omega_h}{I_{yy}}(u_\theta k_{\text{rpm}} + c_{rpm}), \tag{3.19}$$

$$\dot{\psi}^{des} = \frac{8k_M \omega_h}{I_{zz}}(u_\psi k_{\text{rpm}} + c_{rpm}), \tag{3.20}$$

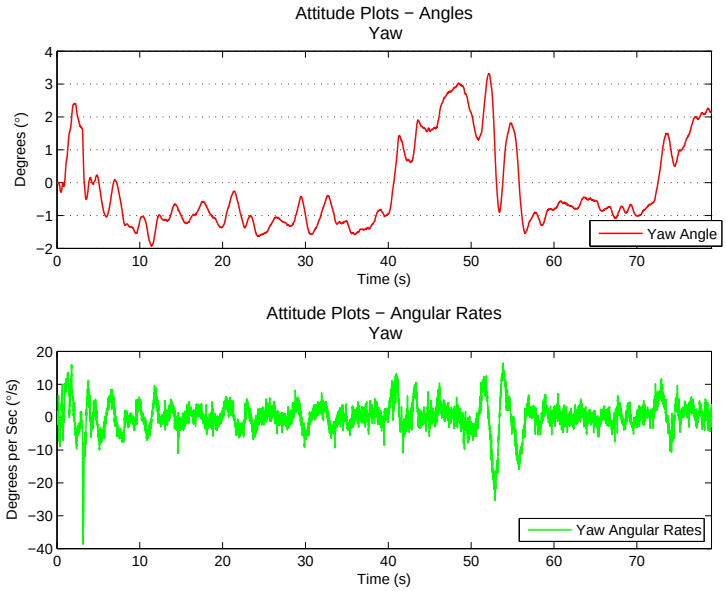


Fig. 3.24: Attitude performance during position controlled hover and path following - yaw axis.

where the motor force and moment constants k_F and k_M are modified based on those from Michael et al. [76]. L is the distance from the axis of rotation of the rotors to the center of the quadrotor, and is 23.2 cm. k_{rpm} converts the u_ϕ, u_θ, u_ψ PD commands to RPMs, as they are in terms of PWM values and is determined per Chapter 2 to be equal to 10 when around the hover thrust region. An offset, c_{rpm}, is needed to match the nominal RPM with the nominal PWM. $u_\phi,\ u_\theta,\ u_\psi$ are determined per Equations (3.13) and (3.17). The terms $I_{xx},\ I_{yy},\ I_{zz}$ are the diagonals of the inertia matrix. The inertia matrix was calculated using direct measurements of the distances and masses of the quadrotor, being as accurate as possible by calculating large components as containing subsets of smaller components. The determined inertia matrix is shown in Equation (3.21) and is similar to other quadrotors [37].

$$\begin{bmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{xy} & I_{yy} & I_{yz} \\ I_{xz} & I_{yz} & I_{zz} \end{bmatrix} = \begin{bmatrix} 0.02509 & 0.00016 & -0.00276 \\ 0.00016 & 0.02610 & 0.00070 \\ -0.00276 & 0.00070 & 0.02262 \end{bmatrix}, \text{ in } kg-m. \tag{3.21}$$

This model based control was implemented subsequent to significant testing with the

more model-independent controller, and so a comparison of the results of the two types of controllers can be obtained. Although actual differences in performance can only be measured by observation and not so much a quantitative comparison since the quadrotor was already flying stably, the model-based control did seem to improve attitude stability and response. An indication of the difference between the model-based desired rates and the actual measured rates is shown in Figures 3.25(a), 3.25(b), and 3.25(c) for roll, pitch, and yaw respectively. The two are rather close mostly, except for at the larger rate changes. This data is from a path maneuver, and so the rates are being calculated based upon the desired movement causing them to be different from the measured rates, specifically at the larger values where activity is occurring. The yaw is drastically different due to the inability of the quadrotor to physically drive the yaw rates to zero.

3.5 Altitude Control

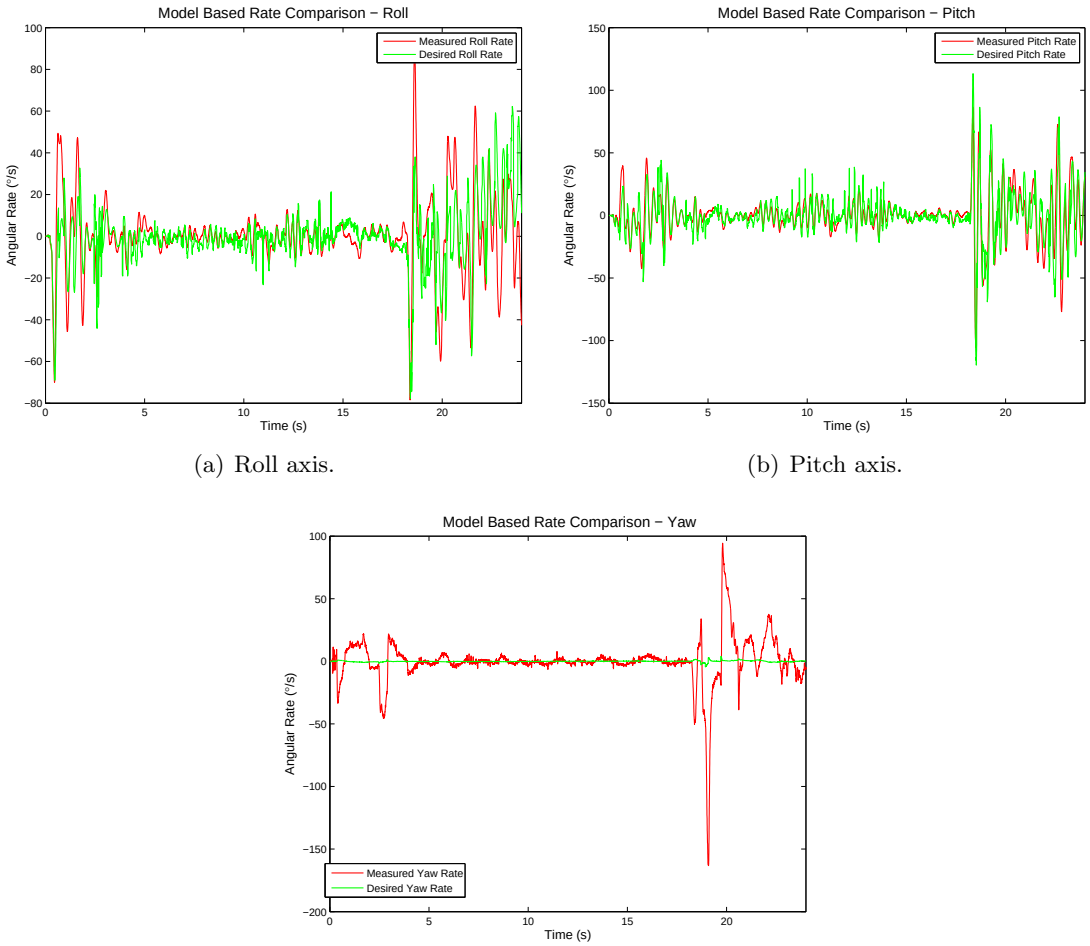
Maintaining consistent height above the ground for a quadrotor vehicle involves controlling the total downward thrust. Since total thrust varies with the tilt angle of the quadrotor as well as airflow effects and reducing battery power, a feedback controller is required.

3.5.1 Control Method

A PID controller is used without dynamic model components [31]. A PID controller, along with a feed forward nominal term, is found to compensate effectively for the nonlinear effects involved in the altitude dynamics of the quadrotor

$$u_{alt} = k_{p,alt}(z^{des} - z) + k_{i,alt}\int_0^t (z^{des} - z)dt + k_{d,alt}(\dot{z}^{des} - \dot{z}) + u_{nom}, \tag{3.22}$$

where z indicates the measured height along the vertical axis, and z^{des} is the desired reference. Unlike the other attitude controllers on the system, the integral component is applicable and necessary with altitude since available power decays over time due to the draining battery and can be successfully accounted for with a linear controller. Using a



(a) Roll axis.

(b) Pitch axis.

(c) Yaw axis.

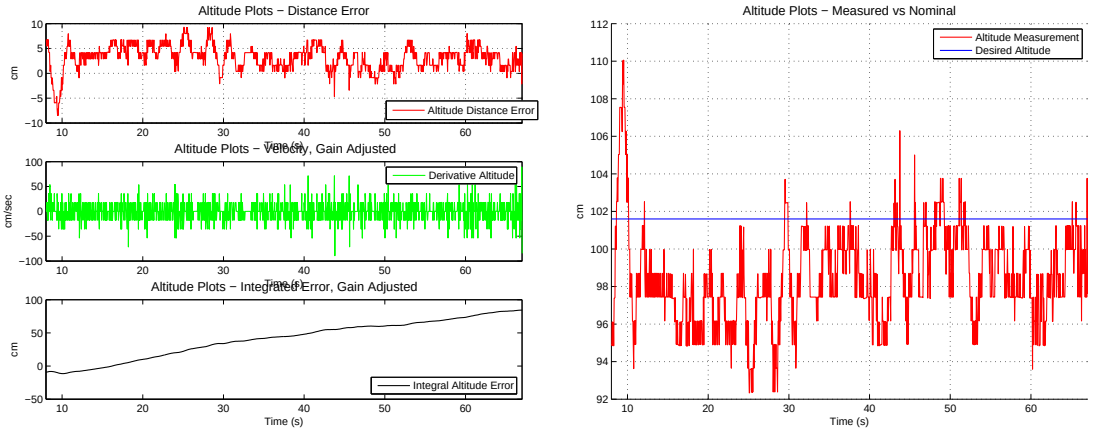
Fig. 3.25: Attitude comparison of model-based desired rates to measured rates during tracking manuever.

dynamics-based height control which will include such aspects as thrust loss due to tilting, is not utilized, but could be included in future work to provide height stabilization improvements.

The performance of the altitude controller during a hover flight is indicated in Figure 3.26(a). Figure 3.26(b) shows the sonar measurements about the nominal height. Steady state altitude approximate error is within 92 to 104 cm, for ±6 cm.

3.5.2 Determining Velocity

A velocity component for the altitude is required for the PID controller, but the only



(a) z axis. (b) Measured height vs nominal.

Fig. 3.26: Attitude performance during position controlled hover - z axis.

available data are the position information from the sonar and the z-axis accelerometer data. Successful use of integrating the accelerometer, per

$$\dot{z} = \int_0^t (\ddot{z})dt, \tag{3.23}$$

where $\ddot{z}$ is obtained directly from the accelerometer, in order to gain velocity has been done, but when tried on this platform it was entirely unusable [31]. As discussed, the noise and inaccuracies in the accelerometers cause the integrated accelerations to be offset over time, yielding false velocities. In addition, the changing accelerations cannot be determined to be tilt or vertical motion.

Simply using single-step differentiation of the median-filtered sonar values gave results that could be controlled off of

$$\dot{z} = \left(\frac{z_{curr} - z_{prev}}{dt} \right). \tag{3.24}$$

However, gains could not be made very high on the derivative term due to the noise in the sonar coming from quantization, sensor resolution, and physical effects that cause the sonar to give values different from the actual height (such as objects or sound reflections). Filtering was not considered an option due to the already delayed values through the median filter. The derivative term is essentially made ineffective about the desired height since the

quadrotor has slow height hover dynamics and the gains are low. The derivative only comes into effect during disturbances or take-off or landing, and acts simply to slow the quadrotor down. Figure 3.27 shows a comparison between the two methods of calculating the velocities.

Similar to the feed forward anticipation term for attitude control, applying such a direct acceleration control approach to the altitude controller for mitigating the delay due to the median filter is also problematic, due to the same problems with the noise of the accelerometer.

3.5.3 Height Control Using Attitude Information

An improvement in the height control can be made by taking into account the angular tilt of the quadrotor [51]. This projection is applied to each sonar measurement before being used to calculate the derivative or the integral in order to achieve the true inertial vertical distance

$$z_{inertial} = (\cos\phi \cdot \cos\theta) \cdot y_{\text{sonar}}, \tag{3.25}$$

$$z_{curr} = z_{inertial}. \tag{3.26}$$

Without this projection, if the quadrotor is not exactly level, than the measurement given by the sonar will not quite reflect the true vertical height in the inertial frame. This yields a rather small difference, as a tilt of 3° at 5 feet will only yield 0.2 inches difference between the sonar measurement and the vertical height. With a sensor resolution of 1 inch, this essentially gets washed out; however a slight accuracy improvement can still be achieved for larger angles in aggressive maneuvering, and the use of a more accurate sensor will make this more useful.

3.5.4 Automatic Landing

A state machine-based automatic landing procedure is employed to safely land the quadrotor, shown in Algorithm 3.1, as manual mode landing often leads to high impact or

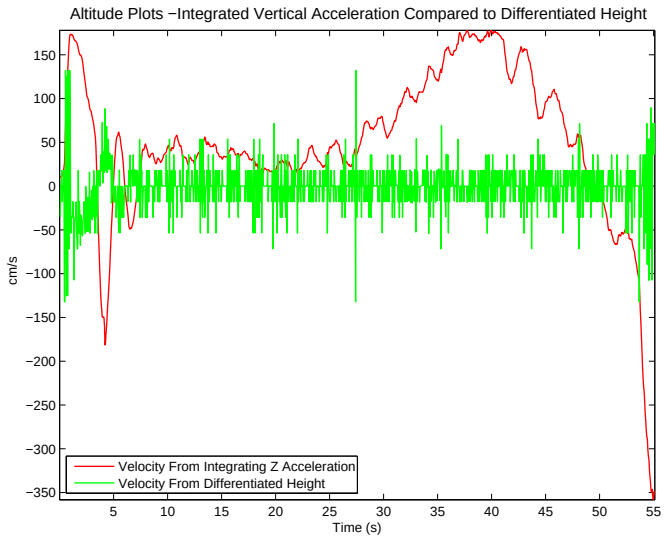


Fig. 3.27: Altitude velocity calculation method comparison - integrated Z acceleration vs differentiated height position.

drifted landings. The automatic landing is performed by reducing the desired height by a fixed amount each time the height controller runs, and then once a low enough height level is obtained, the landing state changes to a terminal landing state. This final state simply reduces thrust to each motor over a fixed number of height control loops until it simply shuts off the motors. The low-height level state change is needed in order to account for the fact that the sonar can only detect distances greater than 6 inches (15.24 cm), and the effect of flying near the ground causes significant disturbances, so the time spent with motors on at this level needs to be minimal.

One other aspect taken care of is an adaptive landing trimming, which accounts for the reducing weight of the camera USB cable that is attached to one side of the quadrotor, so that the aircraft does not drift one direction while landing. This state machine landing performs very adequately, with smooth height reduction and minimal drift even when initiated from an aggressive flight. Robust landing controllers have been proposed, but were determined not to be needed [77].

3.6 Integrated Stabilization Method

The net output to each motor is a combination of the different controller commands,

Algorithm 3.1 Automatic Landing

Input: Sonar Altitude Measurement, y_{sonar}
Begin
  $\dot{z}^{des} \leftarrow$ LandVelocity
  if $y_{\text{sonar}} >$ LandingMin **then**
    $z^{des} \leftarrow y_{\text{sonar}} -$ LandingMin $- 1$
    $\text{trim}_\phi \leftarrow \text{trim}_\phi + \text{trim}_\phi^j$ /*Adjust for reducing weight of camera cable*/
    $\text{trim}_\theta \leftarrow \text{trim}_\theta + \text{trim}_\theta^k$ /*Adjust for reducing weight of camera cable*/
  else /*Final Landing Sequence*/
    if FinalLandingCounter **then**
      $u^i_{\text{PWM}} \leftarrow$ OFF
    else
      $u^{alt}_{nom} \leftarrow u^{alt}_{nom} - u^{alt}_{adj}$
      $\text{trim}_\phi \leftarrow \text{trim}_\phi + \text{trim}_\phi^j$ /*Adjust for reducing weight of camera cable*/
      $\text{trim}_\theta \leftarrow \text{trim}_\theta + \text{trim}_\theta^k$ /*Adjust for reducing weight of camera cable*/
      FinalLandingCounter $\leftarrow$ FinalLandingCounter + 1
    end if
  end if
End

sent to the ESCs as a PWM value. This net motor mixing consists of the roll and pitch controllers, and the yaw and the height controller. All PD/PID outputs (roll, pitch, yaw, altitude) are determined separately and then mixed together at the end for the final command to the motors. This mixing is shown mathematically as

$$\begin{bmatrix} u^1_{\text{PWM}} \\ u^2_{\text{PWM}} \\ u^3_{\text{PWM}} \\ u^4_{\text{PWM}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 1 \\ 1 & 1 & 0 & -1 \\ 1 & 0 & 1 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix} \begin{bmatrix} u_{alt} \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix}, \tag{3.27}$$

where u^i_{PWM} denotes the output PWM value to motor i. The output commands to one motor on the roll and and one on the pitch are shown in Figure 3.28. In the prototype setup, motor 1 is the south motor, and motor 2 is the west motor, with the platform structure as outlined in Figure 3.5. The u^i_{PWM} outputs to each motor are related to the airframe torque,

$\tau_a = (\tau_a^1, \tau_a^2, \tau_a^3)^T$, of the dynamic model in Equation (2.7), with

$$\omega_i = k_{rpm} \cdot u^i_{\text{PWM}} + c_{rpm},\ i \in 1,2,3,4, \tag{3.28}$$

$$\tau_a^1 = L \cdot b(\omega_2^2 - \omega_4^2), \tag{3.29}$$

$$\tau_a^2 = L \cdot b(\omega_1^2 - \omega_3^2), \tag{3.30}$$

$$\tau_a^3 = k(\omega_1^2 + \omega_2^2 + \omega_3^2 + \omega_4^2). \tag{3.31}$$

3.7 Manual Control - Operational Interference

Having a manual control input method is crucial to preventing crashes and improving safety. In addition, several useful features are assigned to the manual controller for easy testing, such as height enable, automatic landing initiation, and navigation enable.

In general, either with the type of joystick controller in use it is difficult to control aerial vehicles, or RC piloting is just difficult, because oftentimes utilizing the manual controller would actually cause aggressive changes. Reducing the sensitivity of the roll and pitch stick prevented such human error commands since large changes in direction could no longer be set, although control authority is reduced.

3.7.1 Direct Actuation for Disturbance-Based Control

Originally for ease of implementation, the manual flight control aspect was implemented as a disturbance, meaning that the controller gave direct roll and pitch commands to the motors in terms of changing the PWM value sent per

$$\begin{bmatrix} u^1_{\text{PWM}} \\ u^2_{\text{PWM}} \\ u^3_{\text{PWM}} \\ u^4_{\text{PWM}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 1 \\ 1 & 1 & 0 & -1 \\ 1 & 0 & 1 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix} \cdot \left(\begin{bmatrix} u_{alt} \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} + \begin{bmatrix} u_{\text{man,thrust}} \\ u_{\text{man},\phi} \\ u_{\text{man},\theta} \\ u_{\text{man},\psi} \end{bmatrix} \right), \tag{3.32}$$

where $u_{\text{man},*}$ refers to the direct outputs from the manual controller sticks.

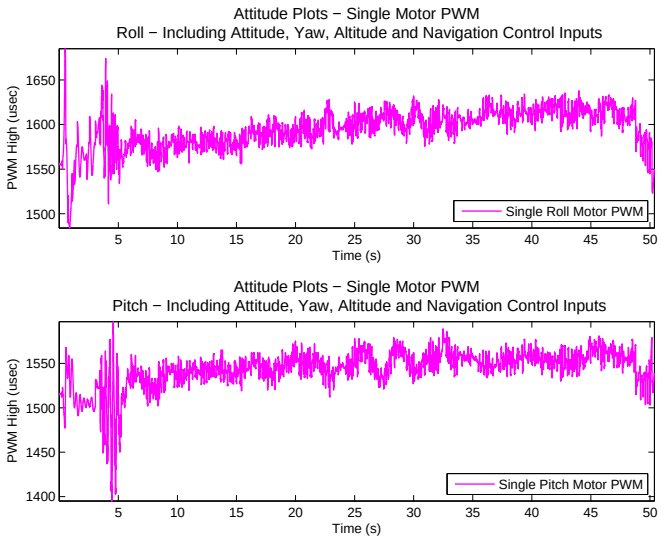


Fig. 3.28: Output command to motors from controller mixing - one motor each for pitch and roll.

This meant that in terms of the quadrotor attitude controller, it is sensing changes in the stability which are not due to its own control, very similar to what the case would be if the quadrotor were to just be moved by hand or other outside disturbance. Such control made the quadrotor difficult to maneuver manually since it would be constantly trying to correct against the manual input, however it was instrumental in tuning the controller to reject disturbance. As described above regarding stiff and loose controllers, a stiffer controller would yield a very tight hover, but a simple movement on the manual would cause fast oscillations that would be slowly damped out, while a loose controller would quickly settle back to the hover point after receiving such a manual input.

3.7.2 Desired Angle Control

Ultimately for full control of the quadrotor in terms of movement around the room, an input that worked with the attitude controller, much like a navigation system, was required.

The values from the manual roll and pitch are then interpreted as desired angles,

$$\begin{bmatrix} \phi^{des} \\ \theta^{des} \\ \psi^{des} \end{bmatrix} = \begin{bmatrix} u_{\text{man},\phi} \\ u_{\text{man},\theta} \\ u_{\text{man},\psi} \end{bmatrix}, \tag{3.33}$$

and go directly into Equations (3.13) and (3.17), after being scaled appropriately in the host code. This desired angle is thus fed into the attitude controller as the desired reference in order to appropriately command the motors to tilt to that angle. Thrust is handled the same way as before, as a direct feed forward term to all four motors; manual is not set up to send desired altitude reference commands as it was not seen to be necessary.

3.8 Latencies and Delays

Excessive latencies and delays ultimately lead to system instability. As such, they are minimized as much as possible. Figure 3.29 shows an overall diagram indicating the latencies and delays within the system.

3.8.1 ADC Sampling

ADC sampling is minimized by continuously sampling and having the communication request be interrupt based, so that the sampling is not done in order to send out the values. Interrupts are disabled for just the three cycles of variable storage during sampling in order to prevent contention over the variable. If an interrupt occurs during the disabled period it is executed upon re-enabling interrupts, yielding a very small delay. Receiving the last cycle's sampled value is recent enough due to the reasonably fast Robostix that not having sampling and sending in the same loop is not an issue.

3.8.2 Communication

To minimize the time required to operate the system, communication start and stop is limited to once per component. For all components attached to the Robostix, a single read

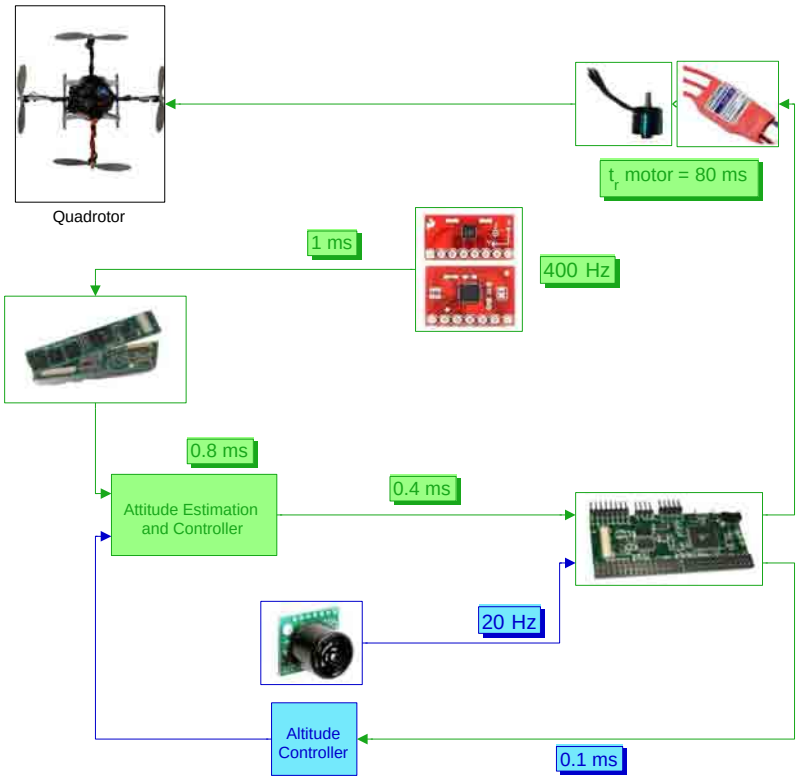


Fig. 3.29: Attitude controller and system latencies.

command gets values for each one, and a single write command gives all the motor PWM values.

Some basic code profiling was performed, using the processor built-in timer. Results indicated that roughly 40% of the total loop time is required just to get sensor data.

I^2C

The I^2C protocol is set to run at 400 Kbps, which is the fastest the devices used will allow. Rough net throughput of data for the protocol is 27.5 Kbytes/sec plus an estimated overhead of 0.3 ms.

Zigbee

Zigbee latencies are primarily involved with the navigation system and is mentioned

further in Chapter 4. The Zigbee is used for transmitting manual control packets, but this does not impact the system as a whole other than the minimal computation involved when checking for packets. The Zigbee easily keeps up with the speed at which a human operator can utilize the manual controller.

3.8.3 Data Logging Delays

Printing commands are notoriously time consuming and problematic for real-time systems. Simply printing to the host computer screen via Wi-Fi adds over 10% to the time required for each control loop. Printing to a file onboard the Gumstix is less damaging, but still leads to large spikes in loop time periodically while the file buffer is loaded into the file. These spikes are typically 90 ms in duration as shown in Figure 3.30. Even with such periodic, excessive delays, the attitude system is still able to maintain level flight in the presence of such latencies.

3.9 Issues

No physical system, and especially a flying system, is built without running into problems, surmounting difficulties and learning a few lessons.

3.9.1 Mechanical and Aerodynamic Considerations

As a flying system, aerodynamics play a large roll in the stability exhibited by the quadrotor. Several key aspects are crucial to understand in order to obtain quality in flight. These main aspects include weight and balance, propellors, and testing by limiting the degrees of freedom.

Structural

Ever since the landing gear was placed underneath the pitch axis, the roll axis has always had more noise and oscillations during flight. This was determined to be from a lack of stiffness on the roll axis, that the landing gear struts added to the pitch axis. The extra noise on the roll axis meant that lower gains had to be used for stable control and

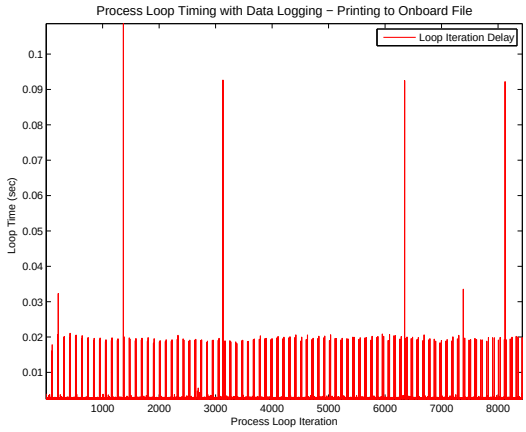


Fig. 3.30: Process delays with data logging. Note the spikes at around 90 ms, plus the more frequent 20 ms loop latencies. However, since the vast majority of loops are at the floor level of around 2.5 ms, the average loop frequency is still about 400 Hz.

that performance was always worse than for pitch. To remedy most of these effects, an aluminum reinforcement plate was mounted beneath the roll axis. This improved the roll, although still not quite to the quality of the pitch axis.

Weight and Balance

Fixed wing aircraft require proper balancing in order to lift off and efficiently maintain level flight. The quadrotor aircraft is no different, except possibly being even more critical, as the dynamics are not self-stabilizing. A CG too far from the actuation plane (the location of the propellors) will place a large moment of inertia on the quadrotor to move in that direction, requiring even more correction from the attitude controller.

Initially, balancing of the quadrotor was performed by holding up the two ends of each axis and seeing the rotational torque. This lead to a misunderstanding of the system behavior due to thinking it was balanced, because in order to make it balanced around the frame, the motors and propellors had to be placed upside down, which actually caused the CG to be above the actuation plane. Since the quadrotor moves by adjusting the speed of a given motor and consequently the thrust of the propellor, having the CG above this plane meant that the quadrotor had to work extra hard to stabilize itself, yielding a reverse pendulum stability problem.

Ultimately, the CG needs to be calculated and adjusted so that it is placed in the center of the quadrotor, close to the rotor plane [37]. Complete coincidence with the rotor plane is not desirable, since the sensitivity to the CG around that point will be high [18]. A slightly below rotor plane CG is more favorable than above, since damping will be better. This configuration was found to be effective for both a desired translational movement as well as disturbance rejection, which was done by disturbing the quadrotor by hitting it. This is contrary to another analysis that a below rotor plane CG will lead to poor disturbance rejection [16]. Possibly the wind gust type of disturbance described is a special kind of disturbance that is difficult to replicate using impositions of physical force.

Propellors

As the main aerodynamic component of the system, and the only one generating thrust, understanding its issues and effects are required for maintaining a proper flying system. One obvious such issue, but subtle in understanding, is the effect of damaged propellors on the flight characteristics. In general, it is just important to know that the flight characteristics do change, and to account for that in empirical testing. Sometimes, however, damaged props can be used to balance out the net quadrotor response due to CG, inertia, and motor properties. Typically the thrust is lower with a damaged prop, but sometimes only the yaw rotational force is changed.

Yaw force is one noticeable anomaly in the specific propellors in use on this quadrotor. Since the propellors are counter-rotating in order to control for yaw, it would seem a given that for equal thrust levels, both propellor types would yield the same amount of rotational torque. This was found to be incorrect. The pusher type propellor, clockwise rotating, always generates significantly more rotational torque than the standard type. This asymmetry is taken care of by trimming one axis. General thrust level is not a concern since roll and pitch are based on relative thrusts of the same type of propellor.

Trimming

Trimming is a required accommodation to the fact that not all motors, propellors or

even just the physical system dynamics, are equal or level. Trimming is performed by giving a fixed offset value to each motor, in the roll, pitch, and yaw axis pairs, as shown in Equation (3.34). Once level flight is achieved, trimming can be calculated based on a general average command for roll, pitch, and yaw stabilization, noting that if it is constantly adjusting one direction it likely means some trimming will help. Prior to flying, trims can be estimated based on what is known about the physical system, and in the case of the propellors used, relational thrust effects.

$$\begin{bmatrix} u^1_{\mathrm{PWM}} \\ u^2_{\mathrm{PWM}} \\ u^3_{\mathrm{PWM}} \\ u^4_{\mathrm{PWM}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 1 \\ 1 & 1 & 0 & -1 \\ 1 & 0 & 1 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix} \cdot \left(\begin{bmatrix} u_{alt} \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} + \begin{bmatrix} 0 \\ \mathrm{trim}_\phi \\ \mathrm{trim}_\theta \\ \mathrm{trim}_\psi \end{bmatrix} \right) \tag{3.34}$$

When attaching the camera to the quadrotor with the subsequent USB cable hanging from it, trims had to be largely adjusted for all three axes. With a lighter camera cable, these trims were reduced. Typical values for these trims are on the order of a net 100 RPM differential between the motors on each axis, depending on some of the factors indicated above, and that is even with a fairly balanced inertia, as shown in Equation (3.21).

It is important to note that trims should not be used for stabilization, but rather for assisting the attitude controller for improved efficiency and so that the gains do not have to be adjusted. It can often be tempting to change them for attitude stabilization, as it was when using the original sensor with its unpredictability. Sometimes a consistent behavior in one direction is due to a problem with the attitude controller and not the trims, so it is always better to have actual evidence that there is a balance or thrust differential problem in the system that needs trimming correction than to apply it purely based on empirical testing. As such, trimming should be a constant across the operational range (excluding such obvious aspects like a wire that adds more weight as the height increases), unlike the controller gains, which would need to change for low ground flight per Section 3.9.3.

Adaptive trims were examined but not considered feasible or advantageous enough, as a human understanding of the situation is needed for appropriately adjusting the trims.

Testing by Limiting the Degrees of Freedom

Following upon the need to balance the quadrotor around the rotor plane and not around the frame is the peril involved with attempting to design or tune the attitude controller based upon testing in a setup where the quadrotor is not free to fly on its own. Great care must be taken when limiting the motion of the quadrotor as testing and analysis results may not transfer over to actual flight. This has to do with both rotor plane actuation as well as dynamic coupling. The most reasonable method seen in literature utilizes a fully gimbaled system, similar to inertial frame IMUs, although coupling dynamics are avoided and the full carryover to true flight is still unknown [71].

Initially, attempts at testing began with the quadrotor mounted in some way to a rig, either by removing the roll or pitch axis and mounting through the frame, or by mounting through the top or underneath the base of the quadrotor. Typically the results were in the form of gains that were too high or too low, but sometimes the rigged quadrotor would simply behave unpredictably with no understanding of what needed to be done.

3.9.2 Sensor Difficulties

Aside from the extensive sensor difficulties faced with the original sensor covered in Section 3.3, sensor issues have been relatively mild. The sonar, which is mounted underneath the quadrotor between the landing gear skids, will sometimes detect the camera cable nearby. This of course causes enough readings of a shorter distance that the median filter passes them through, causing the quadrotor to rise in height due to the interference. Resulting measured error is twice as bad when the cable is roughly under the quadrotor than when it is pulled as far to the side as possible. This could be solved by utilizing another sensor and fusing the data, or placing the sensor in a different location, such as near the motor farthest from the cable.

Another sonar problem is the occasional extraneous value, which can be dealt with through the use of the median filter. However, sometimes the sonar just gives very wrong data, possibly due to reflection scatter. On the ground this leads to extremely large distance measurements (often the max allowable) and in the air is sometimes the opposite, yielding

a thrust up command. Lengthening the median filter is undesirable due to the already low sampling rate and may not even take care of the problem. For the most part this issue is rare enough that it will not repeat on a subsequent flight and so the problem is just accepted and moderated with safety factors.

The power system has a slow startup issue through the Robostix which causes the output pins to not climb to the appropriate voltage levels until after the Robostix has completely booted up. This causes issues with the attitude sensors that derive their power from the Robostix. This was initially a confounding problem, where the gyros would give very large values when stationary. Once determined that the problem was due to the gyro initializing with an improper voltage level, simply reseting the gyro using a built in reset register prior to starting the flight program allowed a proper sensor initialization.

3.9.3 Ground Effect

Thrust generation from a rotating propellor yields a curved field of airflow out and down. At low altitudes of less than one meter, the airflow will bounce off the ground and create disturbances on subsequent propellor rotations [37]. This will create difficulties in stability as there are extensive nonlinear disturbances occurring directly beneath the quadrotor. Attempts to control at this height were abandoned due to the aerodynamic complexity that would need to be taken care of as well as the fact that the system requirements for this quadrotor are for action well above the ground effect altitude.

3.9.4 Safety

For a vehicle that can crash into things and people at high speeds with spinning blades that can damage skin, it is only logical to include significant safety features. Some of these features are simply to prevent the quadrotor from doing anything too crazy, while others are to protect some of the components or the quadrotor itself in the event of a crash.

Crash Resistant Safety Measures

Flight pre-checks are used to ensure that all motors and propellors are firmly attached.

Since a single actuator loss yields instant instability, having a motor or propellor loosen during flight would immediately result in a crash and has happened. Similar checking is done for the rest of the quadrotor, including the battery, the Gumstix, the frame, and the landing gear.

The height controller implements saturation limits on the determined height error and the calculated velocity, since as a physical system, it will not be able to change height fast enough anyway, so this reduces the effects of extraneous measurements.

Damage Prevention Measures

A power switch is installed between the battery and the ESCs, so that an operator can manually turn off the motors if other means are exhausted.

Both a manual controller and the keyboard can turn off the motors, with one being sent over Zigbee and the other over Wi-Fi, yielding communication redundancy. Should the flight program terminate or fault for some reason, a special program is installed, called Down, which when run will simply send a low throttle command to the motors.

A maximum thrust setting is in place, mainly for preventing possible burnout of the motors, since the power supplies and ESCs are capable of handling far more current than can be required.

3.10 Related Work in Attitude Experimentation

A survey of the related work in quadrotor attitude control is presented here, including the type of controller used, the experimental platform, and the results achieved.

3.10.1 Tethered or Restricted Attitude Control

Initial quadrotor research began with restricting the motion of the unstable system in order to determine proper control techniques. Sliding mode and backstepping controllers were implemented on a custom OS4 quadrotor fixed to a test bench to measure roll, pitch, yaw, and were able to regulate large starting angles to close to 0° relatively quickly [55]. The custom heavy X-4 Flyer, controlled with an integrator and inverse plant model, enabled

overshoot of just a few degrees to a large angle step reference input when constrained to a test stand [37]. Restricting a modified Draganfly quadrotor using a ball joint fixture, a PD^2 controller with both model independent and model dependent methods allowed angle and velocity regulation [10, 20]. Using an observer and sliding mode controller, a simulated quadrotor was able to track attitude references using only x,y,z and heading information [72].

3.10.2 Attitude Control Using 3D Tracking System

3D tracking systems became highly popular with the availability of the Vicon system [78]. These systems allow accurate attitude estimation without the problems of noise inherent in IMUs, including mechanical vibration noise. The Vicon system, depending on the number of cameras used and some other factors, is capable of sub-mm accuracy and over 100 Hz update rates. Since they are located outside the quadrotor, the quadrotor has a restricted area within which it can move. A nonlinear saturation controller was applied to the Draganfly platform and using a 3D sensor was able to achieve attitude regulation hovering with restricted position control for a close hover and capable of disturbance rejection; an LQR controller was implemented but was not good enough for flight [51]. Use of the Vicon system and an LQR controller enabled the Draganfly platform to perform hovering and trajectory following to very accurate levels [53, 79]. Utilizing PD control and one of the first Ascending Technologies Hummingbird platforms, position detection using the Vicon enabled a very small trajectory following error of ± 10 cm [6, 38].

3.10.3 IMU-Based Attitude Control

The custom OS4 quadrotor was implemented with a PID controller and was able to regulate attitude; LQ control was explored on a test bench but not able to provide enough stabilization for a free flight [69]. The custom X4-flyer used a nonlinear saturation controller, and was able to achieve short duration tight angles, and when given a manual angular reference to track, was able to achieve the angle within a couple seconds [52]. Using PID control, the custom Starmac quadrotor was able to achieve sinusoidal angular tracking

to within a few degrees and good altitude tracking; for trajectory tracking, desired angle tracking was obtained; an overhead camera implemented hover enabled a position regulation to within an 80 cm diameter circle [31]. Using a backstepping controller, the custom OS4 quadrotor was able to regulate attitude, and using a sonar range finder and PID control, was able to achieve a very tight altitude reference tracking [19]. A PIDD^2 controller applied to the Starmac quadrotor yielded a fast angular reference tracking with an RMS error of 0.65° and very tight yaw angle regulation; a square trajectory tracking error of under 10 cm was obtained using an overhead camera [74]. Employing a hybrid backstepping controller to a modified Draganfly quadrotor with a low center of gravity gave a good attitude angle error even in the presence of some disturbance [54]. The heavy X-4 custom quadrotor demonstrated angular regulation outside using PID control [18].

3.10.4 Commercial Quadrotors

Although minimal published information is available for commercial quadrotors, there are several types available that are successfully stabilized and used by research groups, hobbyists, and industry. The Ascending Technologies Hummingbird and Pelican quadrotors, the Mikrokopter, Quansar, and Microdrone are all effectively stabilized systems for general use [6–9, 50].

3.11 Results

Flight quality achieved with the attitude controller on this quadrotor is highly stable, flat and has very good disturbance rejection abilities. It is easily comparable to any of the successful commercially available quadrotors or other research platforms. The figures throughout the chapter of the measurements from the attitude controller during flight indicate the stability of the system and the control methods. Steady state altitude approximate error is within 92 to 104 cm, for ±6 cm. Yaw is regulated to within an approximated ±2.5°. Attitude system gains used to achieve stabilization are shown in Table 3.1.

Table 3.1: Attitude system gains.

Gain Term	Notation	Value
Roll Proportional	$k_{p,\phi}$	4.0
Pitch Proportional	$k_{p,\theta}$	4.0
Yaw Proportional	$k_{p,\psi}$	8.5
Roll Derivative	$\dot{k}_{p,\phi}$	0.65
Pitch Derivative	$\dot{k}_{p,\theta}$	1.0
Yaw Derivative	$\dot{k}_{p,\psi}$	3.5
Altitude Proportional	$k_{p,alt}$	2.9
Altitude Integral	$k_{i,alt}$	0.025
Altitude Derivative	$k_{d,alt}$	1.0
Altitude Nominal	u_{nom}	1505
Nonlinear Complementary Filter Proportional	k_{NLF}	1.0
Nonlinear Complementary Filter Bias	k_b	0.3

3.12 Chapter Contributions

Many attitude control methods have been implemented on quadrotors. The work presented here includes extensive filtering and attitude estimation techniques required for obtaining effective measurements using very low cost sensors and is the first implementation of the nonlinear passive complementary filter on a quadrotor. Traditional control methods are trimmed to only the effective portions based on the limitations of the low-cost components. As a complete package, this chapter presents an effective approach to attitude stabilization using a built-from-scratch custom quadrotor with strict cost limitations.

3.13 Comparison of Attitude Stabilization Results

Direct comparisons of results are always difficult without a common standard. The differences between each implementation are extensive and so checking numerical results does not take into account the differing conditions and methods. However, from a survey of the literature, the attitude stabilization capabilities of the quadrotor presented here is certainly on the same level as the other successful quadrotors, but has the additional aspect of using components that are over an order of magnitude less in cost than other implementations. One of the commonly used IMU sensors, the Microstrain 3DMG-X1, is over 50 times more expensive than the sensors used here [80].

3.14 Chapter Summary

This chapter presented the requirements of attitude stabilization for a custom quadrotor platform. Sensor filtering and attitude estimation requirements were presented for obtaining reasonable attitude measurements. Some of the crucial lessons learned from attempting attitude control with a lower quality sensor were indicated. The control methods for each portion of the quadrotor attitude were covered with an indication of the results during a typical hover flight. Issues that were discovered to be a factor in obtaining attitude stabilization were mentioned. Finally, the chapter closes with an overview of the relevant research in quadrotor attitude control, with chapter contributions and a general comparison of results.

Chapter 4

Navigation Using Camera Vision

A stable and controllable quadrotor is only the pre-requisite for performing a task. This task is determined and executed at a higher level, working with the attitude controller to move the quadrotor according to the desired goal. Such a higher level controller is often referred to as the navigation controller. There are numerous ways and means of designing this upper level controller, including different sensors to use as well as different algorithms for determining position from the sensor so that the quadrotor can track trajectories or paths accordingly.

4.1 Chapter Overview

This chapter covers the basics of quadrotor navigation and some of the methods used, including specific constraints applied to this quadrotor. Following is a discussion of comparable work that has been done in the field and how it applies to this thesis. The physical and software setup of the navigation system is then presented along with the control algorithms used and the setup of the outer loop controller. The results obtained from several types of navigation implementations are shown and finally, the results are briefly compared to currently available research outcomes from other groups.

4.2 Navigation Systems Overview

The most common form of navigation system is through the use of GPS, which gives coordinates directly to the quadrotor for determining its position. Unfortunately, this system is limited to outdoor use only, and so is not able to be used on an indoor operated quadrotor. A similar system for indoor use is what is known as pseudo-GPS. This takes the form of having some other device, typically an overhead camera, that tracks the quadrotor on a host

system and sends the detected position back, similar to how GPS works outside [31]. Limitations of this system is that it can be difficult to obtain good enough accuracy and speed of position detection for indoor use from an overhead camera. In addition, the quadrotor is limited to operation only in the room with the overhead camera. In the past few years, IR camera positioning detection systems have been widely used, the most well known is the Vicon System [78]. Several cameras are placed around the room with IR reflectors on the quadrotor, which actually enable attitude tracking as well as position tracking. These systems are extremely accurate (sub-mm) and fast (100-300 Hz), allowing for very effective control. They are also quite expensive, the minimum being over $10,000. Plus they have the same limitation as the pseudo-GPS system of being confined to one room for operation.

4.2.1 Challenges of Indoor Navigation

In contrast to flying vehicles, ground vehicle indoor navigation can be done using accurate steering based on kinematics, where high frequency estimates of the relative measurements, such as wheel encoders, give a good estimate of the location of the vehicle. In addition, ground vehicles can move at very low speeds as necessary for adjusting to terrain or computational requirements. With flying vehicles, translational speed is much more difficult to control, and thus a constant motion of some minimal amount prevents a full stop to re-evaluate the pose of the vehicle. Odometry can only be obtained indirectly, so dead-reckoning becomes an alternative but with its inherent errors. As per above, GPS is an option outside, as are magnetometers for giving directional heading, but not only are they not available inside, they are also not accurate enough for the required small spaces of indoor flying. An angle error of 1° will yield an estimated 34 cm translation in 2 seconds, which can be neglected in outdoor environments but not when inside. A high quality outdoor navigation system shows the relative scale of the flight envelope during experimentation [81].

Aside from the basic accuracy difficulties of navigation indoors, unstable systems like quadrotors have a constant motion due to aerodynamic flying effects, plus actuator vibrations, makes sensing difficult as there is a constant motion and vibration noise leading to

poor measurement quality.

4.2.2 Localization and Mapping

Localization and mapping refers to the process by which camera images moving over time are pieced together to form a map with pose estimates, and having the features currently in the image matched with the map in order to determine the current location on the map. In the past, maps were created ahead of time using a ground robot or camera carried by a person, and then that map could be used to determine camera pose localization in real time. However, it became possible to perform both mapping and localization on the fly, known as simultaneous localization and mapping (SLAM) [82]. This approach solves a joint estimation problem for determining both the position of the vehicle, and a map of the surroundings.

SLAM

SLAM uses the widely effective Extended Kalman Filter (EKF) for estimating its current pose and updating the map. The states of the EKF are the pose variables of the vehicle, such as x, y, and yaw for a ground robot, and this state vector is augmented with new state information as the robot moves, observing new features within the environment in the form of distinctive landmarks, creating a larger and larger state vector. As part of the EKF, uncertainties in the measurements and estimates are incorporated. The estimated pose uses a motion model, assuming certain motion smoothness and thus restricting the possible locations of the new pose for faster localization. The drawbacks of this approach include the continuously expanding state vector, requiring heavy computation for matrix inversion. Additionally, the linearization of the motion and measurement model as part of the Kalman filter takes a toll on the accuracy.

SFM

A recent localization and mapping alternative method to SLAM is known as Structure From Motion (SFM), which came from computer science. It involves a matching of images

using a similarity of features between them to determine rotation and distance without necessarily having a sequence to the images, and was often originally used post flight to generate a complete map. Technically, it estimates the ego-motion of the camera in addition to reconstructing the 3D structure of a scene, by projecting it onto moving 2D surfaces obtained from images at different locations and angles, resulting in a 3D point cloud of the feature scene. SFM implementations use a variety of methods, differing in many portions of the algorithm, including the motion model, input measurements, time-frame, and data processing techniques. It is essentially solving a large nonlinear optimization problem to match features to the map. It is typically more accurate and yields more detailed maps than SLAM, but is computationally very expensive and slow. In order to compete with SLAM, the requirements of high speed, low image quality and rapidly changing camera attitude make feature tracking based on SFM a difficult problem. Additionally, SFM suffers from difficulties in recovering absolute translational velocities and true distances to perceived objects, known as the scale factor problem.

4.2.3 Indoor Position Sensing for Navigation

Utilizing onboard sensors for determining position allows for more freedom than the systems listed in Section 4.2, but has its own set of challenges. Typically, sensor data processing for positioning estimation is very computationally intensive and so often it becomes necessary to send the sensor data to a ground station system, which does the extensive computation for position estimation and then sends it back to the quadrotor for it to perform the outer loop controller on the positioning data. The sensors used in this type of data collection vary from lasers, stereo cameras, and regular cameras. Each has different advantages and disadvantages.

Laser Sensor

Laser sensors are not vision sensors, but they can be used to simulate or assist in vision navigation by mapping the environment in 3D by giving range information that can be used to determine pose through a scan-matching algorithm. Lasers can detect distance very well,

and so at high speeds and with the use of multiple sensors or mirrors, a 3D map of the room can be obtained. Lasers have difficulty in environments that cannot be estimated as flat walled 2D, however, such as with clutter. Laser sensors are also typically quite expensive.

Stereo Camera

Stereo cameras can give both depth and color. These images can be used to detect similarities and changes in the view from one camera image to another, thus detecting movement. However, the depth is restricted for short distances of roughly several meters. They also have difficulty in featureless environments or night time operations. Stereo cameras are usually much more expensive than regular cameras.

Monocular Camera

Typical cameras give a 2D pixelated image of the colors detected in the range of view, with movement detected the same way as with stereo camera. Monocular cameras have difficulties with environments of similar colors, like stereo cameras, and so featureless environments and night time operation are problematic. In addition, no direct depth measurement is available, so it has to be obtained from another sensor or estimated using software techniques, however they are still usable for long distances.

4.2.4 Optical Flow Review

Optical flow is a constant processing of image data from a camera looking at the apparent velocities of movement from brightness patterns. This can arise from relative motion of the object in the field of vision or the viewer and its use is often motivated by its similarity to insect vision, called bio-inspiration. The optic flow can then be defined as the apparent motion of the image intensities caused by the 2D projection onto a sensor of the relative 3D motion of feature points. Many successful optical flow tracking algorithms use the Lucas Kanade feature tracker [83]. By itself, optic flow is useful for detecting movement, but not for navigation. Navigation is theoretically possible by obtaining position information from the detected velocities, but in practice this is too error-prone to obtain

accurate navigation information over time, and yields a poor or nonexistent depth map [84]. Such a use of optical flow for navigation has been implemented, utilizing extensive filtering and IMU fusion in order to obtain reasonable positioning, and while this works adequately outside for short distances it would be unacceptable indoors except for hovering [85]. The application of optical flow on an ultralight indoor fixed wing aircraft allowed it to fly in an indoor arena by avoiding the walls, but it had no navigation capability [86]. A consumer available quadrotor system, the AR Drone, uses optical flow to maintain a hover position [49]. In general, optical flow is often used for obtaining a better hover of a quadrotor, as described below, but the data is not used for localization or mapping of the environment, and is thus not considered as an option here for navigation by itself.

4.3 Related Work in Vision Navigation Implementation

Navigation on MAVs is a very important task, and there has been a lot of research in this area, from vision experts, aeronautical groups, UAV researchers and specialists in controls. Different methods have been proposed, tackling different aspects of the problem. This section gives a broad overview of the literature in chronological order within the separate subsections. This section is divided into: research regarding the feasibility of vision use on a quadrotor, implementations that demonstrated drift-free hovers, navigation systems that use a 3D tracking system, navigation that is demonstrated within a controlled environment, navigation performed using a pre-generated map, and finally navigation performed in a previously unknown environment.

4.3.1 Camera Usage Feasibility on a Quadrotor

A structure from motion algorithm based on an improved Lucas Kanade feature tracker and IMU and GPS fusion was implemented on a small traditional helicopter with reasonable results compared to GPS, however the feedback loop was not closed around the position sensor and a laser sensor was used to map the environment separately [87]. A traditional helicopter was outfitted with a camera with a blob-detection algorithm processed on board and decent real time results, however closed loop control was not implemented [88]. A

downward facing low quality monocular camera was used successfully for localization of a self-stabilizing blimp and axial helicopter, using offboard processing and some assumptions about the scale of the feature scene [89].

4.3.2 Hover Capable Implementations

A ground and on-board camera system was implemented on a tethered quadrotor using blob detection with off-board computing, and showed reasonable results for restricted hover conditions [68]. One of the first applications of a camera on a quadrotor for position control involved the use of specific artificial markers seen by the downward facing camera with all processing done at a ground station, and was limited to hover conditions only and with no mapping involved [90]. Position estimation using a single camera and a pre-defined blob feature scene enabled another hover implementation [67]. One of the first onboard closed-loop vision controllers for a quadrotor was implemented on an FPGA using Harris corner detection for optical flow measurements with some success for a hover [91]. An image-based visual servoing approach to a quadrotor using a downward pointing camera and offboard computing for centroid tracking allowed a successful hover restricted design [60]. Another image-based visual servoing approach compared several controller designs for an effective hover, including consideration of the altitude [92]. A quadrotor equipped with a camera used Lukas Kanade optical flow tracking and integrating over space and time the feature movements using three Kalman filters in real time and offboard computation to determine the relative position and velocity, fusing with IMU data to account for rotational effects allowed a good hover inside, as well as outdoor target and trajectory tracking, although position estimates grew over time [93, 94]. A novel 8-rotor helicopter, designed such that the attitude stabilization and navigation components are decoupled, was able to hover effectively using optical flow [95].

4.3.3 Navigation Using a 3D Tracking System

Highly accurate hover, position and way-point tracking was first demonstrated using the extremely accurate external motion vision system, Vicon, using simple RC quadrotors [79].

4.3.4 Within Controlled Environments

An all on-board implementation of wide field integrated optical flow from a fast processed camera set up to observe a 360° field of view (FOV) allowed for a restricted corridor path implementation [96]. Using monocular vision SLAM for low resolution 3D mapping offboard, a quadrotor was able to perform drift free hover and execute a path through obstacles within a controlled environment, using the Vicon system to simulate IMU measurements [97]. A wall collision avoidance implementation on a quadrotor using a depth map based on optical flow and IMU fusion indicated safe maneuvering through a textured corridor, but was only tested with manual control [98]. Using the ARToolkit open source library, a complete onboard system was able to effectively generate a map and perform a hover using artificial markers on the ground [99].

4.3.5 Localization with Pre-generated Maps

A quadrotor performed successful localization over a variance-reducing path algorithm execution using a laser range finder operating on an existing map [100]. Using a ground robot to generate a map, a quadrotor equipped with a laser and using SLAM and Monte Carlo Localization was able to localize itself on the pre-generated map as well as develop its own map which closely matched the map generated by the ground robot [13]. With a self-stabilizing coaxial helicopter and only a camera for sensing, localization, and navigation was possible using optical flow for assisted attitude stabilization and localization for global positioning on a map generated previously by a ground robot [101].

4.3.6 Navigation in Unknown Environments

The winner of the International Aerial Robotics Competition Mission 5 in 2009 was a quadrotor that utilized a laser range finder and two stereo cameras to generate a map and localize itself using multiple computers for offboard laser and vision processing involving a FAST feature detection algorithm, the OpenCV KLT optical flow tracker, an EKF, and a slow processing SLAM for map quality [102–104]. Onboard implementation of a modified monocular SLAM algorithm on a traditional helicopter made mapping and navigation in

an unknown indoor environment possible, with the assumption that the environment can be approximated as made up of corner-like features and straight architectural lines [105]. Using a sophisticated offboard SLAM algorithm and an onboard downward facing camera, a quadrotor was able to accurately hover and navigate indoors [15].

In contrast to the approaches of the implementations above, the system described here uses a custom quadrotor with very low cost attitude and vision sensors, capable of mapping and localization in an indoor unstructured environment using navigation techniques that are computationally reduced enough to be capable of being implemented on-board the quadrotor with available technology.

4.4 Camera and Image Acquisition

A camera was chosen for this platform because of the advantages of cameras of being lightweight, relatively cheap, low-power consumption, very rich information, and capable of dual-use, such as for detecting specific objects in addition to navigation sensing.

This platform utilizes a low-cost web camera, the QuickCam Logitech Pro5000, since it is well known to work with the navigation software that is used [106]. The camera is modified by replacing the standard lens with a wide angle view lens of 2.1 mm, with an 81° FOV. This improves robustness by letting more of the physical environment to be seen, and thus have more features in view and as well as more of the features within the map. This web camera is a typical USB camera; the cable is tied to the quadrotor so that during flight it hangs down and is plugged into the ground computer through an extension cord.

4.4.1 Wired Camera Considerations

A wired camera was chosen due to cost and complexity concerns. As a prototype system, only the necessary and new aspects needed to be accounted for. As per above, onboard processing of vision algorithms has been accomplished and recent developments in technology means it is even easier to have on board computing - the quadrotor is quite capable of handling the space and weight required. The wired camera could have been

attached to a wireless USB device, but this would yield greater delays in the system for gaining a quality that is not a demonstration of anything very novel.

4.4.2 Camera Calibration

Before using a camera to estimate position, and thus distance, from a camera image, the inner camera processing of the image into pixels must first be calibrated. This calibration process, also known as resectioning, calculates the camera matrix that contains the parameters of the camera that produce the actual image. The matrix being determined for this process includes the effects from parameters such as the focal length of the camera, image center, skew, image format, and the principal point. Equation (4.1) shows the calibration matrix values found for the camera used.

$$\begin{bmatrix} \alpha_x & \gamma & u_0 \\ 0 & \alpha_y & \nu_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.5547 & 0.7381 & 0.5536 \\ 0 & 0.5534 & 0.9713 \\ 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

Prior to running the calibration, the camera focus is adjusted manually, which is simpler than utilizing gradient-based programs to obtain the correct focus while still being quite accurate.

From this knowledge of the camera matrix, mapping of the pixels to 3D space can be performed by moving the camera and taking images over a known fixed distance. Implied from this is that the 3D space accuracy from the camera depends directly on the quality of the determined camera matrix. A problem noticed with the navigation system stability can sometimes be directly tied to the software system's inability to deal with the errors it is seeing from a poorly calibrated camera matrix.

4.5 PTAM Algorithm for SLAM

PTAM, or Parallel Tracking and Mapping, is a software suit developed by Klein and Murray [107, 108]. It was chosen since it is highly capable of mapping and tracking in unknown environments, is open for use, and is effective and robust, which is uncommon for

most software developed through research and is necessary for implementation on a system with fast, unstable dynamics. This algorithm was previously implemented successfully by Blösch et al. on a commercial quadrotor equipped with a uEye global shutter camera [15].

4.5.1 Advantages of the PTAM Implementation

Although monocular vision SLAM is highly capable of map generation and localization on the fly, the implementations have robustness problems due to the motion model behind the system. Tracking systems typically rely on a prior pose estimate over the current one, using the prior one to limit the search for visual feature equivalence for rapid localization. Rapid camera motions, or jerks, camera occlusion and blurring from motion can cause the tracker to be unable to find the true location due to the violation of the motion assumption limitations on searching. A complete data driven detection of the pose, without a basis on the previous pose, as in SFM, will yield much more robustness since the pose is re-estimated from scratch each frame; however, this can be computationally intensive, requires a map a priori, and smoothness and accuracy are still not equivalent to SLAM algorithms [109].

Implementing a combination of these approaches enables fast and effective mapping with very robust tracking. A combination of the motion model for pose estimation when it yields high accuracy for reduced computation, fast mapping, and increased smoothness; and data driven localization when tracking from estimated motion fails. PTAM does not use EKF-based state estimation or uncertainty incorporation that SLAM uses, vastly reducing the computational effort involved in inverting larger and larger matrices to obtain the current state. Instead, the elimination of modeling uncertainties is replaced with the large amount of extracted features and refinement using local and global batch optimization.

Although trade-offs are involved with such an implementation regarding efficient and accurate mapping and tracking, PTAM is very advantageous in this situation for two reasons: the robustness of the implementation and the real-time operation. As a software package, PTAM is well designed for camera motions seen on a flying vehicle and is implemented in a way to obtain constant operation and success. Since PTAM was originally designed with augmented reality in mind, it is set up to perform in real time, which is

crucial for navigation control on a flying vehicle.

4.5.2 PTAM Operation

As the name suggests, PTAM uses two separately-scheduled threads: one thread to create a map based on key frames, and a tracking thread to determine where the camera image is located on the map. For best operation, a dual-core processor is needed. Separating these two processes means that tracking is not probabilistically tied to the map-making procedure as in SLAM and so any robust tracking method can be used [82, 107]. Updating the map only based on certain criteria, instead of every frame, means the tracking thread can perform more thorough image processing, improving accuracy and robustness.

The map consists of keyframes and 3D feature points, with information added to the map being optimized through bundle adjustment. Since many video frames contain redundant information, as the camera is not always moving a lot, incremental mapping of each camera image can be reduced to processing a smaller number of more useful keyframes, which are selected using an heuristic criteria. This gives more time to update the map with the new keyframe, allowing for a larger map size of many keyframes and accurate keyframe addition using the computationally expensive but highly accurate batch method of bundle adjustment. In addition, features can be re-visited and refined within the map with occasional full-map optimizations. Bundle adjustment solves a nonlinear least squares optimization problem where the objective function is the reprojection error. The reprojection error is defined as the difference between where a feature is observed and where it is expected to be observed when projected onto the camera frame. This optimization is solved using the Levenberg-Marquardt Algorithm, which takes into account not only the gradient but also the curvature. The intuition behind the algorithm is opposite to that of gradient descent, in that larger steps are taken when the gradient is small.

Tracking is based on keyframe localization, with an effective recovery system implemented by performing a relocalization by training a Features from Accelerated Segment Test (FAST) classifier using down-sampled versions of the map keyframes, allowing for quick detection of the map keyframes in view without excessive processing requirements - a coarse

to fine approach using a robust estimator; it is based on an implementation using feature points of small maps [109]. Tracking quality is measured based on quantity of successful feature matching, and only a certain threshold allows for the map to update keyframes, preventing corruption of the map with poor information.

These separate simultaneous threads for mapping and tracking operate in the following major ways:

Tracking thread sequential operation:

- A new frame is acquired from the camera (640 X 480, at 30 Hz, and converted to 8-bits per pixel (bpp) gray scale for tracking and red-green-blue (RGB) for display).
- A four-level image pyramid is constructed of coarse to fine resolution versions of the image, and FAST corner detection is applied to each pyramid level.
- A prior pose estimate is generated from a motion model.
- Map points are projected into the image according to the frame's prior pose estimate.
- For each single map point, a fixed-range search is performed around the point's predicted location in the image.
- Small numbers (50) of the coarsest-scale features are searched for in the image.
- The camera pose is updated from these coarse matches.
- Larger numbers (1000) of points are re-projected and searched for in the image.
- A final estimated pose for the frame is computed from all the matches found, by minimizing a robust objective function of the re-projection error.
- Tracking quality is estimated at each frame based on the fraction of feature observations that are successful; below a certain threshold, tracking quality is considered poor and while tracking continues, the system is not allowed to send new keyframes to the map.

- Tracking is considered lost for an even lower threshold, at which point a recovery process is initiated.

The mapping process:

Prior to actual continuous keyframe mapping, an initial map must be created using a stereo technique based on a 5-point stereo algorithm. A keyframe is added to the map at the location desired, after which a translation of 10 cm adds a second keyframe; feature correspondences between the two keyframes allows for an estimate of the essential matrix and triangulate the base map using the algorithm, with depth estimation. After this, mapping runs asynchronously in an endless loop as it receives new frames from the tracking thread.

- As the camera moves away from its initial pose, new keyframes and features are added, making the map grow.
- The keyframe addition criteria is:
  - Tracking quality must be good,
  - Time since last keyframe added must be greater than 20 frames,
  - Camera must be a minimum distance away from the last keyframe.
- Restricted keyframe additions prevent wasteful computation while hovering (unlike for Kalman filter SLAM).
- Bundle adjustment iteratively adjusts the map in order to minimize a robust objective function based on the Levenberg-Marquardt bundle adjustment algorithm.
- In the case of convergence of bungle adjustment and no new keyframes are needed due to being in an already known area of the map, the mapping thread improves the map by making new measurements of old keyframes.

An example of the map generated is shown in Figure 4.1(a) for the 3D map of the feature scene, and Figure 4.1(b) for the physical map overlaid with the feature scene. Note

that the patterns of features are used simply for convenience due to the fact that the testing floor looks all white, so there are no features for the camera to see. Features the algorithm can detect are not limited to those shown, and the quadrotor has been successfully flown in various environments, including over children's play rugs depicting city scenes.

4.5.3 Application Specific Modifications

For use with the quadrotor, several modifications had to be performed for proper control. The PTAM code is modified to only send localization information to the quadrotor if the detected image is determined from the algorithm to be of high confidence, meaning that it sees a certain percentage of features that match with the internal map. When the camera is roughly stationary, as in the hover, some of the tracking aspects are turned off to reduce computation time.

The localization information is sent in a packet format, containing a packet type flag for contrast to the manual controller packet, plus the x, y, z position information. The packet also includes the estimated yaw angle from the software.

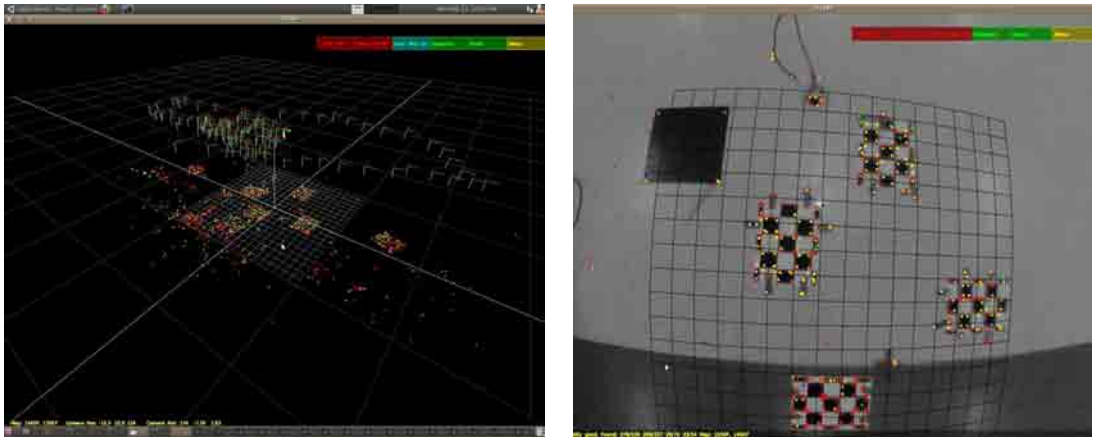
Implementation Specifics

FAST feature detection is utilized, and a standard off-the-shelf bundle adjustment library. Stereo initialization is done by hand, although automation of this is possible. Recovery mode is used while in the air as necessary (tracking quality is below the minimum threshold). No explicit loop closures are implemented, but such has not been discovered to be an issue.

Because the camera is occluded by the floor when the quadrotor is landed, the take-off is only possible if the area the quadrotor rises above is already stored in the map, allowing PTAM to localize itself. Thus, enough thrust must be given on starting in order to reach to near where the initial keyframes were added, allowing proper recovery.

Observations and Issues

Because of the unobservability of scale offered by a monocular camera, the map scale



(a) 3D map of detected features and keyframes. (b) Map of camera view with detected features.

Fig. 4.1: Images of PTAM detected features.

is estimated by hand with each PTAM initialization. No stability problems were observed arising from the scale and orientation drift of the map. In any case, realigning the map in flight was not considered possible due to the limitations of the stability of the quadrotor itself [15]. Over time, the only problem was the corruption of the map with wrong feature or keyframe data, primarily from leaving PTAM running while changing the battery and other such movements that are not within the mapped area.

Use of higher resolution is not desirable in order to keep up with real-time computation. There is some blurring due to very fast motion, which could be avoided with edge feature detection, but an attempt at this did not work.

In order to obtain efficient navigation, several problems had to be discovered and overcome. Some hardware problems included such issues as running the camera on USB 1.1, which is not effective enough for PTAM; also, when using the more recent Ubuntu version, an OpenCV driver had to be used, with the appropriate linking modifications.

Some problems with the mapping and tracking were discovered to be mostly due to poor camera calibration. In such cases, PTAM would not always recover into tracking itself on the map after being on the ground for some time and only building a more extensive starting location map, about a 1 m square, with enough keyframes would prevent this. A better camera calibration prevented such necessities. However, even with the better camera

calibration, using bigger features improved the robustness of the system; flying at a height of over a meter, with a low resolution camera, makes very small features hard to detect.

PTAM is known to work correctly on Nvidia graphics cards, requiring adjustments for other systems. Fortunately, the ground station computer available has an Nvidia graphics card.

4.5.4 Latency

Understanding computational-based latencies is important for maintaining stable navigation and implementing appropriate controllers. The mean tracking time for a reasonably large map of 1100 feature points is 8 ms. The mean time for camera image acquisition within PTAM is 31 ms, which yields a frame-rate throughput of about 32 frames per second.

4.5.5 Computational Intensity

A dual core processing system is used for the ground station computer. When PTAM is running, the system task manager indicates a CPU 1 usage of 60% for the tracking thread, and CPU 2 is nominally 6-7% but will rise to 80-85% for short times while the Levenberg-Marquardt optimization for mapping is performed. These measurements were taken while as few additional processes were running as possible.

4.6 Navigation Control Using Vision

The navigation controller utilizes PID control within a portion of the linearized model of the quadrotor dynamics. The navigation controller is an upper level controller, above the attitude controller; an outer loop of the attitude stabilization control block diagram. This outer loop controls the positioning of the quadrotor by relaying a reference command to the inner attitude controller and operates in two fundamental modes: hover around a specified position or tracking a sequence of predetermined way-point positions. Manual control works similar to an outer navigation loop, but is in place of the navigation (the navigation system will continue to operate, updating its measurements, but no commands will be passed on to the attitude controller).

In the hover mode, the goal of the quadrotor is to obtain the desired position and maintain as minimal a distance to it as possible. In the path tracking mode, the goal of the quadrotor is to move to the next way-point in a controllable manner.

4.6.1 Sensor Coordinate Frame

The axes definitions of the prototyped quadrotor differ from traditional aero convention due to the mounting of the sensor. Although this could be adjusted in software, development of the control system was completed before adapting the proper convention, and so for consistency, all implementation in this thesis uses the axes definitions shown in Figure 4.2, which uses Figure 3.5 as its base for roll and pitch definitions.

4.6.2 Control System Block Diagram

The navigation controller is at the outer loop, layered around the inner attitude controller. The general setup of the control system implementation of the complete navigation with attitude flight controller is shown in Figure 4.3.

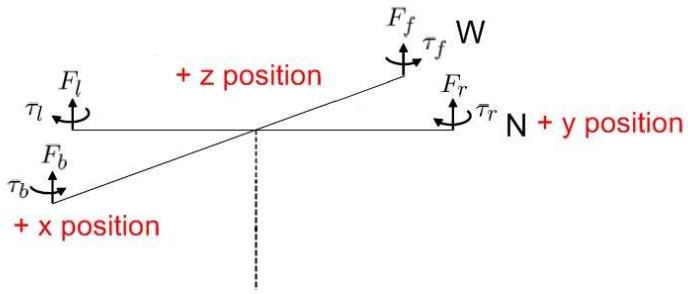


Fig. 4.2: Navigation system axes definition. Note that this differs from the standard aero convention discussed in Section 2.1 due to application specifics, so readers will be required to associate the system aspects in this thesis using this frame setup.

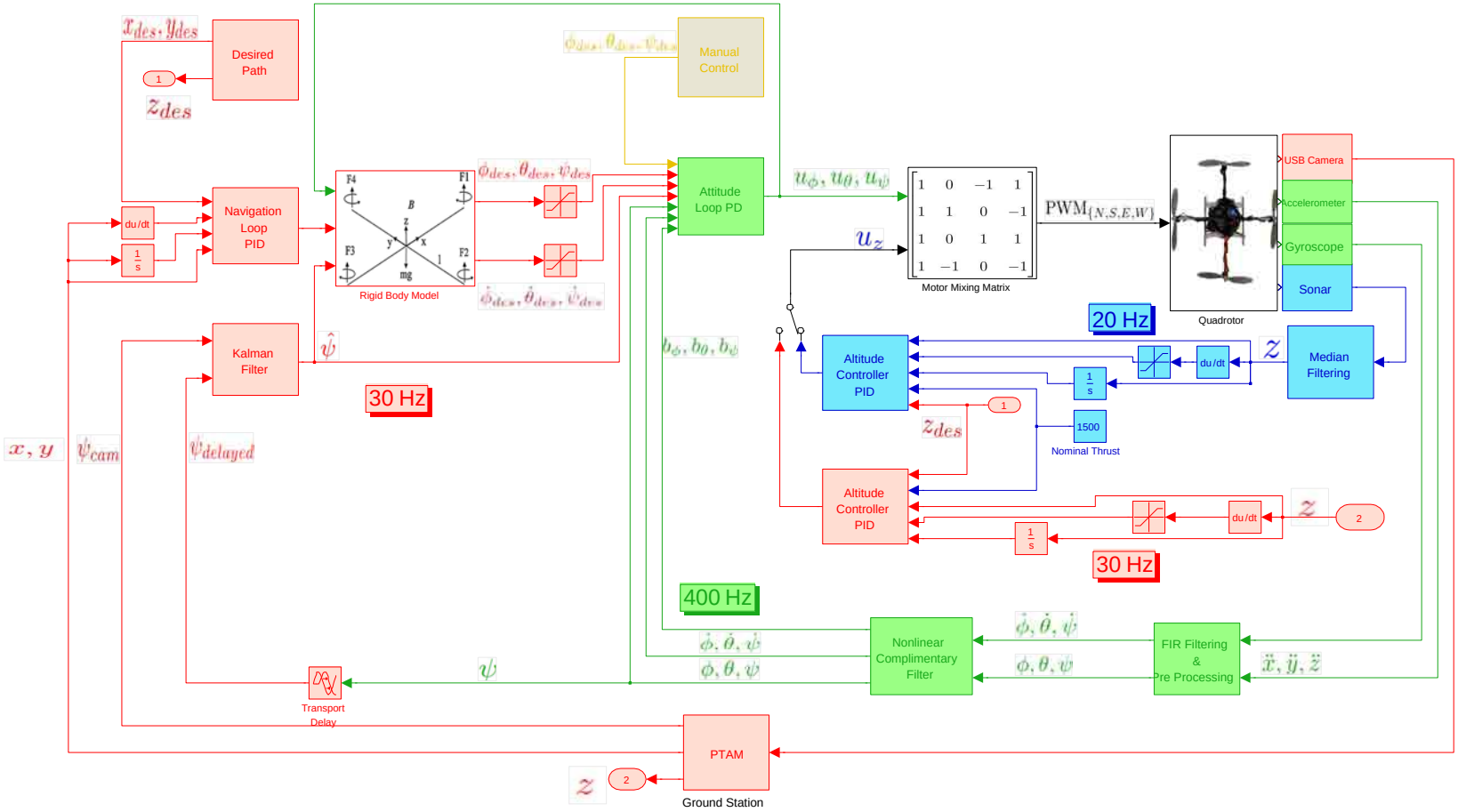


Fig. 4.3: Control system block diagram of nested navigation and attitude controllers. The green blocks and wires indicate the attitude control system; the blue blocks and wires are for the sonar altitude controller; red indicates the navigation system utilizing the SLAM algorithm on the ground station, which returns position information. A backup manual controller for safety is shown in a tan color.

4.6.3 Control Model

The navigation system controller utilizes PID control based upon the x and y positions and the translational velocities with

$$\begin{aligned} u_{nav,x} &= k^x_{p,nav} \cdot (x^{des}_{pos} - x_{pos}) + k^x_{i,nav} \cdot (\int_0^t x^{des}_{pos} - x_{pos})dt \\ &\quad + k^x_{d,nav} \cdot (\dot{x}^{des}_{pos} - \dot{x}_{pos}), \qquad (4.2) \\ u_{nav,y} &= k^y_{p,nav} \cdot (y^{des}_{pos} - y_{pos}) + k^y_{i,nav} \cdot (\int_0^t y^{des}_{pos} - y_{pos})dt \\ &\quad + k^y_{d,nav} \cdot (\dot{y}^{des}_{pos} - \dot{y}_{pos}), \qquad (4.3) \end{aligned}$$

where (des) refers to the desired reference and pos refers to the position.

4.6.4 Control Methods - Analysis and Related Work

The navigation controller is implemented separately from the attitude controller and so can have different requirements, such that a different controller might perform better. PID navigation control has been implemented with good results for path tracking [31, 101]. Backstepping has been applied to quadrotor positioning system done in simulation [19]. A positioning controller implemented with a nonlinear controller using backstepping has also shown good results [60]. Nonlinear nested control methods based on system modeled dynamics has also been shown to provide tracking [81]. PID control was ultimately chosen due to its simplicity and ability to handle un-modeled effects, with the outer-loop treating the inner system as a second-order system, modeled as a point mass that is stabilized to the point that direct commands can be given without need to model other effects.

In the navigation implementation of the hover and path tracking, the fundamental difference is in how the controller is applied to the desired path. A similar approach has been used where the path aspects of within path and horizontally normal to the path positions and velocities are explicitly defined and controlled independently [74]. In such an approach, a PID controller is used to keep the heading from moving out of the path while desired velocity and integrated velocity components (PD control) are used to regulate the

speed and direction of tracking. This is because using a generic navigation PID controller incorporates both in track and out of track errors into the integrator, which will accumulate inapplicable error in the direction of the path, making it difficult to regulate the speed or account for changes in direction. In addition, the derivative controller is only given one desired velocity, the along path velocity, but it is apparent that the desired velocity away from the path should be zero. The path-dependent controller works to reduce the distance away from the desired position and minimize the velocity that is away from the path as well as move at the desired velocity in the desired direction.

Velocity Determination

The camera only gives position estimates, and so finding the translational velocity is not straightforward. Differentiating a non-ideal signal is always a concern due to the noise that is often inseparable from the original signal. A small amount of noise which is negligible when used as the original signal, will get excessively amplified due to a small change over a very small time sampled step. Initially, the velocity was determined from integrating the accelerometers, which is useful as a separate sensing source. The integrated x and y accelerometers were used in order to obtain translational velocities per

$$\dot{x} = \int_0^t (\ddot{x})dt, \tag{4.4}$$

$$\dot{y} = \int_0^t (\ddot{y})dt, \tag{4.5}$$

where $\ddot{x}, \ddot{y}$ are obtained directly from the accelerometers. This yielded terrible results, such that the measurements were not even useful enough to fuse with the velocity calculated from differentiating the position, due to the noise and inaccuracies in the accelerometer measurements, similar to the z-axis altitude situation described in Chapter 3 and for the same reasons. In the end, velocity is simply calculated by finite differentiation, taking the current position value, subtracting the previous position, and dividing that all by the time

between position updates, so that

$$\dot{x} = \left(\frac{x_{curr} - x_{prev}}{dt} \right), \tag{4.6}$$

$$\dot{y} = \left(\frac{y_{curr} - y_{prev}}{dt} \right). \tag{4.7}$$

However, gains could not be made very high on the derivative term due to the rapid changes from measured signal differentiation. Filtering was not considered an option due to the already delayed values from the PTAM algorithm. A comparison between the two methods of calculating the velocity can be seen in Figure 4.4.

The sampling time then becomes a critical factor, as too frequently run loops will recalculate the derivative using the same position, and thus yield a zero velocity, which is not true. However, running too slow will lose information. The controller operates asynchronously, acting only upon receiving a packet from the ground station. This means actual times between packets are not known accurately. A simple solution that has not had issues is to just use the average frequency at which the navigation system (PTAM, camera, Zigbee) sends package updates.

Integrated Position

Since the position measurement is quite accurate and with fairly low noise, unlike the attitude angular measurements, effective use of an integrator control term can be utilized. Although the quadrotor, as an inherently unstable nonlinear system, will never be able to perfectly go to the desired position and maintain there at zero error, a small integrator term can account for small offsets due to in-flight effects such as actuator friction buildup. As expected, an integrator will always be trying to catch up, essentially adding a delayed proportional term to the system, which can easily lead to unstable results if the integrator term is too large. However, this does not necessarily preclude the use of a small integrator term to give the system the extra control margin it needs to correct a more constant offset, and such uses have been successful for other implementations [74]. In this case, utilizing a small integrator in the navigation control noticeably improved the position error.

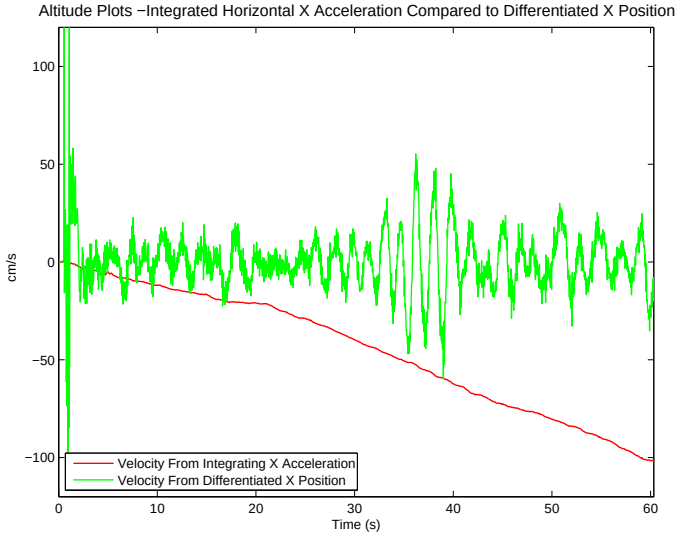


Fig. 4.4: Navigation velocity calculation method comparison - integrated x acceleration vs differentiated x position.

4.6.5 Feed Forward Velocity

Similar to using the acceleration as a feed forward term in attitude stabilization, the idea of using acceleration within the outer control loop for anticipating translational movement has been used [38]. However, similarly to the attitude difficulties, and for the same reasons, using the accelerometer measurements to anticipate translational velocities simply led to noise in the system and visible instability.

4.6.6 Direct Actuation for Disturbance-Based Control

Initially for ease of coding, the navigation control was implemented as a disturbance, directly applying thrusts to each motor according to a PD control, similar to how the manual controller was first set up as discussed in Chapter 3 but with only roll and pitch affects, so that

$$\begin{bmatrix} u^1_{\text{PWM}} \\ u^2_{\text{PWM}} \\ u^3_{\text{PWM}} \\ u^4_{\text{PWM}} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -1 & 1 \\ 1 & 1 & 0 & -1 \\ 1 & 0 & 1 & 1 \\ 1 & -1 & 0 & -1 \end{bmatrix} \left(\begin{bmatrix} u_{alt} \\ u_\phi \\ u_\theta \\ u_\psi \end{bmatrix} + \begin{bmatrix} 0 \\ u_{nav,x} \\ u_{man,y} \\ 0 \end{bmatrix} \right). \tag{4.8}$$

This lead to reasonable results in terms of maintaining a fixed distance, although not

quite as good as desired angle control, but the flight quality of the quadrotor was much more jerky. As such, this method was simply an intermediary method between just attitude control and a more model-based navigation control.

4.6.7 Reference Angle Command for Simple Model-Based Control

Initially a simple angle command was given straight from the PID navigation controller [19]. This means that the output from the PID controller per Equation (4.2), $u_{nav,x}$ and $u_{nav,y}$ are interpreted directly as angle commands by the attitude controller for both roll and pitch with

$$\begin{bmatrix} \phi_{des} \\ \theta_{des} \end{bmatrix} = \begin{bmatrix} u_{nav,x} \\ u_{nav,y} \end{bmatrix}. \tag{4.9}$$

This type of control enabled a stable hover, however the error bound was rather large, on the order of a couple meters, due to the fact that a slight yaw and a desire to just move left would actually yield a movement in both forward and left directions, with the controller unable to compensate for the errors, leading to marginally stable oscillations around the desired hover location. Modeling this yaw effect much improved the quality of flight.

4.6.8 Yaw Compensated Control

A significant improvement to model-independent control is accounting for the yaw of the quadrotor. If the yaw is not taken care of when giving roll and pitch desired angle commands to the attitude controller, than the quadrotor will actually move to the wrong location, yielding a hover path that looks much like a circle. For example, if the yaw were 45 degrees, but the quadrotor acted like it was at 0 degrees, then when the quadrotor simply wanted to go left, the roll command would end up taking it both left and back.

This yaw compensated control utilized an adjustment to the way the PID output is set. Instead of getting angle reference commands from the PID controller, the output is

treated as desired accelerations such that

$$\begin{aligned}\ddot{x}^{des} &= k^x_{p,nav} \cdot (x^{des}_{pos} - x_{pos}) + k^x_{i,nav} \cdot (\int_0^t x^{des}_{pos} - x_{pos})dt \\ &\quad + k^x_{d,nav} \cdot (\dot{x}^{des}_{pos} - \dot{x}_{pos}), \qquad (4.10)\\ \ddot{y}^{des} &= k^y_{p,nav} \cdot (y^{des}_{pos} - y_{pos}) + k^y_{i,nav} \cdot (\int_0^t y^{des}_{pos} - y_{pos})dt \\ &\quad + k^y_{d,nav} \cdot (\dot{y}^{des}_{pos} - \dot{y}_{pos}), \qquad (4.11)\end{aligned}$$

through linearization of Equation (2.2) about the hover region for the acceleration of the center of mass in the inertial frame [76]. Then these accelerations are used for calculating the net reference angle based on the yaw. The yaw model simply accounts for the angle at which the quadrotor is yawed before calculating the desired roll and pitch. This is modeled per

$$\begin{bmatrix}\phi^{des} \\ \theta^{des}\end{bmatrix} = \frac{1}{g}\begin{bmatrix}-\sin\psi & -\cos\psi \\ \cos\psi & -\sin\psi\end{bmatrix}\begin{bmatrix}\ddot{x}^{des} \\ \ddot{y}^{des}\end{bmatrix}, \qquad (4.12)$$

with g representing gravitational acceleration.

4.6.9 Kalman Filter Fusion of Yaw

On the quadrotor there are two measurement sources for yaw, the yaw angle from integrating the yaw gyro, and the camera itself. To provide an optimal estimate of the actual yaw, these two sources are fused using a Kalman filter, with the measurement update occurring with the slower measurement (the camera) and the process update occurring during the attitude control loop per

$$\hat{\psi}_{k|k-1} = \psi^{\text{IMU}}_{\text{delayed},k-1|k-1} , \tag{4.13}$$

$$P^{\psi}_{k|k-1} = P^{\psi}_{k-1|k-1} + Q^{\psi}_{k} , \tag{4.14}$$

$$\tilde{\psi}_{k} = \psi^{nav}_{k} - \hat{\psi}_{k|k-1} , \tag{4.15}$$

$$S^{\psi}_{k} = P^{\psi}_{k|k-1} + R^{\psi}_{k} , \tag{4.16}$$

$$K^{\psi}_{k} = \frac{P^{\psi}_{k|k-1}}{S^{\psi}_{k}} , \tag{4.17}$$

$$\hat{\psi}_{k|k} = \hat{\psi}_{k|k-1} + K^{\psi}_{k}\tilde{\psi}_{k} , \tag{4.18}$$

$$P^{\psi}_{k|k} = (1 - K^{\psi}_{k})P^{\psi}_{k|k-1} , \tag{4.19}$$

where the noise variance for the camera is set as $R^{\psi} = 0.08$ and for the angles integrated from the gyro is $Q^{\psi} = 0.1$ [57]. A flight data example of the fused yaw measurements is shown in Figure 4.5 for a hover. The differences between the two are caused by slightly more than just the fact that they are from different measurement methods. Because the two systems are decoupled, the zero set-point measurements of the two sensors can be different. The yaw from the attitude gyro will consider the placement in which it started the flight to be zero degrees because of the initial calibration; the camera, however, will consider the zero yaw position to be the placement in which the map was initialized. These two such instances will not exactly coincided, due to the human involvement factor in the placement of the quadrotor for flight and in the map initialization. Using the fused yaw as the reference, the yaw is able to be regulated to within an approximated $\pm 2^{\circ}$.

Attitude Yaw Buffer

Using the yaw determined from the attitude controller is not simply a matter of using the value from the attitude controller when the navigation loop runs after getting an updated packet. This is because the navigation has delays of processing and communication that mean the current yaw is not going to be from the same moment as the yaw from the

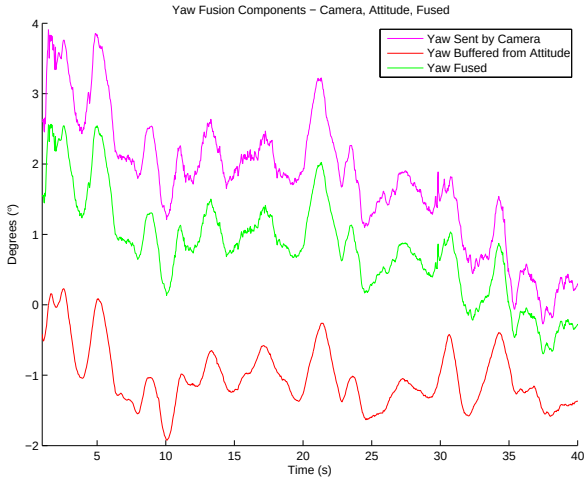


Fig. 4.5: Kalman filter fusion of yaw during a hover.

camera, and thus requires using an older yaw value from the attitude controller. In order to accomplish this, a simple buffer is kept of an appropriate length according to the delay of the navigation packet from the camera and processed through PTAM so that the yaw value picked out of the buffer matches the same camera data, so

$$y_\psi[n] = \psi[n - d], \tag{4.20}$$

where $y_\psi[n]$ is the delayed yaw sent to the navigation system, and d is the buffer delay, which is set to 22, to equal the approximate 55 ms delay for the navigation data. A comparison of the yaw given from the buffer and the true yaw at that point is shown in Figure 4.6.

4.6.10 Fused Yaw for Attitude Control

Since the attitude yaw is only calculated from the gyro with no external sensor, it suffers from gyro drift, integration drift, and general accuracy limitations. The camera gives effective yaw measurements, and so can be used to improve the attitude controlled yaw, in addition to performing improved navigation to a point based on the currently measured yaw. The Kalman filter fused yaw is therefore also directly used in the attitude controller for the yaw, improving the yaw regulation. In addition, since long flights have been noticed to cause large disparities between the gyro calculated yaw and the camera,

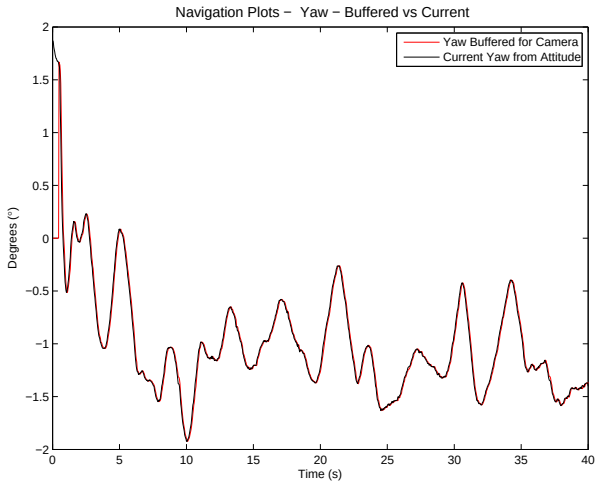


Fig. 4.6: Yaw comparison: delayed buffer vs current yaw at time of navigation control.

with the camera being closer to the truth, if the navigation system turns off, the attitude yaw is reset to the Kalman filter fused yaw.

4.6.11 Gain Tuning

In a PID controller, gain tuning is always striving for a proper balance between the components. One of the issues compounding this tuning is the inability of the accelerometers, and hence the attitude estimation to accurately measure small angles or small changes. This means that the gains of the navigation system have to be large enough to push the attitude controller to the desired value, which often leads to overshooting the actual desired angle. This is considered an improvement over smaller gains, since then the quadrotor will just not keep its position very accurately.

Attempts were made to adjust the attitude gains in order to respond more to the commands from the navigation loop, but it just lead to poor stability. It seems the attitude is independent enough from the operation of the navigation system that it can be tuned separately.

As a single integrated system, issues with one part of the system will couple onto other parts. This was noticed when tuning the altitude controller at the same time as the navigation controller. When the height would have large oscillations due to improper gain

tuning, the navigation would be much worse as well.

4.6.12 Testing Space

Since only the ground station computer has the necessary hardware to run PTAM, all testing was done within a 20-foot diameter circle.

4.6.13 Robustness

The navigation controller is quite robust to delays, determined when it was discovered that a bug in the packet parsing code was causing up to 33% of the packets to be discarded. Even with this packet loss, which amounts to a position update delay up to 60 ms, the quadrotor was capable of maintaining level flight with reasonable position accuracy on the order of a circle with 60 cm diameter.

For a further comparison of robustness of the PTAM software itself, it was compared visually with the "lk demo" optical flow program from OpenCV, which lost feature detection with just a little bit of camera movement, while PTAM is able to handle significant speeds and jerks per the figures below [102].

4.7 Timing Analysis

System latencies and delays can contribute to instability and poor performance. A full timing analysis was done in order to understand what delays impact the system and what can be reduced or adjusted for. A diagram of the system delays is shown in Figure 4.7. In order to take full advantage of the available bandwidth of the navigation system, the navigation controller operates asynchronously, whenever a packet is received from the ground station.

4.8 Vision-Based Hover

As described in Section 4.6, the navigation controller has a special hover-based mode. In this mode, the controller algorithm remains the same, however the desired velocity is set to zero. This makes intuitive sense, as the goal in a hover is to maintain a single position

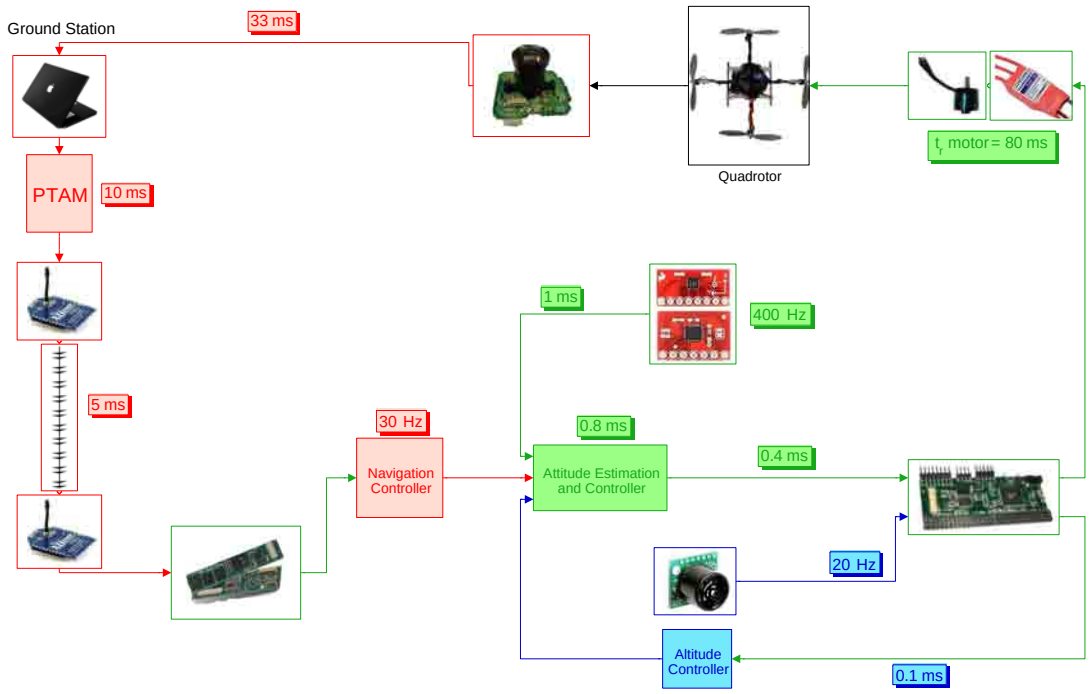
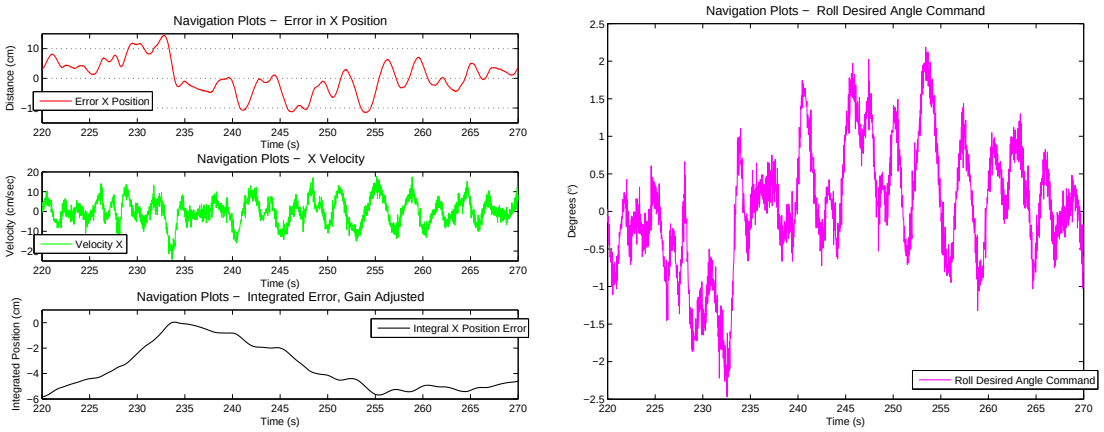


Fig. 4.7: Navigation system latency diagram by component and communication.

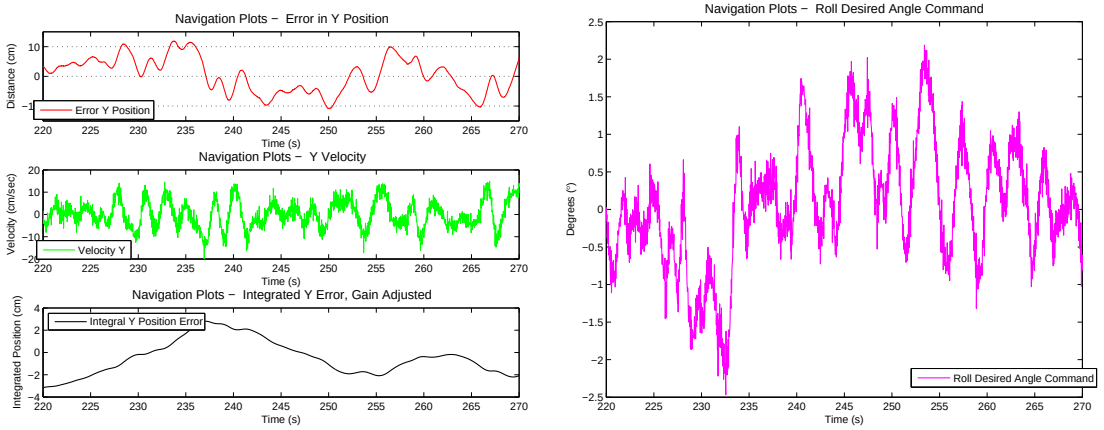
and not deviate from it, much like the attitude controller tries to maintain level flight, with the angular velocity term driving the system to slow to a stop. In such a controller, the proportional distance term will drive the quadrotor to the desired position, while the derivative velocity term will work to prevent the quadrotor from moving fast, or moving from its nominal position.

Care needs to be taken when tuning the gains, as this type of control will only yield tight hovers when close to the nominal position. If large disturbances act upon the tight hover condition, such as moving the quadrotor aggressively from the desired hover point, the quadrotor will react strongly due to the tight gains and cause unstable movements that are very slowly damped. A looser hover controller will enable better disturbance rejection, but will also not maintain as close to its desired hover point as a tighter controller, and thus will have a larger distance error. An example of the roll and pitch measurement results from a typical hover are shown in Figures 4.8(a), 4.8(b), 4.9(a), 4.9(b), respectively, including the output commands in terms of desired angles per Equation (4.12).



(a) Position and velocity. (b) Net output command.

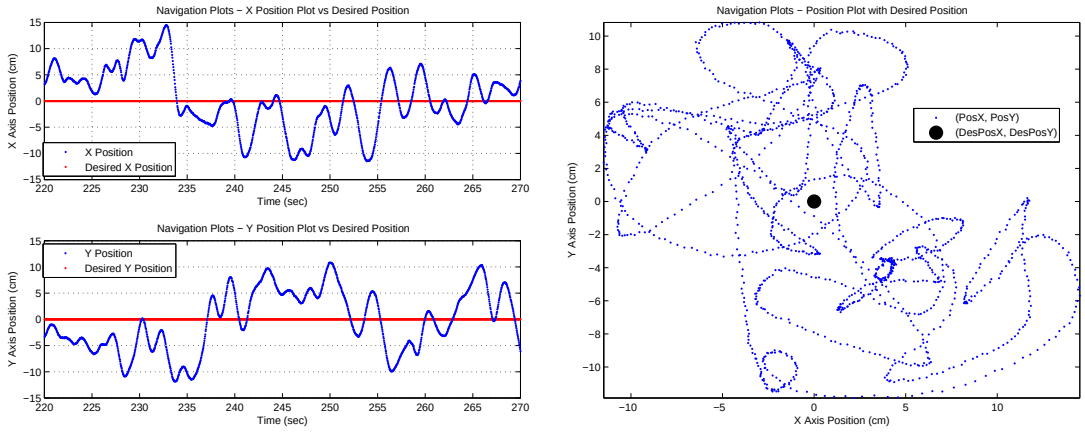
Fig. 4.8: Navigation hover measurements - roll/x axis.



(a) Position and velocity. (b) Net output command.

Fig. 4.9: Navigation hover measurements - pitch/y axis.

The performance of the navigation system during a hover is indicated in Figures 4.10(a), 4.10(b), showing the x and y measured positions versus the desired position as separate variables over time, as well as together on a coordinate plot. The approximated position error on a hover for the x axis is ± 13 cm, and ± 11 cm for the y axis. The y axis has a tighter error due to the differences between the roll and pitch axes, as described in Chapter 3.



(a) x and y separate over time.

(b) Coordinate plot.

Fig. 4.10: Position vs desired for hover.

4.9 Vision-Based Hover with Disturbance Rejection

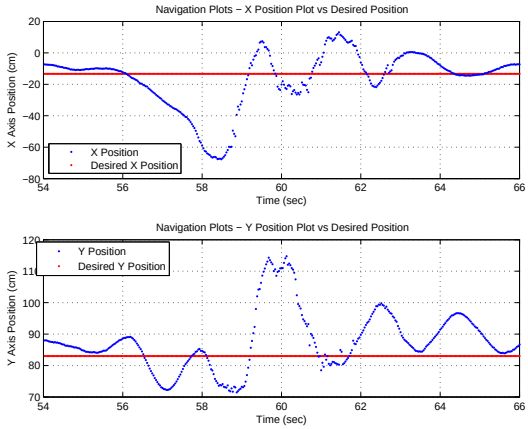
The vision-based navigation system and attitude controller is, in addition to being capable of accurate regulation and tracking, also very robust to disturbances. This robustness is demonstrated by recovery of the quadrotor after hitting one axis of the quadrotor with a stick as well as pulling the camera cable, displacing the quadrotor by over half a meter. Figure 4.11(a) shows an image of the quadrotor being hit by a stick, and Figure 4.11(b) shows the quadrotor being pulled by the cable. The response of the system to this disturbance is indicated in the graphs of Figure 4.11(c) for the x and y positions during the cord pulling disturbance and the recovery.

4.10 Vision-Based Navigation

As described in Section 4.6, the navigation controller also has a navigation path tracking mode. In this mode, the goal of the quadrotor is to track a changing path, and as such, it actually will have a desired velocity component. In this case, a predetermined velocity is given to the quadrotor during tracking mode, which drives the quadrotor to move at a certain speed while at the same time the distance measurement drives the quadrotor to the next point along the path. The controller works on normal and tangent components of the path as described in Section 4.10.4. An example of the measurement results from a square

(a) Quadrotor being hit by a stick.

(b) Quadrotor with cable being pulled.



(c) Positions of x and y during disturbance and recovery.

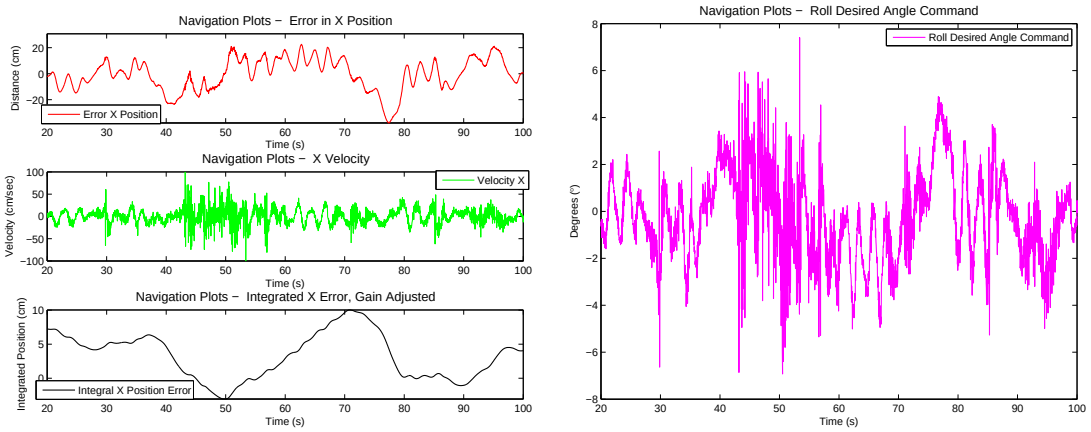
Fig. 4.11: Disturbance images and data.

path are shown in Figures 4.12(a), 4.12(b), 4.13(a), 4.13(b), 4.14(a), 4.14(b).

The performance of the navigation system during a square path is indicated in Figures 4.14(a), 4.14(b), showing the x and y measured positions versus the desired position as separate variables over time, as well as together on a coordinate plot. The approximated maximum position error on such a square path for the x axis is 37 cm, and 20 cm for the y axis. The y axis has a tighter error due to the differences between the roll and pitch axes, as described in Chapter 3.

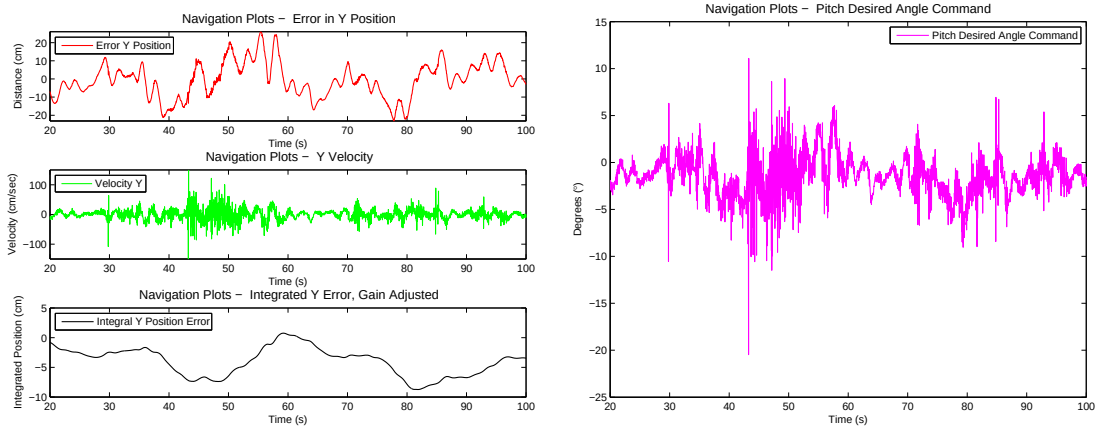
4.10.1 Path Setup

Generating a path requires specifically taking into account operational aspects of the quadrotor. Giving tracking updates of points that are too far apart will yield different



(a) Position and velocity. (b) Net output command.

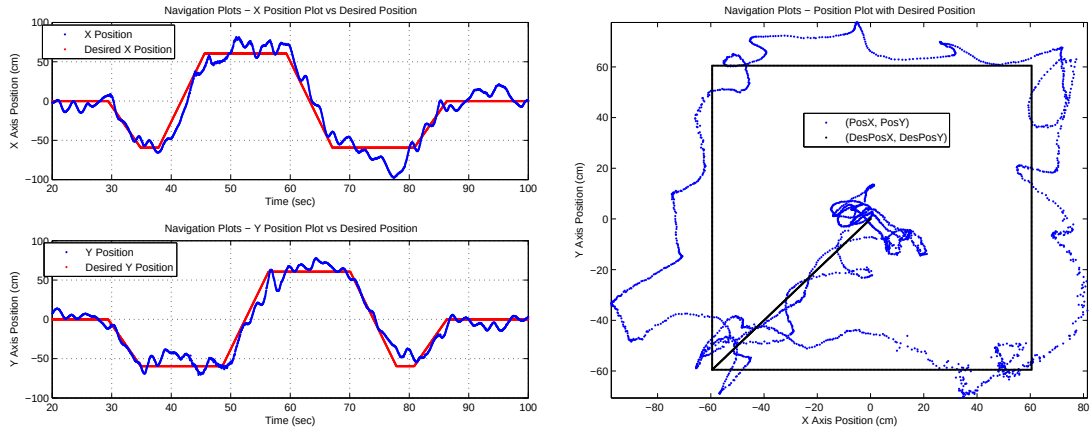
Fig. 4.12: Navigation path measurements - roll/x axis.



(a) Position and velocity. (b) Net output command.

Fig. 4.13: Navigation path measurements - pitch/y axis.

results for the same tracking gains as updates of points that are close together. The path generation used does not take into account any quadrotor dynamics and is based on a purely spatial separation rather than time and takes the form of geometric lines or curves [74]. To maintain simplicity, navigation gains were tuned for the hover mode, and then the path generation scheme - of how many points sent per second and how far apart the points are - is tuned to the pre-set gains. This is acceptable since the path can easily be modified or generated ahead of time on the quadrotor itself according to the needs of the quadrotor, given just a pure desired path based on a few points [74]. Hover conditions are built into



(a) x and y separate over time. (b) Coordinate plot.

Fig. 4.14: Position vs desired for path.

the path generation at path transition points in order to avoid excessive overshoot. Taking vehicle dynamics into account for path generation would eliminate this necessity and can be considered for future work.

4.10.2 Path Definition

In this context, path is used to refer to a sequential set of positions for which the vehicle is required to track, without necessitating any velocity or temporal requirements. Placing such additional requirements for a path tracking is sometimes referred to as a trajectory. For the path following of the quadrotor, a velocity requirement is added, but its value is currently independent of the type of path (as it is set manually as a parameter), and so the terminology for path following is used.

4.10.3 Path Generation

Paths are generated in Matlab for simplicity of testing, but could easily be processed on the quadrotor before flight according to the specific tuning setup, and thus simplify what is required of the operator. An example of the Matlab script for a square path generation is in Algorithm 4.1. The value for the distance gain, which sets the spacing between points, is related to the speed at which the quadrotor is intended to travel the path

and must be adjusted concurrently with the desired speed setting. This is not necessary for a prototype system which is only being used to demonstrate capabilities, and not to robustly interact with an operator. It is expected that having paths generated that take into account the dynamics and restrictions of the quadrotor translational movement, and even be dynamically generated onboard based on uncertainty planning in the map, could improve the quality and capability of path following [74, 100]. Such forward planning capabilities and path generation algorithms are options for future work.

Algorithm 4.1 Path Generation Example - Square

Begin

SampleFreq $\leftarrow 31$

hoverLength $\leftarrow$ SampleFreq $* 3$ /*hover for 3 seconds*/

$k_d \leftarrow 0.5$ /*gain for how far to go for each path point - in cm*/

$z^{des} \leftarrow 40$ /*desired height in inches*/

distToTravel $\leftarrow 200$ /*in cm*/

numPoints $\leftarrow \frac{\text{DistToTravel}}{k_d}$

/*Square Path with diagonal movement to/from corner */

/*where $[x]_{m,n}$ indicates an m by n matrix of values x*/

$$\text{path} \leftarrow \begin{bmatrix} \frac{-1}{\sqrt{2}} * k_d * [1]_{\frac{numPoints}{2*\sqrt{2}},1} & \frac{-1}{\sqrt{2}} * k_d * [0]_{\frac{numPoints}{2*\sqrt{2}},1} & z^{des} * [1]_{\frac{numPoints}{2*\sqrt{2}},1} \\ [0]_{hoverlength,1} & [0]_{hoverlength,1} & z^{des} * [1]_{hoverlength,1} \\ k_d * [1]_{numPoints,1} & [0]_{numPoints,1} & z^{des} * [1]_{numPoints,1} \\ [0]_{hoverlength,1} & [0]_{hoverlength,1} & z^{des} * [1]_{hoverlength,1} \\ [0]_{numPoints,1} & k_d * [1]_{numPoints,1} & z^{des} * [1]_{numPoints,1} \\ [0]_{hoverlength,1} & [0]_{hoverlength,1} & z^{des} * [1]_{hoverlength,1} \\ -k_d * [1]_{numPoints,1} & [0]_{numPoints,1} & z^{des} * [1]_{numPoints,1} \\ [0]_{hoverlength,1} & [0]_{hoverlength,1} & z^{des} * [1]_{hoverlength,1} \\ [0]_{numPoints,1} & -k_d * [1]_{numPoints,1} & z^{des} * [1]_{numPoints,1} \\ [0]_{hoverlength,1} & [0]_{hoverlength,1} & z^{des} * [1]_{hoverlength,1} \\ \frac{1}{\sqrt{2}} * k_d * [1]_{\frac{numPoints}{2*\sqrt{2}},1} & \frac{1}{\sqrt{2}} * k_d * [0]_{\frac{numPoints}{2*\sqrt{2}},1} & z^{des} * [1]_{\frac{numPoints}{2*\sqrt{2}},1} \\ [0]_{hoverlength,1} & [0]_{hoverlength,1} & z^{des} * [1]_{hoverlength,1} \end{bmatrix}$$

End

4.10.4 Path Following Control

A path is defined, $P \in N \times \mathbb{R}^2$, by a sequence of N desired way-points in two dimensions, $[x_i^{des} \;\; y_i^{des}]^T$, along a path segment P_i connecting way-point i to $i+1$, with a constant desired speed of travel for the path, $[\dot{x}_{pos}^{des} \;\; \dot{y}_{pos}^{des}]^T$. This path definition is shown visually in Figure 4.15. Let t_i be the unit tangent vector in the direction of travel along the path from

$[x_i^{des} \quad y_i^{des}]^T$ to $[x_{i+1}^{des} \quad y_{i+1}^{des}]^T$, and n_i be the unit normal vector to the path. Then, given the actual current position of the vehicle, $[x_{pos} \; y_{pos}]^T$, the normal path errors, $e_{nP}, \dot{e}_{nP}$ and tangent path errors, $e_{tP}, \dot{e}_{tP}$ are

$$e_{nP} = \left(\begin{bmatrix} x_i^{des} \\ y_i^{des} \end{bmatrix} - \begin{bmatrix} x_{pos} \\ y_{pos} \end{bmatrix} \right) \cdot n_i \ , \tag{4.21}$$

$$e_{tP} = \left(\begin{bmatrix} x_i^{des} \\ y_i^{des} \end{bmatrix} - \begin{bmatrix} x_{pos} \\ y_{pos} \end{bmatrix} \right) \cdot t_i \ , \tag{4.22}$$

$$\dot{e}_{nP} = - \begin{bmatrix} \dot{x}_{pos} \\ \dot{y}_{pos} \end{bmatrix} \cdot n_i \ , \tag{4.23}$$

$$\dot{e}_{tP} = \left(\begin{bmatrix} \dot{x}_i^{des} \\ \dot{y}_i^{des} \end{bmatrix} - \begin{bmatrix} \dot{x}_{pos} \ , \\ \dot{y}_{pos} \end{bmatrix} \right) \cdot t_i \ , \tag{4.24}$$

where the tangent path components of the desired velocities must be taken since the desired velocities are implemented independently from the desired path in terms of x and y components.

Both the along path error and error rates are used in order to simplify the controller operation between hover and path modes. This allows for an essentially seamless difference between the two modes, utilizing the same controller but just different desired positions and velocities. In the path, the error is used to drive the system to the desired position, while the error rate is used to keep the quadrotor to a specified velocity. Thus, the along

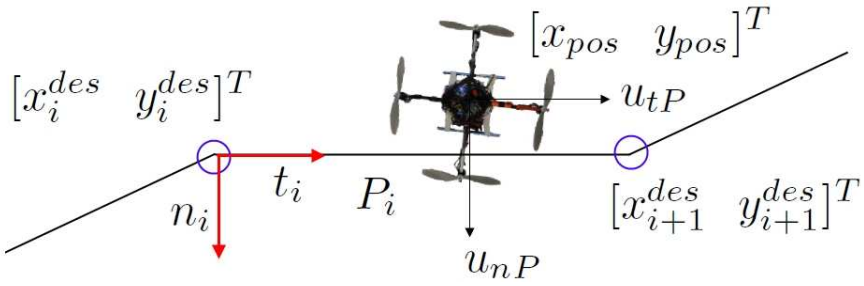


Fig. 4.15: Path definition, with way-points i and $i+1$ and corresponding path segments, noting the tangent and normal path components.

path tracking uses PD control, while the normal path regulation uses PID control, per

$$u_{tP} = \begin{bmatrix} u_{nav,x} \\ u_{nav,y} \end{bmatrix}_{tP} = \begin{bmatrix} k^x_{ptP,nav} & 0 \\ 0 & k^y_{ptP,nav} \end{bmatrix} \cdot e_{tP} + \begin{bmatrix} k^x_{dtP,nav} & 0 \\ 0 & k^y_{dtP,nav} \end{bmatrix} \cdot \dot{e}_{tP} \quad (4.25)$$

$$u_{nP} = \begin{bmatrix} u_{nav,x} \\ u_{nav,y} \end{bmatrix}_{nP} = \begin{bmatrix} k^x_{pnP,nav} & 0 \\ 0 & k^y_{pnP,nav} \end{bmatrix} \cdot e_{nP} + \begin{bmatrix} k^x_{dnP,nav} & 0 \\ 0 & k^y_{dnP,nav} \end{bmatrix} \cdot \dot{e}_{nP} + \begin{bmatrix} k^x_{inP,nav} & 0 \\ 0 & k^y_{inP,nav} \end{bmatrix} \cdot \int_0^t e_{nP} dt. \quad (4.26)$$

For simplicity, the path gains and the hover gains are set equal, limiting the need to tune the path following separate from the hover. Although task specific gains could be explored in future work, the hover gains were found to have a good carryover to the path following. Thus, $k^x_{ptP,nav} = k^x_{pnP,nav} = k^x_{p,nav}$, $k^x_{dtP,nav} = k^x_{dnP,nav} = k^x_{d,nav}$, $k^x_{inP,nav} = k^x_{i,nav}$, and similarly for the y-axis gains.

Since u_{tP} and u_{nP} are ultimately implemented in x and y axes for roll and pitch desired accelerations for decoupled control, u_{tP} and u_{nP} need to be merged, such that the outputs of Equation (4.10) become

$$\ddot{x}^{des} = u^{tP}_{nav,x} + u^{nP}_{nav,x}, \quad (4.27)$$

$$\ddot{y}^{des} = u^{tP}_{nav,y} + u^{nP}_{nav,y}. \quad (4.28)$$

Completion of segment i for transition to segment $i+1$ occurs at the point that the quadrotor reaches way-point $i+1$. Upon completion of P_i, the integrators for the normal path error are reset if the direction to way-point $i+2$ is not tangent to t_i. This is to account for the fact that the built up error for the normal path is tied to the path direction which is based upon the components of x and y, and will not carry over directly to a different path direction. Alternatively, low curvature paths where the angle between t_i and t_{i+1} is small, the integrator could be kept. Additionally, it might be beneficial in future work to take components of the built up integrator for appropriate application to the current direction,

allowing for reduction in the time required for the integrator to build up for the new path segment.

As noted above, the path does not incorporate changes in the vertical path dimension. Extending the path controller to three dimensions can be explored in future work.

4.11 Altitude Control Using Vision

Using camera vision to control the behavior of the quadrotor, specifically altitude, is a recent and hot topic, as this improves the capability of a single sensor. Camera algorithms are still not fast or robust enough yet to control the attitude of a quadrotor vehicle, but controlling the height is a first step towards this goal, and reduces the number of other sensors needed for autonomous control. Altitude control has been performed on a quadrotor using ground plane estimation with a laser [100]. Some unpublished work uses an inexpensive stereo camera to control the altitude, also using ground plane estimation techniques. For the quadrotor here, a PID controller is used, exactly identical to the controller used for the sonar-based height control, with the model

$$u_{alt,nav} = k^{nav}_{p,alt}(z^{des} - z) + k^{nav}_{i,alt}\int_0^t (z^{des} - z)dt + k^{nav}_{d,alt}(\dot{z}^{des} - \dot{z}) + u_{nom}, \tag{4.29}$$

where $u_{alt,nav}$ is used as the input to all four motors, the same way u_{alt} is used for the sonar based height controller; and u_{nom} is the same value as for the sonar altitude controller. The integral and derivative terms are calculated the same way for the vision altitude control as they are for the sonar height control, only the actual measurement source is different. Also, no median filtering is done on the camera z measurement, as the data is quite reasonable.

A Kalman filter is in place to fuse the height between the sonar and the camera, however it is not actively used. The quality of the camera altitude measurements is fast enough and good enough that fusing with the sonar does not provide much additional accuracy. Also, there is some difficulty involved in the implementation of the fusing filter. For one, the camera measured height depends on the manual initialization technique described in

Section 4.5.2, meaning that a constant absolute height measurement is not currently being given by the PTAM algorithm, so before fusing, the camera height would have to be offset by an amount specific to the recent initialization. Additionally, the cases when the camera is not giving data or is not activated will have to be explicitly taken care of.

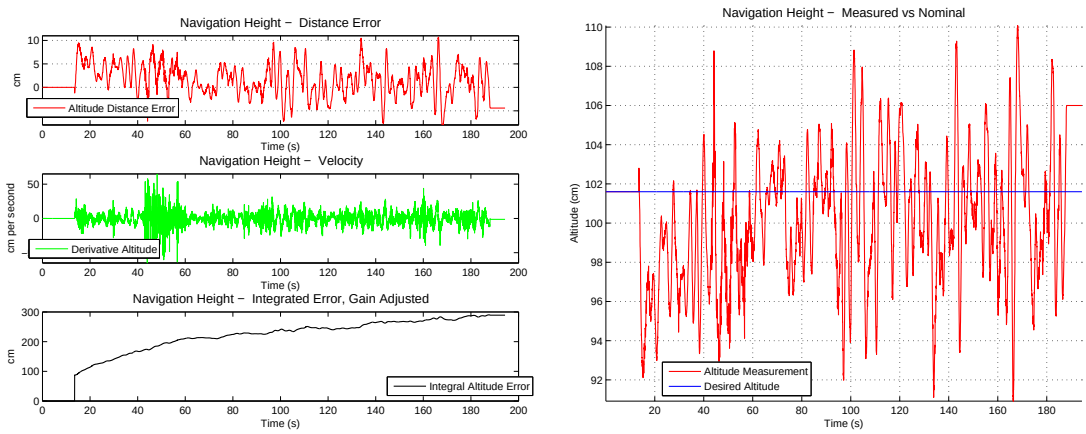
An interesting feature in this quadrotor application, is that the sonar-based height control actually runs slower than the camera information. Although the camera has a larger delay between the true state of the system and the measurement, the median filter on the sonar brings the relative delay to roughly equal. In the application of vision-based height control, the camera position data is quite a bit smoother than the sonar measurements, due to the reduced quantization issues and greater accuracy. The results obtained are shown in Figures 4.16(a) and 4.16(b). A height approximate error of ± 6 cm is obtained using vision, which is approximately equal to the capability achieved with sonar.

4.12 Outdoor Environment Ready

The quadrotor navigation controller utilizes x, y, and z input positions to control its position and heading. Inside, a sonar and camera yield effective results for height, position, and heading. Simply replacing these sensors with a barometer, GPS and magneto meter would yield a system equally capable of controlling its altitude, position, and heading while in an outdoor environment. Such an approach has been demonstrated before on the Starmac [31]. Drop in replacements of these sensors would be easy to find and implement, making this platform usable in various environments.

4.13 Safety

As outlined in Chapter 3, safety measures are a crucial piece of maintaining a flying vehicle. Since the navigation controller is already dependent on the attitude controller for maintaining overall stability, the safety measures of the navigation system are primarily to prevent it from interfering with the attitude controller or giving an attitude reference command that would be unrecoverable. These safety measures include:



(a) Error. (b) Compared to nominal.

Fig. 4.16: Altitude performance using vision.

- The net command from the navigation controller to the attitude controller has an angle saturation limit;
- The navigation system does not activate, or will turn off, if the quadrotor is not within a certain boundary of the desired altitude, based on the sonar measurements;
- If the ground station stops sending navigation packets (due to poor tracking quality or activation of the recovery mode), or the quadrotor does not receive any, the navigation controller will turn off;
- The navigation controller is inactive during a landing.

4.14 Results

The capabilities and accuracies of the system are shown in the figures throughout this chapter. In general the quadrotor is capable of autonomously generating a map of its surroundings, localizing itself upon the map and performing hover, way-point following and path tracking. The yaw is able to be regulated to within $\pm 2°$. A hover accuracy with an approximated error circle diameter of 13 cm can be obtained. Path following can be achieved within a maximum approximated error bound of 40 cm. Using purely vision to control altitude, steady state approximated error is only 6 cm. It is important to remember

that these accuracy results are achieved with a USB wire hanging from the quadrotor, which pulls on it from a constant location. This acts like a constant outside disturbance on the system.

Navigation system gains used to achieve accurate hover and path following are shown in Table 4.1.

Table 4.1: Navigation system gains.

Gain Term	Notation	Value
X Proportional	$k^x_{p,nav}$	1.15
Y Proportional	$k^y_{p,nav}$	1.20
X Integral	$k^x_{i,nav}$	0.002
Y Integral	$k^y_{i,nav}$	0.002
X Derivative	$\dot{k}^x_{p,nav}$	0.70
Y Derivative	$\dot{k}^y_{p,nav}$	0.95
Navigation Altitude Proportional	$k^{nav}_{p,alt}$	3.1
Navigation Altitude Integral	$k^{nav}_{i,alt}$	0.023
Navigation Altitude Derivative	$k^{nav}_{d,alt}$	1.2
Yaw Kalman Filter Process Noise Variance	Q^{ψ}	0.1
Yaw Kalman Filter Observation Noise Variance	R^{ψ}	0.08

4.15 Comparison

Direct comparisons of results are always difficult without a common standard. The differences between each implementation are extensive and so checking numerical results does not take into account the differing conditions and methods. However, from a survey of the literature, the navigation system capabilities of the quadrotor presented here is certainly on the same level as other indoor quadrotors that are using just onboard sensing. Even when compared to the highly accurate 3D vision system results, this quadrotor is off by less than an order of magnitude. The additional aspect of using components that are over an order of magnitude less in cost than other implementations shows the true advantage of this system.

4.16 Chapter Contributions

A complete navigation system, capable of navigating through unknown environments,

using a custom quadrotor with very low-cost sensing capabilities has been demonstrated. The system is capable of an autonomous hover and following a path, both with good accuracy. The vision system, PTAM, has been adjusted for use on a flying vehicle and is the first implementation of the algorithm on a quadrotor with such low-cost sensors. The quadrotor is also capable of regulating altitude using monocular vision only.

4.17 Chapter Summary

This chapter presented an overview of navigation systems in general, the challenges for indoor quadrotor vehicles and some relevant navigation implementations that have been demonstrated previously. Some of the vision-based navigation methods were presented and contrasted with the approach taken by the vision system presented here. The function of the vision software was covered and how it is used on the quadrotor. The complete navigation control system was outlined in detail and finally, vision-based height control, hover and path tracking experimental results were presented and discussed.

Chapter 5

Nonlinear Navigation and Altitude Control

This chapter presents a novel nonlinear controller for application to the navigation and altitude systems. The motivation for the new approach to the control of these systems from what was presented in Chapter 4 is covered, with a discussion of the limitations of the original controller. The controller design is shown, with specific applications to altitude and navigation control, then the results of the nonlinear controller and a comparison to the original design are covered.

5.1 Motivation of New Method

The design of the quadrotor system made specific assumptions about the dynamics, specifically considering the region around the hover point, where angles are small and system dynamics are fairly linear. This is an acceptable approach, given the success of the quadrotor presented, but begins to come apart when the boundaries of the system are stretched. Specifically, two issues were noticed during experimental testing that motivated the implementation of a new controller. These two issues occurred during takeoff and way-point navigation when far from the desired location.

5.1.1 Linear Controller Limitations

The linear controller was designed only for perturbations around the hover point. Stability is achieved with a linear PID controller per Chapter 4 when the quadrotor is near the desired hover point, but instability appears when the distance to the hover point becomes too large. Aggressive flights in which rapid movements occur are not considered in this problem analysis, but rather situations that occur during typical hover and path navigation.

5.1.2 Altitude

There are only two cases in a typical flight that the distance from the nominal height above the ground will be large: when far below the hover point as when taking off, or when far above the hover point, when initial thrust was too high and the altitude controller is engaged well above the desired height.

Activation of the altitude controller during the initial stages of takeoff, when the quadrotor is well below the desired height and gaining altitude purely from the initially applied nominal thrust, the output of the PID controller is an excessive value, causing the quadrotor to overshoot the desired height, leading into a very slowly damped oscillation while drawing excessive power to change the motor speeds by such large margins. A nearly identical result occurs if the initially applied nominal thrust is too large, causing the quadrotor to exceed the nominal height rapidly on takeoff; an engagement of the altitude controller at that point will result in oscillations.

Landing is another situation in which the system is far from the hover point, however this is taken care of using a state machine approach, per Chapter 3. This allows simply a change in the desired height in small steps, keeping the system in the expected linear region until very low to the ground, at which point a state machine landing maneuver is executed due to the sensor measurement restrictions.

5.1.3 Navigation Tracking

There are two cases in which the navigation system will find itself far from the desired location: when a disturbance or other anomaly causes the quadrotor to be moved far from the steady state desired location, or when a desired position is given that is far away.

Disturbance rejection often involves the discussion of loose versus tight controllers [76]. A tight controller will maintain good desired reference tracking, but just slight disturbances or unaccounted-for dynamics will cause the system to oscillate. Loose controllers have good damping when such disturbances occur, but have the trade-off of reduced tracking capability. In the navigation controller, a balance could be struck that was generally satisfactory, but better tracking results are desired and large disturbances will still cause oscillations.

Way-point tracking to a distance far from the current position will yield a similar overshoot into oscillation. Even worse, if the way-point is a large enough distance from the current position that the desired angle command from the navigation system crosses a certain threshold, the quadrotor may not even be able to recover from such a large angle and will simply flip over and crash before it even makes it far from the current position. Setting way-points using a parsing algorithm for the quadrotor to piecewise achieve intermediary way-points is a way around this problem, but is not a satisfactorily robust solution in general.

5.2 Nonlinear Control Options

From the above situations of the difficulties with the linear controller, it is clear that the issues only arise when far from the nominal point, and that when around this point the linear controllers perform well. This means that a new controller should have a mostly linear behavior around the boundary of the nominal point, but have some reducing effect for large values away from the nominal. For such a desire, the application of a common saturation structure becomes a simple solution, but this is not the only option.

5.2.1 Saturation on Output

Implementing a simple saturation ceiling for the output of the controller would certainly prevent excessive commands due to large distances from the nominal, however this is not an ideal solution, as there is no consideration for the balance between the proportional and derivative terms. Since either or both position or velocity could contribute to large desired outputs, a simple saturation may put too much emphasis on one over the other, yielding undesirable stability problems. Saturations of the individual proportional and derivative terms would be an improvement, but still problematic, as such a sharp transition from the linear region to the saturation level may lead to performance degradation and the non-differentiable nature of the controller means stability analysis is more difficult. However, a nested saturation controller for a vision-based hover position stabilization system was empirically shown to provide smoother flight than implementations of backstepping or

sliding mode control [110].

5.2.2 Sigmoidal Control

A simpler solution to the problem is the use of a nonlinear curve that already exhibits the behavior that is desired: a linear region close to the origin (the nominal position); an asymptotic saturation as the values increase; and a smooth transition between the linear region and the horizontal asymptote. The class of sigmoid curves follow exactly this description. Sigmoids are S-shaped curves produced by several mathematical functions and are often used to model growth; where over time, increases in the measurement occur until they come to reach a saturation point. Common sigmoid functions are the arctangent, the hyperbolic tangent, the logistic function, the Gompertz curve, and the error function.

5.3 Control Implementation Using Sigmoid

The controller design is implemented using the sigmoidal generalized logistic function [111]. The logistic function is a very flexible sigmoid function; for the requirements here, several of the flexible variations are not applicable. A symmetrical function is desirable so that no switching action needs to be done, thus the upper and lower asymptotes of the function will be equal and opposite. And with the output required to be zero when the input is zero, the function takes the simplified form

$$Y(t) = \frac{A}{1 + e^{-Bt}} - A/2, \tag{5.1}$$

where A sets the upper and lower asymptotes and B sets the growth rate. This function is used over other sigmoid functions due to its slope and saturation set point flexibilities, its symmetrical property, and the use of a single exponential for simpler stability derivation.

5.3.1 Altitude Regulation

The altitude is regulated using a PID controller, with the P and D commands derived using the sigmoid function. The integral term is kept using the linear control, since its

purpose is to counteract the decaying battery power, which can be handled using a linear approximation. Although the integrator will build up when far from the nominal height, it is slow moving enough that the nonlinear controller will be able to achieve the desired height before the integrator becomes too large. In future work, an anti-windup scheme could be used to reduce this effect. The controller is then

$$u_{alt} = \frac{A_{kp}^{alt}}{1+e^{-B_{kp}^{alt}(z^{des}-z)}} - \frac{A_{kp}^{alt}}{2} + k_{i,alt}\int_0^t (z^{des}-z)dt + \frac{A_{kd}^{alt}}{1+e^{-B_{kd}^{alt}(\dot{z}^{des}-\dot{z})}} - \frac{A_{kd}^{alt}}{2}, \quad (5.2)$$

where u_{alt} is used as described in Chapter 3, with the motor mixing to add to the PWM for all the motors, per Equation (3.27). The gain constants used for the nonlinear altitude are indicated in Table 5.1.

5.3.2 Navigation Positioning Control

The navigation positioning for both hover and path following is regulated using a PID controller, with the commands derived using the sigmoid function. The controller is then

$$\begin{aligned} u_{nav,x} &= \frac{A_{kp}^{nav,x}}{1+e^{-B_{kp}^{nav,x}(x_{pos}^{ref}-x_{pos})}} - \frac{A_{kp}^{nav,x}}{2} + \frac{A_{kd}^{nav,x}}{1+e^{-B_{kd}^{nav,x}(\dot{x}_{pos}^{ref}-\dot{x}_{pos})}} - \frac{A_{kd}^{nav,x}}{2} \\ &\quad + ki_{sig}^{nav,x}\int_0^t \frac{A_{kp}^{nav,x}}{1+e^{-B_{kp}^{nav,x}(x_{pos}^{ref}-x_{pos})}} - \frac{A_{kp}^{nav,x}}{2}dt, \quad (5.3) \\ u_{nav,y} &= \frac{A_{kp}^{nav,y}}{1+e^{-B_{kp}^{nav,y}(y_{pos}^{ref}-y_{pos})}} - \frac{A_{kp}^{nav,y}}{2} + \frac{A_{kd}^{nav,y}}{1+e^{-B_{kd}^{nav,y}(\dot{y}_{pos}^{ref}-\dot{y}_{pos})}} - \frac{A_{kd}^{nav,y}}{2} \\ &\quad + ki_{sig}^{nav,y}\int_0^t \frac{A_{kp}^{nav,y}}{1+e^{-B_{kp}^{nav,y}(y_{pos}^{ref}-y_{pos})}} - \frac{A_{kp}^{nav,y}}{2}dt, \quad (5.4) \end{aligned}$$

where $u_{nav,x}, u_{nav,y}$ are used as described in Chapter 4 with the output being interpreted as

Table 5.1: Altitude nonlinear controller gains.

Gain Term	Notation	Value
Altitude Proportional Asymptote Bound	A_{kp}^{alt}	60
Altitude Proportional Growth Rate	B_{kp}^{alt}	0.2
Altitude Derivative Asymptote Bound	A_{kd}^{alt}	40
Altitude Derivative Growth Rate	B_{kd}^{alt}	0.071

a desired acceleration and then used to determine the desired angle according to Equation (4.10). The gain constants used for the nonlinear navigation are indicated in Table 5.2.

Integration Term

Note that a linear integrator is not used here, as unlike for the altitude control, the integrator will begin to build up excessively if far from the desired position, causing an unrecoverable situation. As shown, the integrator term is determined from integrating the position error after it has been modified through the nonlinear proportional function, making it somewhat of a hybrid linear-nonlinear integrator, as it linearly accumulates the nonlinear system output. Although if constantly far from the nominal position, this integrator would get large, in general it stays small enough due to the gain value and the integration of nonlinear saturated outputs. Calculating the integrator term via integration of the actual position error and then run through its own nonlinear function for growth rate and saturation level tunings is not a good option. This is because when far from the desired position, the integrated position error will get large very fast, and if this value is used for integration, it will be very large most of the time, and thus when run through the nonlinear controller, will always be at the saturation level. An alternative method is to blend the two, where the integration is performed on the nonlinear system output, as it is now, but then that value is operated on by its own nonlinear function, with separate growth rate and

Table 5.2: Navigation nonlinear controller gains.

Gain Term	Notation	Value
Navigation x Proportional Asymptote Bound	$A_{kp}^{nav,x}$	100
Navigation x Proportional Growth Rate	$B_{kp}^{nav,x}$	0.04
Navigation y Proportional Asymptote Bound	$A_{kp}^{nav,y}$	100
Navigation y Proportional Growth Rate	$B_{kp}^{nav,y}$	0.04
Navigation x Derivative Asymptote Bound	$A_{kd}^{nav,x}$	90
Navigation x Derivative Growth Rate	$B_{kd}^{nav,x}$	0.042
Navigation y Derivative Asymptote Bound	$A_{kd}^{nav,y}$	90
Navigation y Derivative Growth Rate	$B_{kd}^{nav,y}$	0.052
Navigation x Integral Gain	$ki_{sig}^{nav,x}$	0.004
Navigation y Integral Gain	$ki_{sig}^{nav,y}$	0.005

saturation level tunings. This could be explored in future work for possible improvements in performance.

5.4 Results

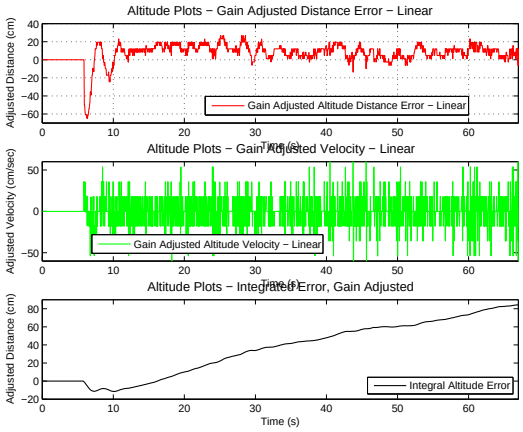
Experimental testing has validated the use of such a controller to address the original limitations of the linear PID controllers. Performance is similarly capable when around the nominal hover point while also being stable and effective in bringing the system back to the hover point when the system is far from the desired position.

5.4.1 Altitude Results

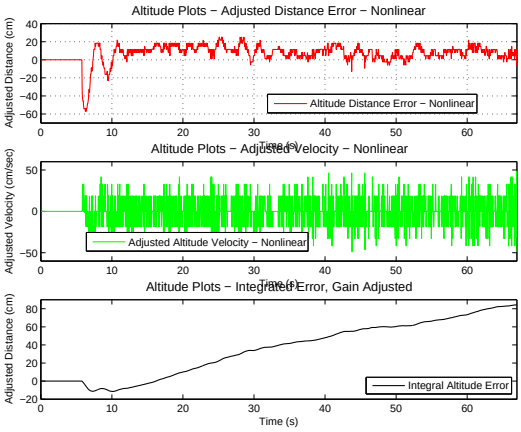
The function of the altitude controllers, both linear and nonlinear, are shown in Figures 5.1(a) and 5.1(b). These figures show the proportional and derivative values internal to the controller after gain adjustment for the same flight. In this case, the nonlinear controller was the active controller. For the linear controller, gain adjustment just means a multiplication by the linear gains. For the nonlinear controller, this means the application of the sigmoidal function shown in Equation (5.2). The integral is common between the two. Also, the net thrust command to the motors, in terms of the PWM value, is shown for the two controllers in Figure 5.1(c). As can be seen from these graphs, as expected the nonlinear controller smooths out the peaks in both the derivative and proportional controller components, and thus, reduces the net output command. Since these peaks are not related to actual movements in the system, being too fast for the system dynamics, the reduced output command of the nonlinear controller prevents response anomalies as the system tries to actuate all the motors quickly.

5.4.2 Navigation Results

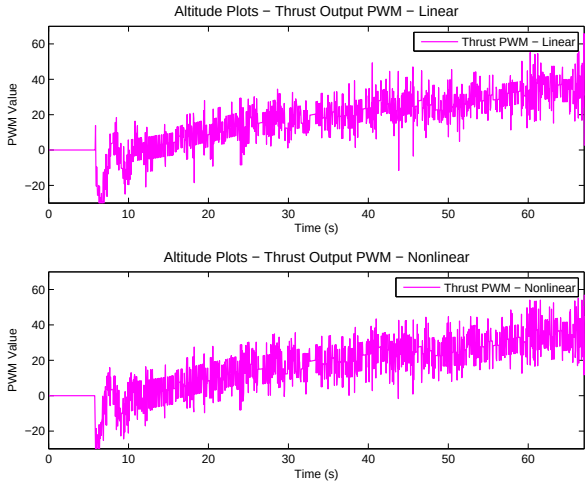
The function of the navigation controllers, both linear and nonlinear, are shown in Figures 5.2(a) and 5.3(a) for linear, and Figures 5.2(b) and 5.3(b) for the nonlinear controller during a hover. These figures show the proportional and derivative values internal to the controller after gain adjustment. In the case of the linear controller, this just means a



(a) Gain adjusted PD values, linear.



(b) Gain adjusted PD values, nonlinear.

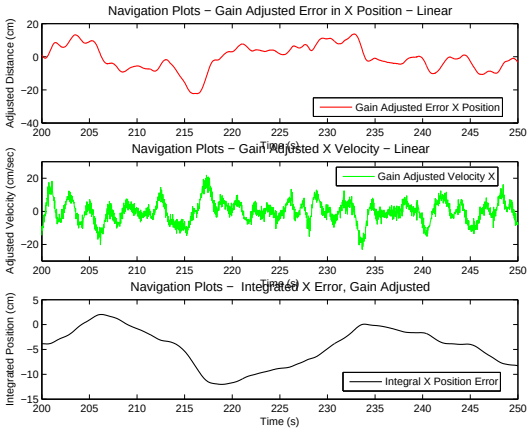


(c) PWM command values for the two controllers.

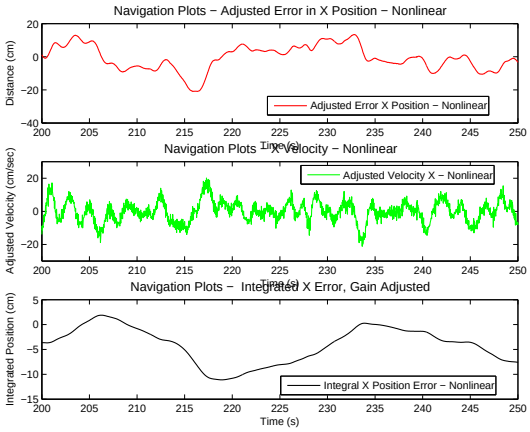
Fig. 5.1: Altitude controller comparisons.

multiplication by the linear gains. For the nonlinear controller, this means the application of the sigmoidal function shown in Equation (5.3). The values come from the same hover flight, with the nonlinear controller being the active controller. Also, the net command to the attitude controller, in terms of the acceleration, is contrasted for the two controllers in Figures 5.2(c) and 5.3(c) for the x and y axes.

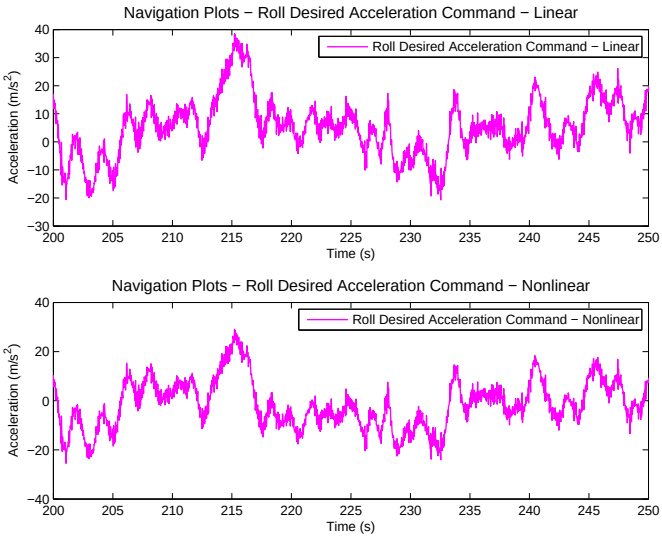
When the quadrotor is following a path, where a movement is desired within a specific range, there is even more difficulty for the linear controller, as measurements from the camera have a greater chance of errors and a greater effect on the system when used. To mitigate this, the gains would need to be lower, and thus diminish the accuracy of the



(a) Gain adjusted PID values - x axis, linear.



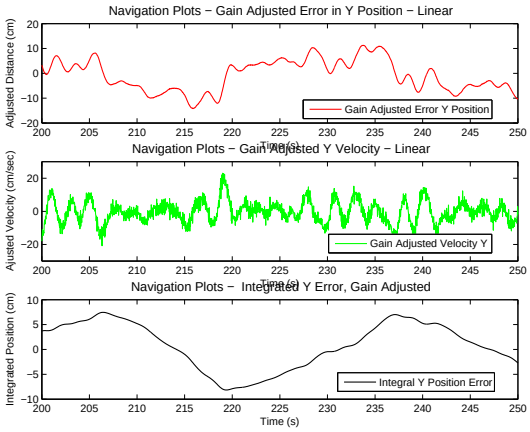
(b) Gain adjusted PID values - x axis, nonlinear.



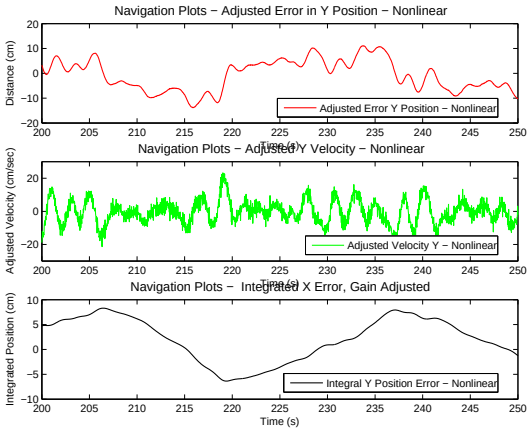
(c) Desired accelerations for the two controllers - x axis.

Fig. 5.2: Navigation controller comparisons for hover - x axis.

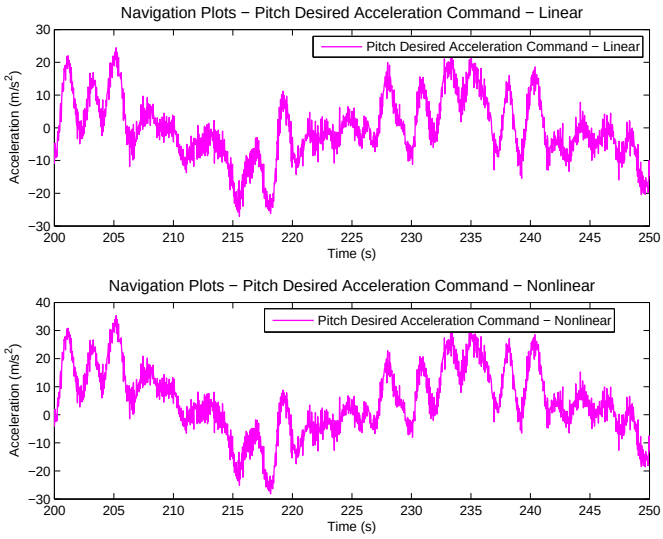
path following. The function of the two navigation controllers during a path following are shown for the roll/x axes in Figures 5.4(a) and 5.4(b) for the function of the controllers, and Figures 5.4(c) for contrasting the acceleration controller outputs. These graphs show how the nonlinear is even more critical during maneuvers, as responsiveness can be obtained without causing undesirable stability issues from occasional spikes in the measurements from noise or finite differentiation.



(a) Gain adjusted PID values - y axis, linear.



(b) Gain adjusted PID values - y axis, nonlinear.

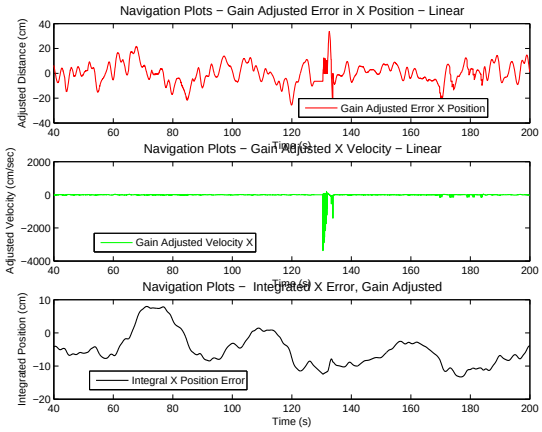


(c) Desired accelerations for the two controllers - y axis.

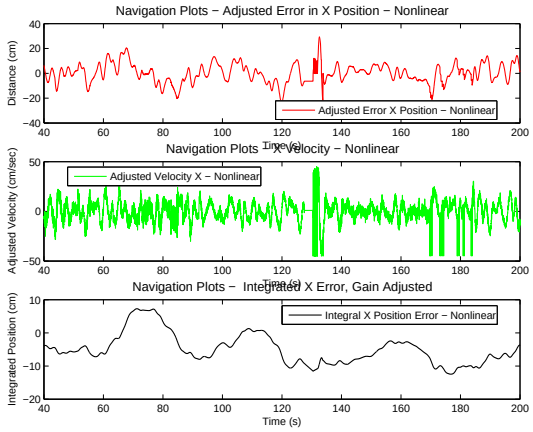
Fig. 5.3: Navigation controller comparisons for hover - y axis.

5.5 Noisy Measurement Benefits of Controller

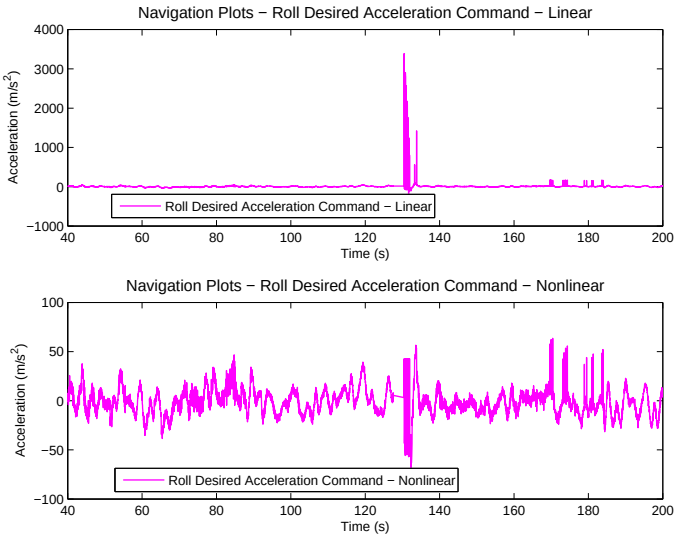
An auxiliary unexpected benefit of the nonlinear controller is the limiting of damaging effects from noise in the measurements. Although the nonlinear sigmoid implementation cannot replace the function of a filter and would be problematic in a system where large instantaneous measurement values are actually accurate representations of the system dynamics, on the quadrotor, large changes in position between frames (just 33 ms), and thus large velocities, are not possible responses of such a system. Its slower dynamics prevent such excessive changes, so that removing them via saturation does not result in a loss of



(a) Gain adjusted PID values - x axis, linear.



(b) Gain adjusted PID values - x axis, nonlinear.



(c) Desired accelerations for the two controllers - x axis.

Fig. 5.4: Navigation controller comparisons for path - x axis.

useful information for control stabilization. As can be seen in Figure 5.1(a), the altitude velocity changes frequently due to quantization and resolution noise. Although the threshold for the values shown are mostly below the saturation asymptote of the sigmoid, occasionally the camera wire is seen by the sonar which is not filtered out by the median filter.

The navigation positioning measurement issues are much more apparent, due in part to the small dt scaling factor. In Figure 5.4(a), the gain adjusted velocity component just after the 130 sec mark would be over a value of 3,000 for the linear controller, which would certainly cause an excessive desired angle over what is actually needed, as can be seen from

the desired acceleration output at that point compared to the nonlinear controller output.

5.6 Comparison to Linear Control

The performance of the two controllers can only be roughly compared from two different flights. Many tests would need to be done to determine statistical significance of the comparison, but qualitatively the performance improves with the nonlinear controller, with the main performance difference being that of consistent results due to robustness to many varying environmental and sensor effects, rather than a dramatic difference in accuracy.

5.7 Chapter Summary

In this chapter a novel approach to application specific problems with the linear control method on the altitude and navigation systems was presented. The form of the new controller for the two systems and the results from them were shown. Finally, an added benefit was discussed and a comparison between the linear controller and the nonlinear controllers was examined indicating the advantage of the nonlinear controller.

Chapter 6

Perching Aerobatic Maneuver

Building upon the capabilities presented in Chapter 3 and Chapter 4, the idea of pushing the flight envelope assumptions of near hover region control becomes feasible. Although the capabilities already presented yield a highly effective autonomous UAS that can be used for many types of missions, it becomes apparent upon the study of expert manually controlled MAVs, as well as natural flying creatures such as birds, that autonomous MAVs are capable of performing more maneuverable flight, and such capability is a very desirable trait for the ultimate goals of these vehicles.

This chapter examines the attributes of the general class of aggressive maneuvers, specifically discussing a specific aerobatic maneuver called perching. The motivation and relevance of such a maneuvering capability is covered, followed by an examination of the current body of related work in this specialized field and what is novel in the research presented in this thesis. The methods of achieving perching on this quadrotor platform are then covered, including the sensors used and the control architecture employed. Finally, a presentation of the results achieved from experimental testing is given, concluding with an overview of the contributions of the chapter and a summary of what was done.

6.1 Aggressive Maneuvering

High-speed flight for MAVs can easily be seen as desirable, as it can perform its mission faster, as well as achieve goals such as evasive maneuvering. With high-speed flight however, comes the necessary requirements of effective aggressive maneuvering, requiring flight changes outside of the typical aerodynamic region. Such changes outside of the typical aerodynamic region provided by standard control surfaces can also be required for aggressive maneuvers that are not necessarily from high-speed translational flight, but are required

based upon the desired end achievement, such as landing on abnormal surfaces or achieving a desired aerobatic trajectory, such as a flip.

Such aerobatic agility typically requires dynamic transitions of the attitude of the vehicle beyond the near-horizontal flight envelope and often uses extreme configurations such as near vertical configurations occurring within a translational movement, known as a post-stall transition. This post-stall configuration is characterized by airflow separation along the aircraft, causing rapid deceleration. One such maneuver that includes a dynamic transition from high-speed translational flight to a post-stall configuration is perching.

6.2 Perching

Perching is a specific type of aerobatic aggressive maneuver. It is characterized by a rapid reduction in speed through the use of a high angle of attack of the aircraft wing surfaces such that a point landing can be achieved without excess forward motion. Such a maneuver is routinely performed by birds, such as in Figure 6.1.

6.2.1 Motivation

This type of maneuver would allow such abilities as landing in constrained spaces. The applications of this capability include surveillance, inspection, and mobile sensor networks. The practical use of such abilities is already being examined, including the capability to perch on power lines for recharging [112]. The military is currently considering the use of

Fig. 6.1: Bald eagles landing on a perch.

groups of quadrotor UAVs that can perch on power lines, rocks and rooftop ledges ahead of convoys, for obtaining advanced surveillance, possibly even allowing determination of enemy locations through the use of sound triangulation.[1]

6.2.2 Being Precise

The use of dynamics of aggressive maneuvering, where the aerodynamics are much less stable, typically requires a trade-off between accuracy and agility. This is primarily due to the not well known nature of the dynamics during these maneuvers in order to control them appropriately for transitions that are accurately measured against a desired trajectory. Hobby aircraft, which routinely perform such maneuvers under experience pilot control, have much difficulty in executing these maneuvers in any sort of constrained environment. Birds, however, can effectively perform such dynamic maneuvers on a regular basis, and to do so with extreme accuracy.

6.2.3 Controlled Maneuverability

Obtaining such accuracy while performing such highly dynamic maneuvers on an autonomous aircraft requires consideration of the needed changes to the traditional autonomous flight controls. Such control must take into account the post-stall aerodynamic effects and the induced changes in maneuverability at such dynamic transitions in order to effectively obtain accurate aerobatic transitions within an constrained trajectory.

Due to the high speed of the perching maneuver, and specifically the transition to the post-stall configuration for making a point landing, the effectiveness of thrust producing plants is drastically reduced. The limitations on the responsiveness of the thrust plant prevent drastic changes in the force output from these systems, meaning that obtaining a significant reduction in speed using these plants is not feasible. Speed changes must be accomplished by effectively transitioning the wing surfaces of the vehicle to obtain aerodynamic braking, as occurs during the post-stall maneuver. The controller must then take

[1] http://spectrum.ieee.org/automaton/robotics/robotics-software/quadcopter-hexacopter-octocopter-uavs

into account the proper use of the force producing components of the vehicle, utilizing relative thrust magnitudes in order to manipulate the angle of attack for obtaining desired aerodynamic responses, as well as for adjusting the error of the vehicle normal to the trajectory.

6.2.4 Rotary-Wing Perching

Birds, as representatives of a flapping-wing vehicle, are the inspiration for perching maneuvers, and as such, their carryover holds mostly for similar style structures, in the general fixed-wing style. Fixed-wing aircraft are similar to most birds in that they are unable to hover, requiring maintaining a minimum speed in order to stay in the air. Such a restriction means that landing on a constrained spot, such as in a perch, is even more important, as the perching maneuver is the only means by which to land in such a location.

With rotary-wing aircraft, however, their hovering capability means that many landing locations are more accessible, simply by flying over to the location and landing from a hover. Such a hovering capability does not eliminate such vehicles from the need for constrained landings, such as perching, however. Some surfaces, as well as the orientation of the surface, prevents simply landing from a hover, limiting the landing options for the rotary aircraft. Additionally, having the agility to land in a variety of conditions widens the operational envelope of these vehicles.

The dynamics of the perching maneuver are also different on rotary-wing aircraft. Since the direct motion of the wing surfaces are providing thrust, as opposed to in fixed-wing aircraft where it is the movement through the air that generates lift, it means that rotary-wing aircraft must appropriately take into account the movement of the wing surfaces to obtain the desired vehicle aerodynamic transitions. The post-stall aerodynamics are thus different for rotary-wing aircraft as there is a spinning wing surface while at a high angle of attack and initial fast translational speeds, incorporating both the detached airflow effects from the drag surfaces of the body of the vehicle, as well as the changes in relative torque on the airframe from the rotary wings at high vehicle angles. There are various aerodynamic effects that become significant at rotor descent and translation speeds that

are comparable to the induced wind speed, including blade flapping, translational lift, and toroidal vortexes [31, 113].

Specifically regarding quadrotor rotary-wing perching, the architecture of the system becomes important. Compared to helicopters, quadrotors are typically flatter; their thrust engines much lower to the ground when stationary. Additionally, the propellors extend outwards from the frame on each axis, elongating the distance between the front end of the quadrotor and the landing gear. These two aspects - lower height and longer distance to landing gear - mean that special care has to be taken that the perching maneuver is both accurate enough and aggressive enough to move the landing gear to the correct spot without snagging any of the rest of the quadrotor architecture on the landing pad.

6.3 Related Work

This section covers the state of the art in aggressive maneuvering as it relates to the focus of the work in this thesis, in order to highlight the contributions of the work presented. An initial broad overview of the progress in aggressive maneuvering is covered first, followed by implementations that specifically address the perching aerobatic maneuver, in the form of fixed-wing and rotary-wing aircraft. The section is concluded with the aspects of the contributions of this thesis that differentiate it from the current literature.

6.3.1 Aggressive Maneuvering

In the last decade, aggressive maneuvering, in the form of aerobatic maneuvers, has seen increasing quantities of successful demonstrations under automatic control. Shown on the MAV class, these vehicles have long shown such aerobatic capabilities under expert pilot manual control, and the challenge of obtaining such aggressive aerobatics under automatic control has only recently been successful in widespread maneuvers. Using apprenticeship learning techniques from expert pilot demonstrations combined with apprenticeship learning algorithms for modeling the dynamics of the helicopter and combining these algorithms using a receding horizon variation of linear quadratic control for nonlinear systems to design optimal controllers, aerobatic maneuvers such as in-place flips and rolls, loops, hurricanes,

chaos and tic-tocs have been successful on a traditional helicopter performed outside [114]. The STARMAC quadrotor was shown to be capable of aggressive turning at speeds around 8 m/s, reaching angles of 30 degrees, achieving control using aerodynamic modeling compensation of blade flapping and drag forces in order to counteract the impact of aerodynamic effects during such maneuvers [113]. The same STARMAC quadrotor system was shown to be capable of an autonomous backflip in outdoor environments using reachable sets generated using Hamilton-Jacobi game formulation for calculating provably safe switching conditions for the three stages of the maneuver [115]. Using a commercial quadrotor and the Vicon system, a quadrotor was demonstrated to be capable of autonomous multi-flips using a simple learning strategy, using optimization of parameters by inverting a Jacobian matrix based on the final state error following multiple iterations, using a model derived from first-principles [116]. With a similar quadrotor and Vicon system, tracking of fast moving trajectories was demonstrated with an LQR controller, while a robust H_∞ controller was simulated on a nonlinear model but was not capable of experimentally stabilizing the quadrotor [117]. Although extremely impressive, none of these works address the accuracy challenge, as all maneuvers are performed outside in open spaces or within highly accurate offboard vision systems, nor does it address the navigation and sensing needs of performing the maneuvers at certain locations or under certain external criteria.

6.3.2 Perching Using Fixed-Wing MAVs

Fixed-wing perching has seen only very recent research. A lightweight plane was implemented with a forward facing sonar and using an open loop state transition, controls to a pitch up configuration upon close proximity to a vertical perching spot, and is also capable of taking off again from the vertical wall [118]. This system uses a novel wall latching mechanism to latch onto vertical walls using splines. In different design, using model-based characterization and the Vicon system, a glider with only a motor controller the rear elevator was shown to be capable of perching on a string using an infinite-horizon optimal feedback controller setup [119]. Due to the small basin of attraction for the maneuver, this system only succeeded in perching about once every five trials. Incorporating a time varying

LQR trajectory controller, the glider was able to perch even with imposed variations in the starting conditions [120].

6.3.3 Perching with Rotary-Wing MAVs

Using a hobby traditional helicopter and Vicon, perching on inclined surfaces was demonstrated using velcro at low speeds [121]. In this approach, a state machine is used to transition between level flight and an open-loop perching maneuver. The velcro sheet on the landing pad allows the helicopter to be caught at low speeds mid-flight. With a commercial quadrotor and Vicon, aggressive perching on inclined, vertical, and inverted surfaces was demonstrated using velcro from short distances away, using a model derived from first principles with trajectory controllers defined by a sequence of segments, each with a goal state within a 12-state state space and based on parameter adaptation from measured errors after experimental testing [122].

The ultimate goal in the research presented here, in contrast to those above, is to close the navigation feedback loop based upon immediate sensing of the location of the perch, so that disturbances during the aggressive portion of the maneuver can be robustly handled, and to do so on a quadrotor using only onboard sensing.

6.4 Vision-Based Perching

For initial proof of concept and controller tuning, perching was performed using the camera as the only navigation/guidance sensor. The location of the perch was known, and the path determination for the perching maneuver was determined based upon this fore-knowledge. To accomplish this, the path to the perch, and the perch itself, was pre-mapped for location determination and to prevent navigation delays due to mapping updates. An example of the map generated for perching is shown in Figure 6.2.

6.4.1 Landing System

The landing surface is a hand-made wooden podium, about 2 feet high with a landing surface of 2 feet by 2 feet, making it roughly 9 times the area of the quadrotor landing gear,

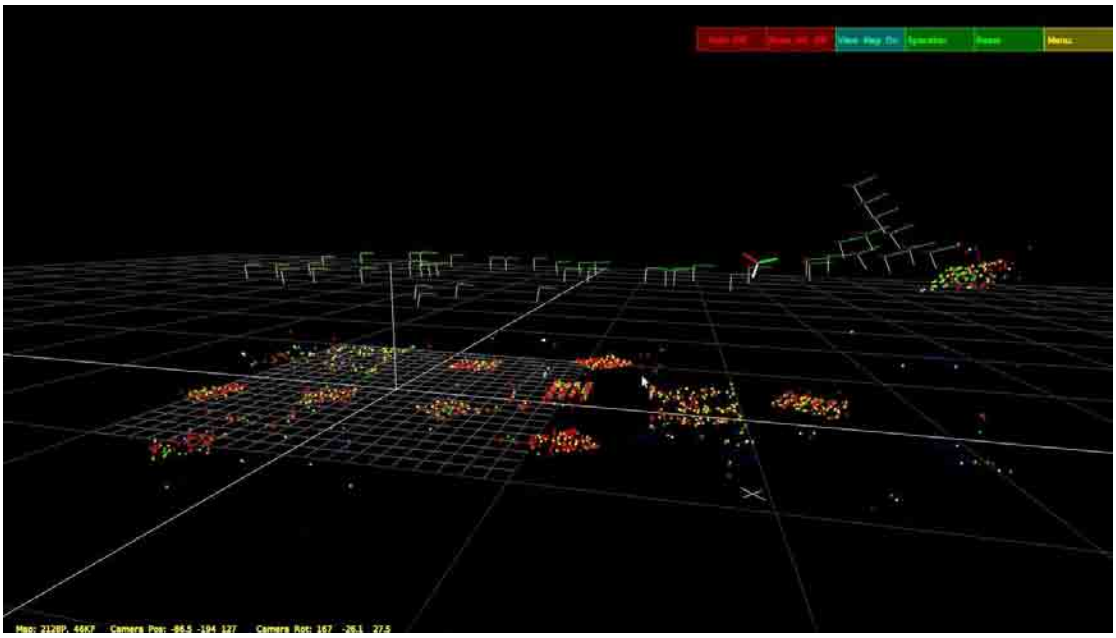


Fig. 6.2: Map example of the features making up the perch and the path.

and is similar to another perching setup [121]. The angle of the podium landing surface is adjustable for 30, 45, and 60 degree angles. The surface is covered with a half inch layer of high-density foam, both to provide a little bit of friction for landing, as well as protection of the quadrotor.

Initially, perching on a string was considered, but was determined to be unfeasible due to several reasons. Following a point landing on the string, the quadrotor would tip over. Even utilizing a type of hook in order to stay attached to the string, the quadrotor would not remain upright and possible fall to the ground. Just allowing such a hook to take the landing weight of the quadrotor was determined as too risky, since the custom quadrotor design does not allow easy access to the center structural part of the frame, limiting the safety of such a hook attachment. Additionally, landing on the string would prevent the showcase of taking off again, as can be performed from an inclined surface.

6.4.2 Controller

The controller used for perching is the nonlinear saturation controller described in Chapter 5. This controller is required due to the rapid dynamic transitions needed for the

perching maneuver, which must be limited to a predefined boundary that the linear PID controller would not be able to obtain, even with gain scheduling. Part of this is due to the noise within the tracking measurements from the navigation system from the high speed and quick transitions.

The only modification to the nonlinear saturation controller needed for perching is specifically allowing more control authority to the navigation system, required for performing such aggressive maneuvers. Without such a change in control authority, the maximum attitude command that the navigation system could require of the attitude system is roughly 10 degrees of change, which would not be enough to accomplish the initial high-speed translational flight nor the high angle of attack post-stall maneuver.

In order to modify the control authority of the nonlinear navigation controller, the saturation asymptotes are simply increased to an estimated and then empirically tuned value, based upon the needed desired angle for the speeds being achieved. The gains and saturation levels used for the maneuver are shown in Table 6.1. This saturation level adjustment specifically for the perching maneuver can be seen as a form of gain scheduling, but is really just a removal of a constraint on the traditional flight envelope which it does not typically achieve during normal maneuvers. This is similar to the way modern fighter jet controllers work, which have a safety system that prevents excessive maneuvering unless switched off by the pilot.

6.4.3 Maneuver Through Path Generation

Since the goal of the navigation system with the controller described above is simply to maintain the desired speeds and positions given to it, the perching maneuver can be accomplished simply by generating an appropriate path such that the navigation system will be required to perform the desired maneuver in the process of tracking the path given.

Such a path is characterized by high desired speeds for the initial fast translational movement requirement of the maneuver, and is followed by a transition to a zero desired speed at the point at which the maneuver must take place in order to reach the perch with a nearly zero velocity while also at a high angle of attack. The controller operation on the

Table 6.1: Perching nonlinear controller gains.

Gain Term	Notation	Value
Navigation x Proportional Asymptote Bound	$A_{kp}^{nav,x}$	100
Navigation x Proportional Growth Rate	$B_{kp}^{nav,x}$	0.04
Navigation y Proportional Asymptote Bound	$A_{kp}^{nav,y}$	200
Navigation y Proportional Growth Rate	$B_{kp}^{nav,y}$	0.013
Navigation x Derivative Asymptote Bound	$A_{kd}^{nav,x}$	90
Navigation x Derivative Growth Rate	$B_{kd}^{nav,x}$	0.042
Navigation y Derivative Asymptote Bound	$A_{kd}^{nav,y}$	360
Navigation y Derivative Growth Rate	$B_{kd}^{nav,y}$	0.0128
Navigation x Integral Gain	$ki_{sig}^{nav,x}$	0.002
Navigation y Integral Gain	$ki_{sig}^{nav,y}$	0.0035
Navigation y Desired Velocity	$\dot{(y)}_{nav}^{des}$	350 cm/s

generated path is the same as described in Section (4.10.4). Since the implementation of the path is for the quadrotor to perform the maneuver with one of the axes in line with the perch for simplicity of the saturation adjustment, the path controller simplifies to decoupled x, y, and correspondingly, roll and pitch controllers. Thus, the navigation controller from Equations (4.27) and (4.28) becomes

$$\begin{aligned} \ddot{x}^{des} &= k_{p,nav}^{x} \cdot (x_{pos}^{des} - x_{pos}) + k_{i,nav}^{x} \cdot (\int_{0}^{t} (x_{pos}^{des} - x_{pos}) dt) \\ &\quad -k_{d,nav}^{x} \cdot \dot{x}_{pos}, \end{aligned} \tag{6.1}$$

$$\ddot{y}^{des} = k_{p,nav}^{y} \cdot (y_{pos}^{des} - y_{pos}) + k_{d,nav}^{y} \cdot (\dot{y}_{pos}^{des} - \dot{y}_{pos}), \tag{6.2}$$

since the perch maneuver is completed along the y axis.

This controller is then implemented for a specified path which is generated according to the behavior it exhibits when executed on the quadrotor for the perching maneuver. This path is based upon the same generation described in Algorithm 4.1, and is essentially only used for the initial high-speed translational flight, with the incorporation of the state transition to the high angle of attack, post-stall maneuver for a rapid deceleration prior to landing on the perch. The path has one key difference from the path generation method of standard paths mentioned in Chapter 4, which is that the distance gain, k_d is increased in order to obtain larger proportional commands during the translational flight portion.

As the desired speed is increased, to values above 300 cm/s, and with a distance gain, k_d near 4 cm, over a translational distance to the perch of about 3 meters, the quadrotor actually begins to overpass the way points in the path due to its rapidly increasing speed from the initial commands. This is actually used advantageously for the perching maneuver, as the crossing point is reached, the command values from the controller are reduced over a short period of time until they become reversed. This then causes a rapidly increasing controller command in the opposite direction of travel, initializing the post-stall maneuver. Around this post-stall initialization, the path way points end, at which point the default response of the controller is to return to a hover.

The hover requirement is for $\dot{y}_{pos}^{des}$ to be zero, and y_{pos}^{des} to be the point on the map at which the hover was initialized. Since the quadrotor is still rapidly moving along and has only barely managed to begin to pitch up - the fastest speeds were noted just before the pitch up maneuver, meaning the crossing point along the path only reduces the acceleration to zero, but does not actually have enough time to reduce the velocity before the real high angle of attack is completed by means of the transition to the hover mode after the path length is completed.

The length of the path is thus tuned to enable the hover mode switch at the appropriate time, such that that quadrotor stays at the high angle of attack long enough to slow down to a speed at which the perch can be made, without giving so much time that the quadrotor begins to turn around and come back. A simple state transition is used upon reaching the perch, at which point the thrust is rapidly decreased and then turned off. The crossing point and transitions can be seen in Figure 6.3 which shows the x and y position vs desired plots during the maneuver. In the y plot, the path starts execution at time 7.5 sec, the crossing point of the path is reached at time 9.2 sec, and the transition to the hover occurs at time 9.75 sec. The quadrotor reaches the perch just before 10.5 sec, whereupon the program is terminated in order to turn off the motors.

During this maneuver, the system relies entirely on visual height control to maintain altitude. Without this capability, perching would not be an option for two reasons.

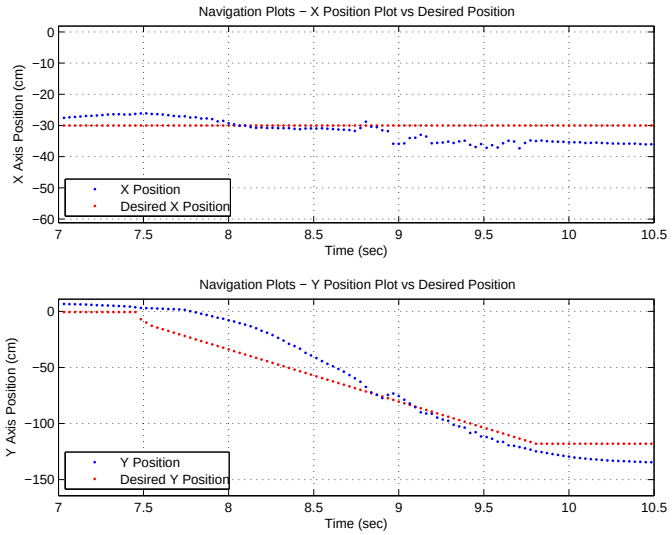


Fig. 6.3: Position vs desired for vision perch, x and y separate.

- The sonar gives more frequent outliers in the data points than the visual height despite the visual height running faster. This causes the quadrotor to try to rapidly adjust total thrust levels, in the end coupling disturbances over to the navigation response and thus reducing accuracy.

- When the quadrotor pitches up for the post-stall maneuver, the sonar is no longer looking at the ground, since in a room, there are objects and the other wall which it will see first, causing the sonar to read values much closer than the attitude of the quadrotor could account for. This is an obvious problem as the quadrotor approaches the perch, since the sonar will give a range of values as the quadrotor is passing over the perch, which will not be consistent enough just to determine the distance from the perch, let alone for using the sonar to determine height from the ground.

6.4.4 Results of A Priori Perch Knowledge

Perching using the path generation successfully perched on a 30 degree inclined surface, achieving angles of 10 degrees on the high-speed translational portion, and angles up to 15 degrees for the post-stall maneuver. Translational speeds of 3 m/s were achieved. The behavior of the attitude of the vehicle during the maneuver can be seen in Figure 6.4.

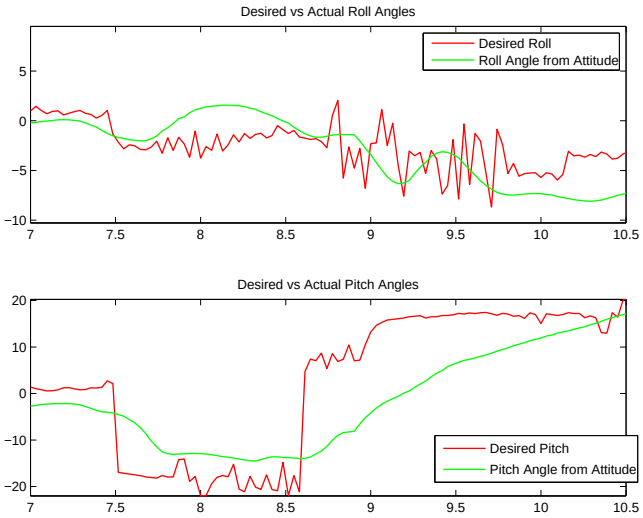


Fig. 6.4: Attitude performance during vision based perching.

Although the angles and speeds are not large, the indoor testing area was limited to only 4 meters, restricting the speeds, and thus the angles, of the maneuver. It is expected that these slower speeds allow for adequate trajectory tracking even with linearized models and PID control, as aerodynamic effects such as blade flapping and vehicle drag are not significant [113]. Speeds around 9 m/s were used for perching on the fixed-wing glider mentioned in Section 6.3 [120]. Although top speeds are not given, a speed of only 0.8 m/s normal to the perching surface are needed for the quick aggressive perch by a quadrotor using Vicon [122]. A sequence of action shots of the maneuver are shown in Figure 6.5.

6.5 Perching Using Guidance Sensing

To achieve the bird-like perching capability, not only must the vehicle have onboard navigation sensing capabilities for performing the perching maneuver, it must also be able to sense the perching location and perform the maneuver based upon this additional sensing information, without the need for prior knowledge of the perch or the path, as described in the implementation above. However, perception of the perch in a fast and accurate enough way to base the aggressive aerobatic perching maneuver on is not currently feasible with the sensors and processing methods available.

As a first step towards this autonomous aggressive perching without prior knowledge,

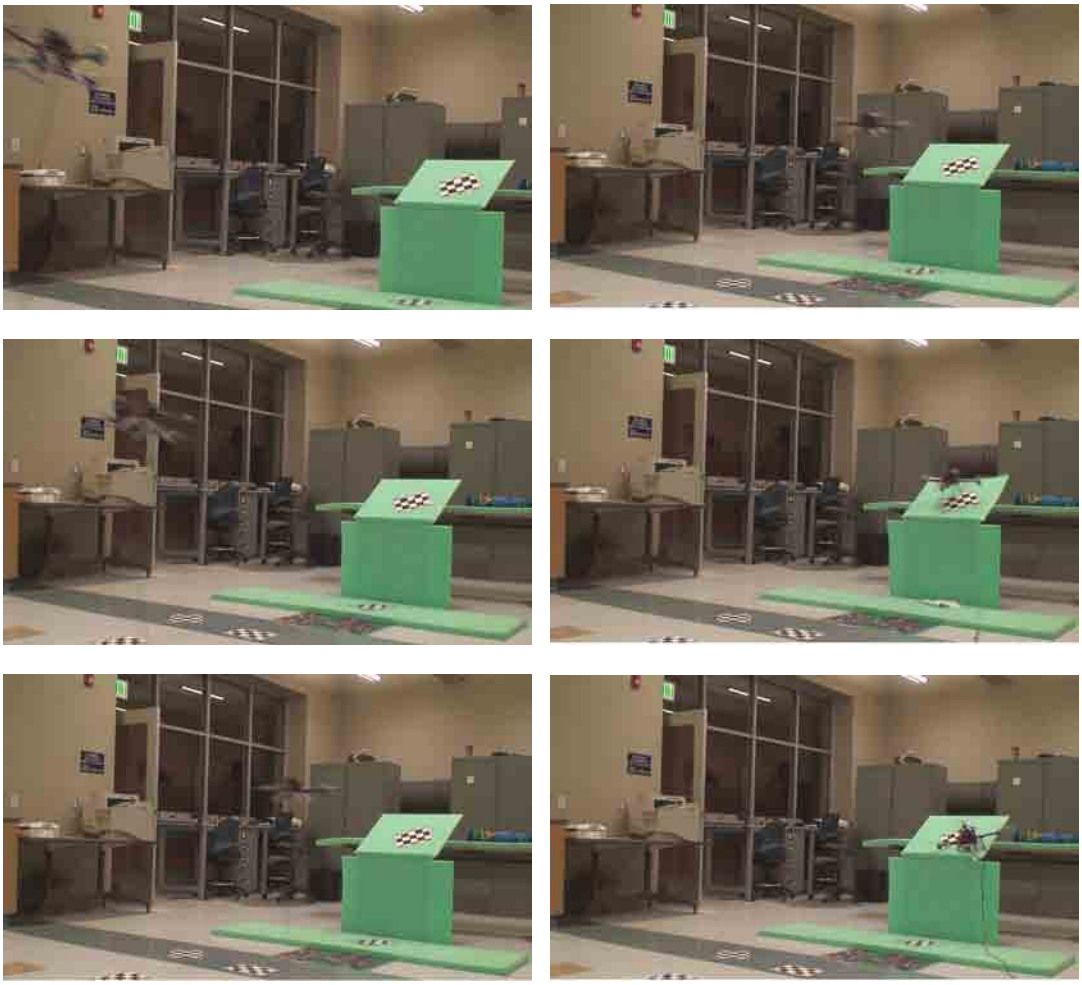


Fig. 6.5: Perching maneuver action shot sequence.

an onboard camera sensor is used that can detect IR LED blobs and return the pixel position of these blobs using an inbuilt system on chip. The placement of these IR LEDs is at the three critical points of the maneuver: the start of the high-speed translational movement, the dynamic transition to high angle of attack for post-stall deceleration, and the location of the perch for final tracking corrections and reducing thrust in order to land appropriately.

The overall setup and system are the same as described in the vision based perching, Section 6.4, with the difference being the use of a new guidance sensor and controller for obtaining the state transitions that were previously obtained using the pre-generated path.

6.6 Path and Landing Sensor

A camera extracted from a Nintendo WiiMote is used, owing to its low-cost, lightweight, low-power consumption, and specialized sensing capability [123]. The camera consists of a 1024 x 768 pixels Charged Coupled Device (CCD) sensor and a custom system-on-a-chip that is capable of tracking up to four IR light sources simultaneously. It reports the x and y positions of the IR light sources, or blobs, along with the estimated blob size, as a value ranging from one to six. These measurements can be obtained at a rate of up to 200 Hz over I^2C. The camera can detect IR blobs up to a distance of 5 m and has a field of view (FOV) of 41 degrees horizontal and 31 degrees vertical. Parameters such as the minimum and maximum blob size and camera gain can be set over I^2C. The gain parameter is related to the sensitivity of the camera and a gain of 255 was experimentally found to provide the best performance.

6.6.1 Implementation Details

The IR camera is found to be most sensitive to the 940 nm wavelength, so IR LEDs having peak emittance at 940 nm are used as markers. These markers are placed along the path of the perch, at specific points in order to achieve the appropriate dynamic transitions at the appropriate points in the path, as well as to provide outer guidance level control that is passed on to the navigation controller.

6.6.2 Sensor Interfacing

The camera can be interfaced with in two ways: communication via Bluetooth, or communication over I^2C [66,124]. Due to concerns of latency from using Bluetooth, the camera is interfaced directly over I^2C. Using schematics obtained from another implementation, a board was built that houses the camera and supporting components, shown in Figure 6.6 [124]. A diagram of the manner in which the camera is interfaced, as well as the way it is mounted on the quadrotor, are shown in Figures 2.6 and 2.9, respectively.

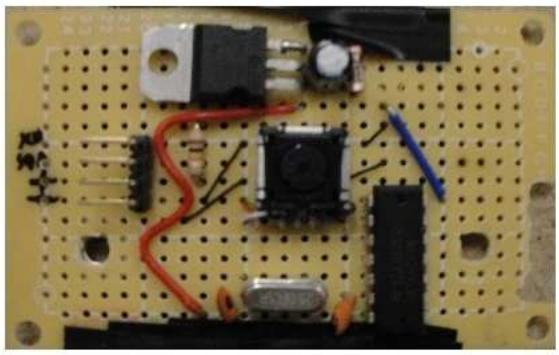


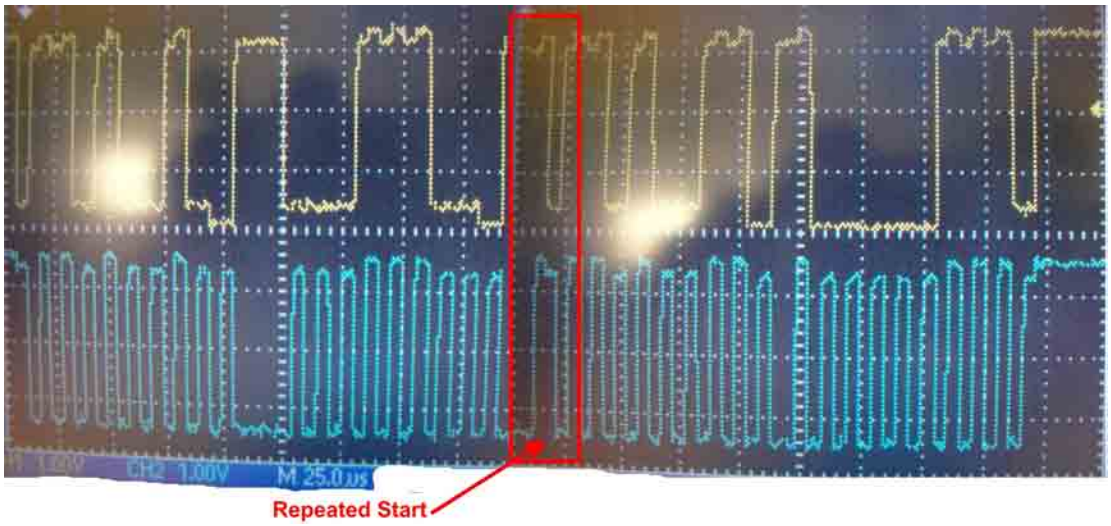
Fig. 6.6: IR camera along with supporting circuitry mounted on a board.

6.6.3 Issues

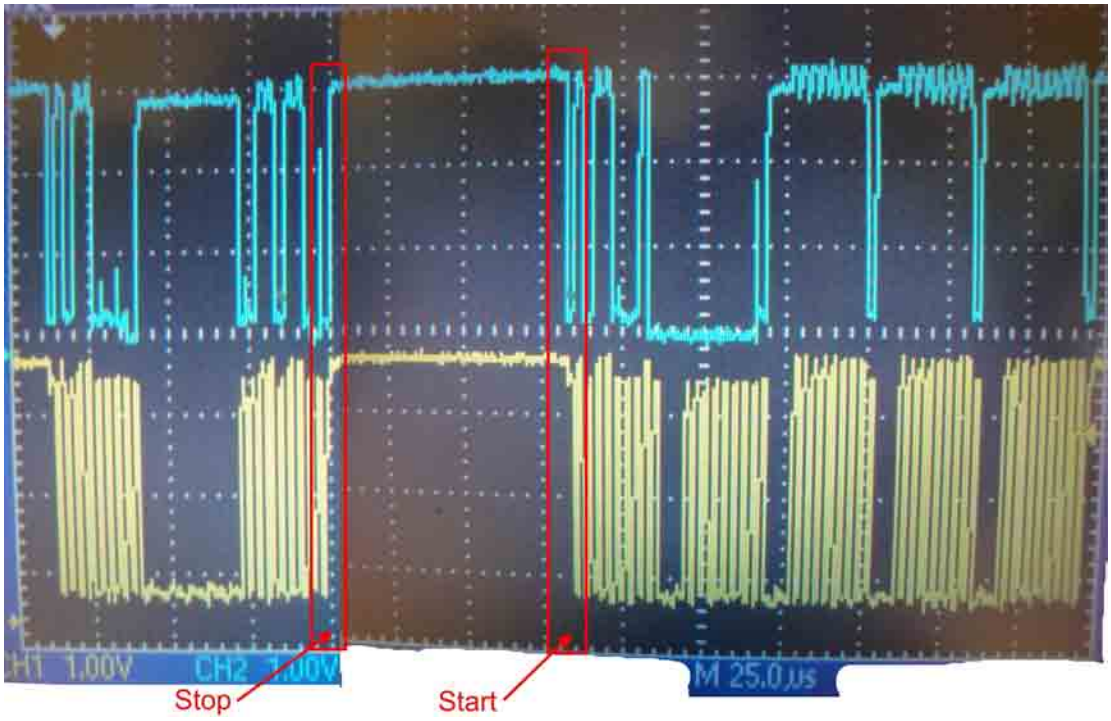
The camera gave several implementation problems, most importantly its I^2C uniqueness. The camera does not communicate over I^2C the way the protocol is intended to function, and how the Gumstix/Robostix software is set up. The protocol function in question is that when performing a read command, I^2C protocol is to first execute a write command, sending the address of the device to read from, then send a read command after a repeated start action, after which the specific register to be read is sent. The IR camera would hang and shut down the whole I^2C bus for any repeated start signal, requiring a full stop and start instead. These two implementations are shown in Figure 6.7, which show the way that the Gumstix reads from the gyro sensor using a repeated start, as well as the way the Gumstix has to read from the IR camera using a stop and then start [125].

6.7 Guidance Control System Architecture

This guidance-based perching aggressive maneuver uses the same nonlinear controller as discussed in the vision-based perching. A third outer control loop is added, set up as a guidance loop, shown in Figure 6.8, which changes the desired position of the quadrotor based upon the measured position of the located IR blob and also initiates the appropriate state transitions. A feedback loop is required due to the quantized and noisy blob location data returned by the camera and the lack of true relative positioning information. This general setup is modified based on each blob location. Three blob locations are used: one is



(a) This gyro read is for gyro address 0x69, register to read from of 0x1C, and value returned of 0x83. The repeated start is indicated on the diagram, and is characterized by a high to low transition of the SDA line while the SCL line without needing to perform a stop condition first.



(b) This IR camera read is for camera address 0x58, register to read from of 0x36, and return values of 0x00, 0xFF, 0xFF, and 0xFF. The separate stop and start conditions are indicated. The stop is a low to high transition on the SDA line while the SCL line is high. The start condition is high to low transition on the SDA line while the SCL line is high.

Fig. 6.7: Gyro vs IR camera I^2C communication.

used at the desired location for the start of the high-speed translational component, another for the transition to the high-angle of attack post-stall maneuver, and the last on the perch itself.

For the start of the perching maneuver, the quadrotor must first detect the x location of the perch so that it can start the high-speed translational flight portion heading in the correct direction. This detection takes place over the first IR LED, located on the ground at the point where the quadrotor initiated the high-speed maneuver for the vision-based perching. Since the translational dynamics of the quadrotor are relatively slow while hovering, an integrator alone is sufficient for convergence close enough to the desired position for the maneuver. The output of the integrator loop

$$x_{\text{offset}} = k^x_{i,IR} \cdot \int_0^t (x_{des} - x), \tag{6.3}$$

is the offset for the desired x position, x^{des}_{pos}, and added directly within the navigation loop, with the desired position being set to the current position upon initial detection of the IR blob. This controller runs for 10 seconds, giving the quadrotor time to converge the desired position to the actual position of the IR LED, as well as for the quadrotor to settle over the desired point. After this time, the quadrotor switches to the high-speed translational flight portion of the perching maneuver, and has the same control characteristics as for the vision-based perching.

Another LED is placed on the ground at the point where the quadrotor must begin the high angle of attack post-stall maneuver. This LED is used purely for the state transition, as the speed of the quadrotor, plus the already determined desired x position, eliminates the need for guidance control using this blob. Upon detection of this blob, the desired y position is set to the current position, and the desired y velocity is set to zero. The position of this blob is not quite the same as the point at which the path ended for the vision-based perching, nor the point at the crossing point mention in Section 6.4.3. This is because the blob is used as an immediate transition, unlike the path which allowed an initial gradual transition before the abrupt one. Implementing a smooth transition based upon the initial

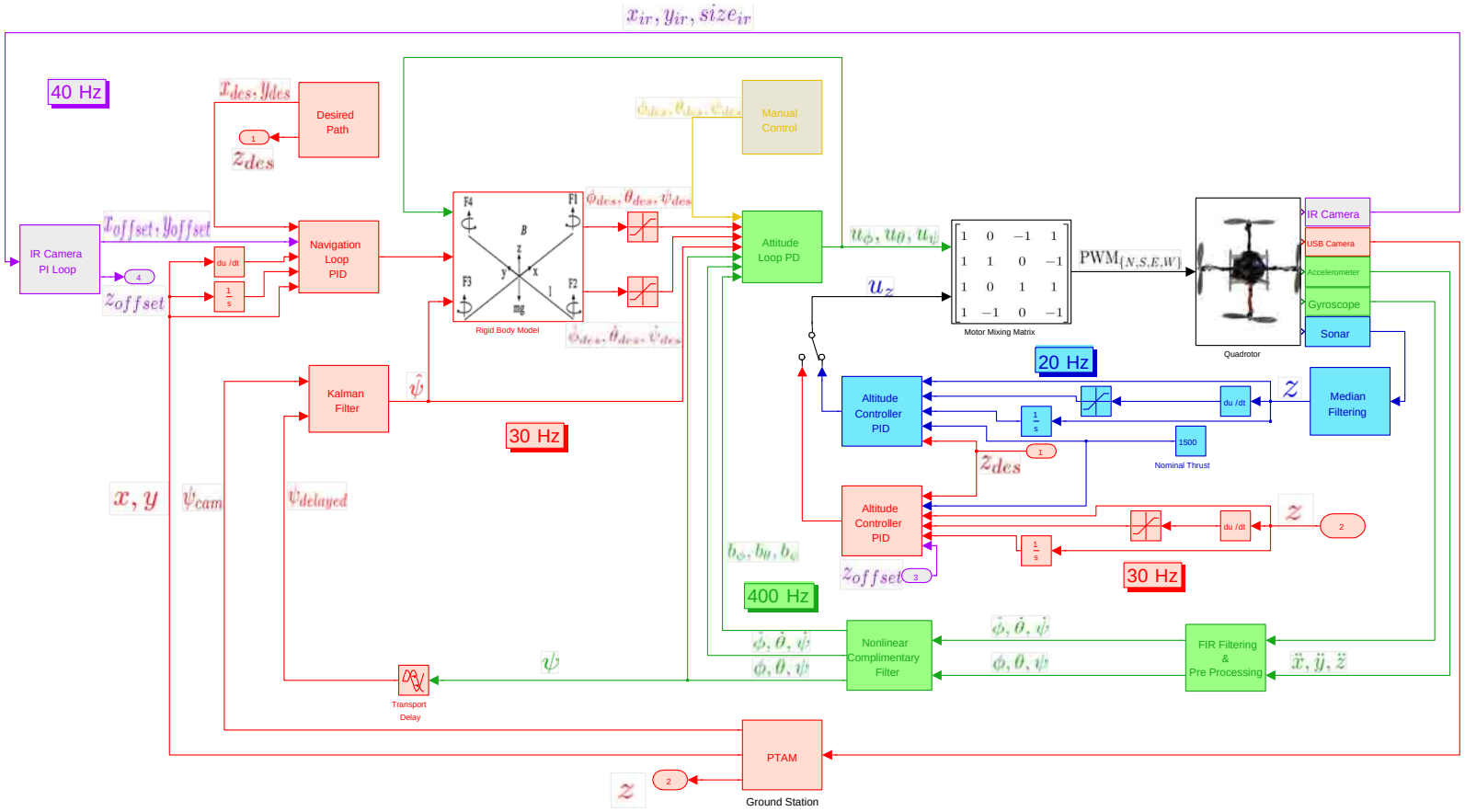


Fig. 6.8: Perching control system block diagram. The green blocks and wires indicate the attitude control system; the blue blocks and wires are for the sonar altitude controller; red indicates the navigation system utilizing the SLAM algorithm on the ground station, which returns position information. The purple refers to the IR camera perching guidance controller. A backup manual controller for safety is shown in a tan color.

blob detection could be explored in future work for improving the behavior of the quadrotor during the maneuver.

Shortly after passing over the second blob, the quadrotor is now rapidly deceleration by having a high angle of attack. Since the dynamic transition was initiated at the right moment, the quadrotor is facing the blob that is located on the perch, and is only a short distance away. The last blob is then used to set the final desired y position, using an integrator as for blob one, so

$$y_{\text{offset}} = k^{y}_{i,IR} \cdot \int_{0}^{t} (y_{des} - y), \tag{6.4}$$

with the same setup as for Equation (6.3). Although at this point, the translation dynamics of the quadrotor are not slow, an integrator is still used for the guidance controller. This is because, ultimately, the job of the guidance controller is to tell the navigation controller where to go. If a fast dynamics controller were used, such as PD, the guidance controller would be overriding the navigation controller, preventing the navigation controller from accurately tracking to the desired location. Since the guidance loop is capable of running even faster than the navigation loop, this problem can be compounded. Instead, the job of the guidance controller is to converge to an accurate estimate of the absolute desired position, and feed that information to the navigation controller so that it can perform the navigation portion.

During this final perch location tracking, the blob size information from the camera is used to reduce the thrust from the motors for a gentle landing. When a blob size of 5 is detected, the motors are commanded to turn off, which is when the quadrotor is just over the perch and has just enough momentum to make it the final distance for the landing.

6.7.1 Latency

System latencies and delays can contribute to instability and poor performance, highly problematic during such a fast maneuver that requires such accuracy. For reducing overhead, the sonar and IR camera controllers are not run simultaneously. The navigation and

attitude system delays are updated to incorporate the guidance controller, as shown in Figure 6.9.

6.8 Perching Results Using Onboard Sensing

Using onboard sensing for guidance level control enabled successful perching on a 30 degree inclined surface, with similar attitude and flight behaviors of the a priori vision perching, and so the action sequence in Figure 6.5 is still representative of the guidance-based perching. Even though no significant difference is shown in the behavior of the perch between guidance-based perching and a priori perching, demonstrating successful perching using real-time sensing of the markers for guidance of the navigation system brings the system capabilities a step closer towards autonomous perching without need of foreknowledge of pre-installed markers. Since the markers are simulating output from a sensor system that is capable of dynamically detecting the perch, this online detection and control is a good representation of the end perching goal.

The outputs of the IR camera for the detection and control of the first and third blobs are shown in Figures 6.10 and 6.11, respectively.

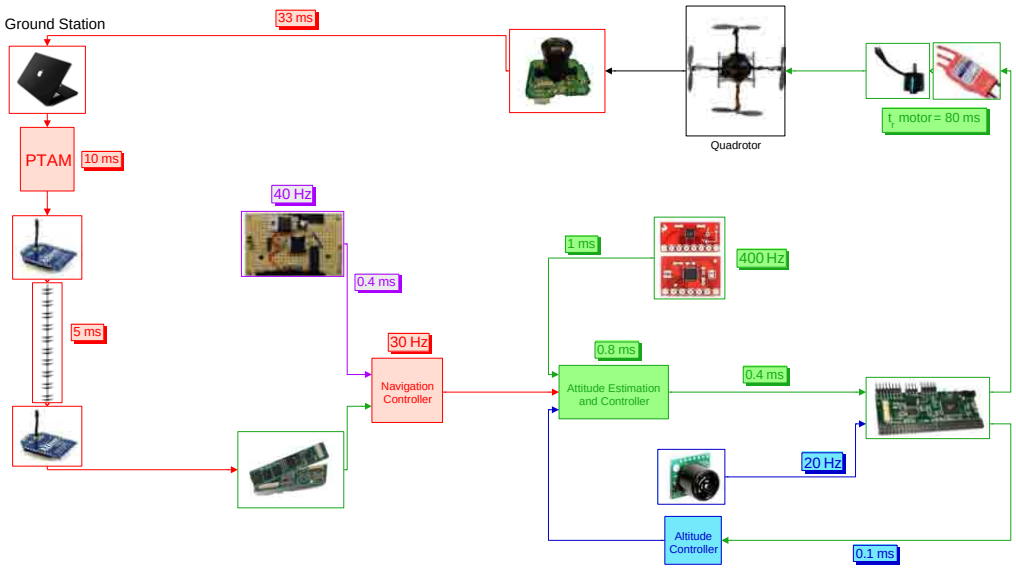
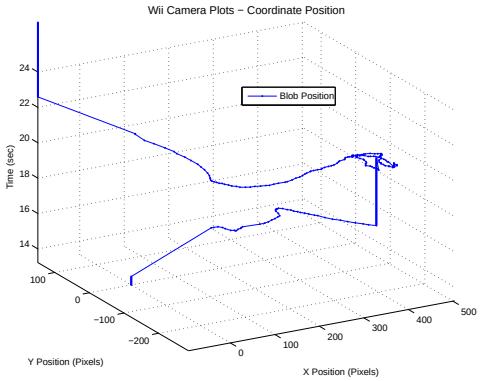
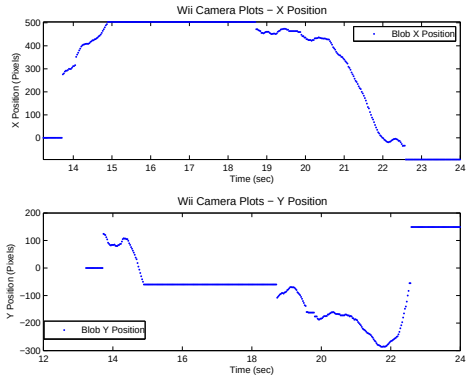


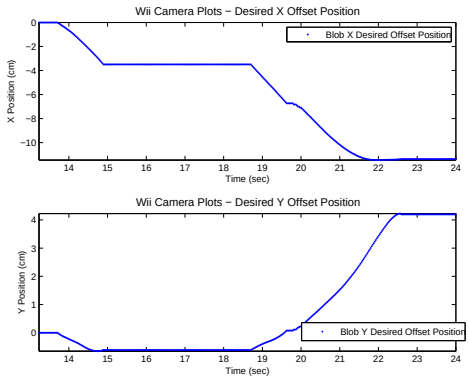
Fig. 6.9: Perching system latency diagram by component and communication.



(a) Blob one coordinate position versus time.



(b) Blob one x and y positions, separate over time.

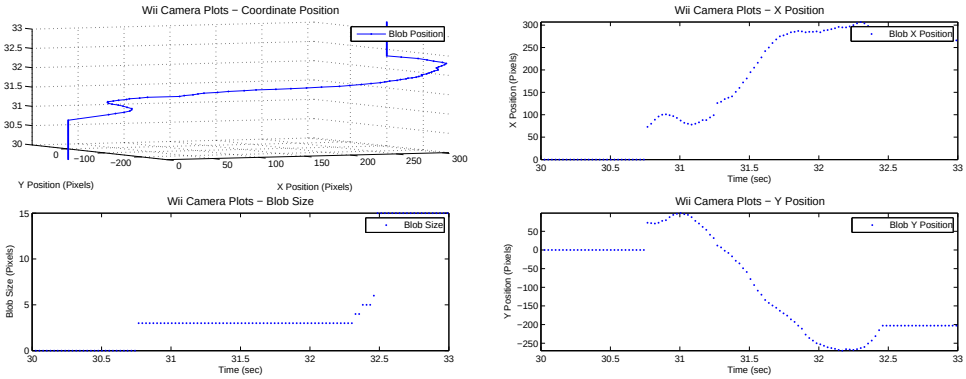


(c) Blob one x and y offsets to desired positions.

Fig. 6.10: Blob one measurements.

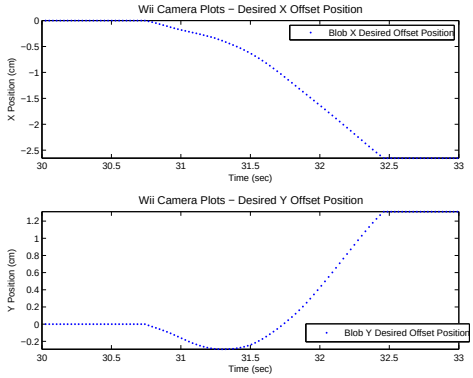
6.9 Chapter Contributions

The capabilities of the system outlined in the previous chapters were stretched to levels beyond the original hover region design, showing the robustness and effectiveness of the platform, attitude and navigation systems. The navigation control method developed in Chapter 5 enabled more than just stability when far from the desired position, but to actually generate robust aggressive maneuvering capabilities. A perching controller is proposed that uses saturation scheduling, state dependent path transitioning, and guidance-level tracking control of measurements returned from an onboard IR blob camera. The quadrotor is then shown to successfully perch on a 30-degree inclined surface.



(a) Blob three coordinate position versus time and blob size.

(b) Blob three x and y positions, separate over time.



(c) Blob three x and y offsets to desired positions.

Fig. 6.11: Blob three measurements.

6.10 Chapter Summary

This chapter has demonstrated aggressive maneuverability using a quadrotor, in the form of a perching aerobatic on an inclined surface. In order to obtain this capability, the fully developed system covered in Chapters 2, 3, 4, and 5 was required in order to obtain effective and robust dynamic transitioning between speeds and attitude. In addition, a sensor was added along with a guidance controller to implement the perching maneuver without the use of a priori knowledge of the environment. In order to generate effective aggressive maneuvering for a perching aerobatic, the saturation limits were scheduled for the maneuver, with a path dependent controller using state transitions based upon input from the onboard IR blob camera.

Chapter 7

Conclusion and Future Work

This concluding chapter provides a summary of the work presented in these thesis, and the contributions within, discussing what the research means in regards to practical capabilities of MAVs. A critical examination of the assumptions and limitations of the system are addressed, followed by describing future work that would improve the capabilities and practical implementation of the system presented.

7.1 Summary

This thesis covered the development of a complete quadrotor vehicle capable of autonomous indoor navigation using only low-cost onboard sensors. The full build up and specifics, both mechanical and electrical, of a customized quadrotor platform are covered with analyses of specific components and changes required to make flight possible. The quadrotor physical system was shown to utilize low-cost components when compared with other available quadrotors, while still offering similar, if not improved, capabilities. Sensor filtering and attitude estimation requirements were presented for obtaining reasonable attitude measurements using low-cost sensors with the control methods for each portion of the quadrotor attitude covered, culminating in a successful attitude stabilized hover flight. The upper-level navigation system that allows for indoor navigation in unknown environments is presented, covering the framework for simultaneous localization and mapping using a monocular camera and the navigation control system, ultimately resulting in successful vision-based height control, a tight hover and accurate trajectory tracking. A novel nonlinear control approach to application specific problems with the linear control method on the altitude and navigation systems was presented with a favorable comparison between the linear and nonlinear controllers. Finally, the completely designed system was used to

implement an aggressive perching maneuver using an outer guidance controller based on only onboard sensors.

7.2 Contributions Summary

This thesis has shown the implementation of an autonomous, indoor, navigation-capable quadrotor. In particular, the following are contributions of this thesis:

- The complete mechanical and electronic physical system design and adaptations necessary for autonomous flying on a quadrotor using only low-cost components,
- Attitude estimation techniques required for attitude stabilization on a heavy quadrotor with noisy sensor measurements,
- Attitude and navigation controllers implementation and experimental validation on a custom low-cost quadrotor,
- Adaptation of a real-time state-of-the-art visual simultaneous localization and mapping algorithm for use on a custom quadrotor,
- Attitude and navigation system fusion methods for improved attitude and path tracking,
- A nonlinear controller for altitude and navigation that improves stability and tracking,
- Navigation control adjustments for obtaining accurate aggressive maneuvering,
- An onboard sensing approach to autonomous perching using a guidance level feedback and state transition controller.

7.3 Critical Analysis

Despite the successes of the work presented in this thesis, the problems of indoor navigation and fully autonomous capabilities for a quadrotor are far from solved. The contributions presented in this thesis push the edge of state of the art quadrotor functionalities,

however, application specific assumptions and some general limitations of the system are necessarily present with a research-based rapid-prototype system.

7.3.1 Assumptions

Assumptions here are referred to as system aspects that are assumed to be true for the conclusions of this thesis to be obtained, however these will not hold true for all flight envelopes that the system might be exposed to, and not including them may reduce capabilities under certain conditions. Addressing these assumptions in future work will broaden the scope of what types of functions the quadrotor system can perform.

- Aerodynamics and other system dynamics have negligible effects on flight.
- The quadrotor will not be commanded to flip over during flight or otherwise near large angles.
- Low altitude flight operations will not occur.
- Navigation does not require explicit loop closing for effective tracking and mapping.
- Environments will contain enough usable features on the ground.
- The environment is static.

7.3.2 Limitations

Limitations in this context refer to restrictions on the extent of the capabilities of the quadrotor due to the way the system is implemented. Incorporating changes to the architecture of the system to eliminate these limitations would enable more effective use of the quadrotor vehicle.

- Poor outdoor localization, due to lack of GPS in this prototype, so limited to vision only navigation capability.
- Requires close proximity to ground station due to off-board navigation.

- Reduced stabilization quality due to wired control, so hanging cable puts outside force on the vehicle.
- Short flight time due to small battery and large power consumption from poor weight distribution and amount.
- Must take off in a pre-mapped location.

7.4 Future Work

The approach and success for indoor autonomous quadrotor navigation discussed in this thesis address some of the problems of the current state of the art, while also opening up new areas for future research. These new areas, coupled with the limitations of the developed system, lead to many new avenues for future work. These suggestions can be broken down into several categories based upon the subsystem that they affect: improvements to the physical architecture, greater robustness and stabilization for the attitude, extensions and improvements to the navigation system, and finally perching aerobatic maneuvering changes for better performance in agility and accuracy.

7.4.1 Physical Architecture Improvements

Physical architecture improvements address the fact that the vehicle is a prototype system, utilizing shortcuts in the building process for timeliness, using less than perfect components, or discovering problems with the chosen parts.

- Improve weight and balance by reducing component volume by going to surface mount PCB and components.
- Implement custom made components of lighter weight and better function, such as for the landing gear.
- Distribute costs more efficiently, so improved quality can be obtained on the important components for a robust system, such as with the motors.

- Improve mechanical damping by determining vibration frequencies and amplitudes and utilizing materials and components specific to this application.
- Provide better frame structure stiffness through cross braces, improved materials or better mounting techniques.
- Enable battery replacement without power interruption to the onboard systems, improving flight operational consistencies and testing throughput.
- Modify the frame for a larger battery, or additional batteries for combined use.
- Design battery voltage reading circuitry and software for better power compensation control for altitude regulation, as well as for preventing over usage of the battery or risking flight times exceeding available battery.

7.4.2 Attitude Stabilization and Disturbance Rejection

Improvements to the attitude stabilization system involve hardware or software changes that improve the capabilities of the system, or eliminate flight robustness problems.

- Separate the low-level attitude controller from the upper navigation controller.
- Use a microcontroller for the attitude system so that it can run at a faster rate, perform more efficient computation, have direct control of the motors, and widen the options for attitude sensing since the possibility of onboard sampling becomes feasible.
- Perform more accurate system identification for better model and control.
- Implement a takeoff controller that does not depend on foreknowledge of the nominal thrust, eliminating operator errors and reducing operator duties.
- Implement attitude control in quaternions in order to perform flipping maneuvers without having to handle the singularities involved with Euler angles.
- Improve low altitude attitude stabilization, possibly through gain scheduling or other controller implementations, to mitigate ground effect oscillations.

- Improve stabilization by implementing D^2 term using differentiated Gyro rates.
- Model effect of angular tilt on maintaining height.
- Include aerodynamic modeling for improved altitude, attitude, and navigation.

7.4.3 Navigation System

The navigation system includes both the SLAM algorithm as well as the controller. Making improvements to either of these two components will yield better navigation capabilities.

- Put the navigation SLAM processing onboard the quadrotor.
- Implement onboard path planning, using uncertainty covariance of mapping and tracking to determine the places that the navigation system can most accurately localize, for proper planning within the information space.
- Utilize modeling for translational movement to dynamically generate the way-point assignments based on the path planning algorithm.
- Implement outdoor capable navigation using GPS and magnetometer.
- Use a forward facing camera to map and localize for better feature availability as well as object recognition, object avoidance, and navigation in cluttered environments.
- Improve the SLAM algorithm to account for:
  - Larger maps,
  - Perform explicit loop closing for robust tracking,
  - Improve speed of mapping and tracking algorithms.

7.4.4 Perching Aerobatic Maneuver

The perching maneuver uses the full capabilities of the quadrotor system, along with an additional sensor for obtaining perch-sensing capabilities. Improvements in the base

system, perch sensing or perching control would allow more aggressive, accurate and robust perching.

- Measure and model the aerodynamic effects on the quadrotor of the dynamic transition from high-speed translational flight to a high angle of attack post-stall configuration for better control and perching robustness.
- Utilize additional sensing options for more realistic perch path and perch sensing.
- Have dynamic saturation scheduling based upon the desires of the guidance controller rather than upon separate state transition inputs.

7.5 Conclusion

Fully autonomous MAVs, capable of both indoor and outdoor navigation, can provide significant benefits. Enabling goal-directed mission planning with minimal need for operator oversight is a desirable capability, as information and intelligence gathering through close observation can be obtained while allowing human operators to perform more strategic level planning and activities, vastly improving the productivity of the group. Quadrotor VTOL MAVs can provide many of these benefits, once navigation, safety, cost, and other factors are improved. Outdoor capabilities for quadrotors have become more useful and widespread, although there is still much room for improvement in accuracy and robustness. Indoor navigation options are severely limited, although research in the area has been rapidly expanding. Improving capabilities while reducing costs are important goals towards successful completion of a fully capable quadrotor. With the contributions of this thesis, the gap towards this final goal is reduced. A complete quadrotor built from scratch and using very low-cost sensors and components has been shown to be able to autonomously stabilize and navigate itself in unknown and unstructured environments using only onboard sensors. MAVs that are limited in their operational range are not as desirable, so to this end, the quadrotor was shown to be capable of aggressive and accurate path following by the demonstration of a high speed perching maneuver.

References

[1] D. Lyon, "A military perspective on small unmanned aerial vehicles," *IEEE Instrumentation and Measurement Magazine*, vol. 7, no. 3, pp. 27–31, Sept. 2004.

[2] General Atomics Aeronautical Systems, Inc., "Predator," [http://www.ga-asi.com], 2011.

[3] AeroVironment, Inc., "Raven," [http://www.avinc.com], 2011.

[4] S. Bouabdallah, P. Murrieri, and R. Siegwart, "Towards autonomous indoor micro VTOL," *Autonomous Robots*, vol. 18, pp. 171–183, 2005.

[5] N. Bouabdallah and B. Sericola, "A simple priority mechanism for shared-protection schemes," *IEEE Communications Letters*, vol. 11, no. 2, pp. 197–199, Feb. 2007.

[6] Ascending Technologies, "AscTec Hummingbird," [http://www.asctec.de], 2011.

[7] MikroKopter, [http://www.mikrokopter.com], 2011.

[8] Quanser, "Qball-X4," [http://www.quanser.com], 2011.

[9] Microdrones GmbH, [http://www.microdrones.com], 2011.

[10] Draganfly Innovations, "Draganflyer V Ti Pro," [http://www.rctoys.com], 2010.

[11] Aeryon Labs, Inc., "Aeryon Scout," [http://www.aeryon.com], 2011.

[12] AeroQuad and ArduPilot, "ArduCopter open source hobby quadrotor," [http://diydrones.com/profiles/blogs/announcing-arducopter-the], 2011.

[13] S. Grzonka, G. Grisetti, and W. Burgard, "Towards a navigation system for autonomous indoor flying," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2878–2883, May 2009.

[14] A. Bachrach, R. He, and N. Roy, "Autonomous flight in unstructured and unknown indoor environments," in *Proceedings of the European Conference on Micro Air Vehicles (EMAV)*, 2009.

[15] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based MAV navigation in unknown and unstructured environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 21–28, May 2010.

[16] P. Bristeau, P. Martin, E. Salaün, and N. Petit, "Role of propeller aerodynamics in model of quadrotor UAV," in *Proceedings of the IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pp. 683–688, Aug. 2009.

[17] P. Pounds, R. Mahony, and J. Gresham, "Towards dynamically favourable quad-rotor aerial robots," in *Proceedings of the ARAA Australasian Conference on Robotics and Automation (ACRA)*, 2004.

[18] P. Pounds and A. Dollar, "Hovering stability of helicopters with elastic constraints," in *Proceedings of the ASME Dynamic Systems and Control Conference (DSCC)*, 2010.

[19] S. Bouabdallah and R. Siegwart, "Full control of a quadrotor," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 153–158, Nov. 2007.

[20] A. Tayebi and S. McGilvray, "Attitude stabilization of a VTOL quadrotor aircraft," *IEEE Transactions on Control Systems Technology*, vol. 14, no. 3, pp. 562–571, May 2006.

[21] hexTronik Limited, "Turnigy Plush - 30A ESC," [http://www.hobbyking.com], 2011.

[22] hexTronik Limited, "Hacker Style KDA20-22L," [http://www.hobbyking.com], 2011.

[23] InvenSense, Inc., "ITG-3200 3-Axis Gyroscope," [http://www.invensense.com], 2011.

[24] Bosch Sensortec GmbH, "BMA180 3-Axis Accelerometer," [http://www.bosch-sensortec.com], 2011.

[25] Maxbotix, Inc., "LV-MaxSonar-EZ2," [http://www.maxbotix.com], 2011.

[26] Gumstix, Inc., "Gumstix Verdex Pro XL6P COM," [http://www.gumstix.com], 2011.

[27] Gumstix, Inc., "Netpro-vx," [http://www.gumstix.com], 2011.

[28] Gumstix Inc., "Wi-Fi Module FCC," [http://www.gumstix.com], 2011.

[29] Gumstix, Inc., "Robostix-TH," [http://www.gumstix.com], 2011.

[30] Atmel Corporation, "ATmega128 Microcontroller," [http://www.atmel.com], 2011.

[31] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference (GNC)*, Aug. 2007.

[32] Align Corp., Ltd., "T-Rex 600 Landing Gear," [http://www.align.com.tw], 2011.

[33] RCgroups, "Common BL Motor Comparison Table," [http://www.rcgroups.com], 2011.

[34] Landing Products, "10x47 Slow Flyer Propeller," [http://www.apcprop.com], 2011.

[35] P. Pounds, R. Mahony, and P. Corke, "System identification and control of an aerobot drive system," in *Proceedings of the IEEE Conference on Information, Decision and Control (IDC)*, pp. 154–159, Feb. 2007.

[36] National Instruments Corporation, "Labview," [http://www.ni.com], 2011.

[37] P. Pounds, R. Mahony, and P. Corke, "Modelling and control of a quad-rotor robot," in *Proceedings of the ARAA Australasian Conference on Robotics and Automation (ACRA)*, 2006.

[38] D. Gurdan, J. Stumpf, M. Achtelik, K. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 kHz," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 361–366, Apr. 2007.

[39] Quax, "BLDC Motor ESC Firmware," [http://home.versanet.de/b̃-konze], 2011.

[40] Atmel Corporation, "ATmega8 Microcontroller," [http://www.atmel.com], 2011.

[41] Atmel Corp., "AVR Studio 4," [http://www2.atmel.com], 2011.

[42] Digi International, Inc., "Xbee-PRO 802.15.4 Module," [http://www.digi.com], 2011.

[43] V. Ghadiok, "Autonomous aerial manipulation using a quadrotor," Master's thesis, Utah State University, Logan, Utah, 2011.

[44] Sparkfun Electronics, [http://www.sparkfun.com], 2011.

[45] Gumstix, Inc., "Tweener," [http://www.gumstix.com], 2011.

[46] Advance Energy, Inc., "TP2600-3SPL Battery," [http://www.thunderpowerrc.com], 2011.

[47] National Semiconductor Corporation, "LM1084 Low-Dropout Regulator," [http://www.national.com], 2011.

[48] STMicroelectronics, Inc., "LD1117V33 LDO Regulator," [http://www.st.com], 2011.

[49] Parrot SA., "AR.Drone," [http://ardrone.parrot.com], 2011.

[50] Ascending Technologies, "AscTec Pelican," [http://www.asctec.de], 2011.

[51] P. Castillo, R. Lozano, and A. Dzul, "Stabilization of a mini rotorcraft with four rotors," *IEEE Control Systems Magazine*, vol. 25, no. 6, pp. 45–55, Dec. 2005.

[52] N. Guenard, T. Hamel, and V. Moreau, "Dynamic modeling and intuitive control strategy for an "x4-flyer"," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, vol. 1, pp. 141–146, June 2005.

[53] J. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Systems Magazine*, vol. 28, no. 2, pp. 51–64, Apr. 2008.

[54] J. Colorado, A. Barrientos, A. Martinez, B. Lafaverges, and J. Valente, "Mini-quadrotor attitude control based on hybrid backstepping and Frenet-Serret theory," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1617–1622, May 2010.

[55] S. Bouabdallah and R. Siegwart, "Backstepping and sliding-mode techniques applied to an indoor micro quadrotor," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2247–2252, Apr. 2005.

[56] P. Martin and E. Salaun, "The true role of accelerometer feedback in quadrotor control," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Dec. 2009.

[57] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, pp. 35–45, 1960.

[58] N. Metni, J. Pflimlin, T. Hamel, and P. Soueres, "Attitude and gyro bias estimation for a flying UAV," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1114–1120, Aug. 2005.

[59] R. Mahony, T. Hamel, and J. Pflimlin, "Nonlinear complementary filters on the special orthogonal group," *IEEE Transactions on Automatic Control*, vol. 53, no. 5, pp. 1203–1218, June 2008.

[60] N. Guenard, T. Hamel, and R. Mahony, "A practical visual servo control for an unmanned aerial vehicle," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 331–340, Apr. 2008.

[61] R. Mahony, T. Hamel, and J. Pflimlin, "Complementary filter design on the special orthogonal group SO(3)," in *Proceedings of the IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pp. 1477–1484, Dec. 2005.

[62] InvenSense, Inc., "IDG-500 2-Axis Gyroscope," [http://www.invensense.com], 2011.

[63] Analog Devices, "ADXL335 3-Axis Accelerometer," [http://www.analog.com], 2011.

[64] LEGO Group, [http://www.lego.com], 2011.

[65] C. Dobler, "Development of flight hardware for a next generation autonomous micro air vehicle," Master's thesis, Swiss Federal Institute of Technology - Zurich, Zurich, Switzerland, 2010.

[66] T. Lee, M. Leok, , and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5420–5425, Dec. 2010.

[67] H. Romero, R. Benosman, and R. Lozano, "Stabilization and location of a four rotor helicopter applying vision," in *Proceedings of the AACC American Control Conference (ACC)*, June 2006.

[68] E. Altuğ, J. P. Ostrowski, and C. J. Taylor, "Control of a quadrotor helicopter using dual camera visual feedback," *International Journal of Robotics Research*, vol. 24, pp. 329–341, May 2005.

[69] S. Bouabdallah, A. Noth, and R. Siegwart, "PID vs LQ control techniques applied to an indoor micro quadrotor," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2451–2456, Sept. 2004.

[70] T. Madani and A. Benallegue, "Backstepping control for a quadrotor helicopter," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3255–3260, Oct. 2006.

[71] F. Hoffmann, N. Goddemeier, and T. Bertram, "Attitude estimation and control of a quadrocopter," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1072–1077, Oct. 2010.

[72] M. Guisser and H. Medromi, "A high gain observer and sliding mode controller for an autonomous quadrotor helicopter," *International Journal of Intelligent Control and Systems*, vol. 14, no. 3, pp. 204–212, Sept. 2009.

[73] J.-Y. Wen and K. Kreutz-Delgado, "The attitude control problem," *IEEE Transactions on Automatic Control*, vol. 36, no. 10, pp. 1148–1162, Oct. 1991.

[74] G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin, "Quadrotor helicopter trajectory tracking control," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference (GNC)*, Aug. 2008.

[75] G. Gremillion and J. S. Humbert, "System identification of a quadrotor micro air vehicle," in *Proceedings of the AIAA Conference on Atmospheric Flight Mechanics*, 2010.

[76] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar, "The GRASP multiple micro-UAV testbed," *IEEE Robotics and Automation Magazine*, vol. 17, no. 3, pp. 56–65, Sept. 2010.

[77] D. Cabecinhas, R. Naldi, L. Marconi, C. Silvestre, and R. Cunha, "Robust take-off and landing for a quadrotor vehicle," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1630–1635, May 2010.

[78] Vicon, "Vicon MX Systems," [http://www.vicon.com/products/viconmx.html], 2011.

[79] M. Valenti, B. Bethke, G. Fiore, and J. P. How, "Indoor multi-vehicle flight testbed for fault detection, isolation, and recovery," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference (GNC)*, Aug. 2006.

[80] MicroStrain, Inc., "3DM-GX1 IMU," [http://www.microstrain.com], 2011.

[81] F. Kendoul, Y. Zhenyu, and K. Nonami, "Embedded autopilot for accurate waypoint navigation and trajectory tracking: Application to miniature rotorcraft UAVs," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2884–2890, May 2009.

[82] A. Davison, I. Reid, N. Molton, and O. Stasse, "MonoSLAM: Real-time single camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, June 2007.

[83] J. Y. Bouguet, "Pyramidal implementation of the Lucas Kanade feature tracker," 1999, Tech. Rep., Intel Corp.

[84] B. K. Horn and B. G. Rhunck, "Determining optical flow," in *Artificial Intelligence 17*, pp. 185–204, 1981.

[85] F. Kendoul, I. Fantoni, and K. Nonami, "Optic flow-based vision system for autonomous 3D localization and control of small aerial vehicles," *Robotics and Autonomous Systems*, vol. 57, no. 6-7, pp. 591–602, Feb. 2009.

[86] J. Zufferey and D. Floreano, "Fly-inspired visual steering of an ultralight indoor aircraft," *IEEE Transactions on Robotics*, vol. 22, no. 1, pp. 137–146, Feb. 2006.

[87] T. Kanade, O. Amidi, and Q. Ke, "Real-time and 3D vision for autonomous small and micro air vehicles," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*, vol. 2, pp. 1655–1662, Dec. 2004.

[88] P. Beasley, P. Gray, K. Usher, and N. Bergmann, "Design, construction and modelling of a low cost miniature UAV using machine vision," in *Proceedings of the ARAA Australasian Conference on Robotics and Automation (ACRA)*, pp. 1–6, Dec. 2007.

[89] B. Steder, G. Grisetti, C. Stachniss, and W. Burgard, "Visual SLAM for flying vehicles," *IEEE Transactions on Robotics*, vol. 24, no. 5, pp. 1088–1093, Oct. 2008.

[90] G. P. Tournier, M. Valenti, and J. P. How, "Estimation and control of a quadrotor vehicle using monocular vision and moiré patterns," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference (GNC)*, pp. 2247–2252, Aug. 2006.

[91] S. Fowers, D. Lee, B. Tippetts, K. Lillywhite, A. Dennis, and J. Archibald, "Vision aided stabilization and the development of a quad-rotor micro UAV," in *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA)*, pp. 143–148, June 2007.

[92] O. Bourquardez, R. Mahony, N. Guenard, F. Chaumette, T. Hamel, and L. Eck, "Image-based visual servo control of the translation kinematics of a quadrotor aerial vehicle," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 743–749, June 2009.

[93] F. Kendoul, K. Nonami, I. Fantoni, and R. Lozano, "An adaptive vision-based autopilot for mini flying machines guidance, navigation and control," *Autonomous Robots*, vol. 27, pp. 165–188, 2009.

[94] F. Kendoul, Z. Yu, and K. Nonami, "Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles," *Journal of Field Robotics*, vol. 27, no. 3, pp. 311–334, 2010.

[95] H. Romero, S. Salazar, and R. Lozano, "Real-time stabilization of an eight-rotor UAV using optical flow," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 809–817, Aug. 2009.

[96] J. Conroy, G. Gremillion, B. Ranganathan, and J. Humbert, "Implementation of wide-field integration of optic flow forautonomous quadrotor navigation," *Autonomous Robots*, vol. 27, pp. 189–198, 2009.

[97] S. Ahrens, D. Levine, G. Andrews, and J. How, "Vision-based guidance and control of a hovering vehicle in unknown, GPS-denied environments," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2643–2648, May 2009.

[98] S. Zingg, D. Scaramuzza, S. Weiss, and R. Siegwart, "MAV navigation through indoor corridors using optical flow," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3361–3368, May 2010.

[99] P. Rudol, M. Wzorek, and P. Doherty, "Vision-based pose estimation for autonomous indoor navigation of micro-scale unmanned aircraft systems," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1913–1920, May 2010.

[100] R. He, S. Prentice, and N. Roy, "Planning in information space for a quadrotor helicopter in a GPS-denied environment," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1814–1820, May 2008.

[101] S. Soundararaj, A. Sujeeth, and A. Saxena, "Autonomous indoor helicopter flight using a single onboard camera," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5307–5314, Oct. 2009.

[102] Intel Corporation, "OpenCV Computer Vision Library," [http://www.intel.com], 2011.

[103] M. Achtelik, "Vision-based pose estimation for autonomous micro aerial vehicles in GPS-denied areas," Master's thesis, Technische Universität, München, Germany, 2009.

[104] M. Achtelik, A. Bachrach, R. He, S. Prentice, and N. Roy, "Stereo vision and laser odometry for autonomous helicopters in GPS-denied indoor environments," in *Proceedings of the International Society for Optical Engineering (SPIE)*, Apr. 2009.

[105] K. Celik, S.-J. Chung, M. Clausman, and A. Somani, "Monocular vision SLAM for indoor aerial vehicles," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1566–1573, Oct. 2009.

[106] Logitech, "QuickCam Pro 5000," [http://www.logitech.com], 2011.

[107] G. Klein and D. Murray, "Parallel tracking and mapping for small AR workspaces," in *Proceedings of the IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 1–10, 2007.

[108] G. Klein and D. Murray, "Improving the agility of keyframe-based SLAM," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 802–815, Oct. 2008.

[109] B. Williams, G. Klein, and I. Reid, "Real-time SLAM relocalisation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 1–8, Oct. 2007.

[110] L. R. Garcia-Carrillo, E. Rondon, A. Dzul, A. Sanche, and R. Lozano, "Hovering quad-rotor control: A comparison of nonlinear controllers using visual feedback," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*, pp. 1662–1667, Dec. 2010.

[111] F. J. Richards, "A flexible growth function for empirical use," *Journal of Experimental Botany*, vol. 10, no. 2, pp. 290–301, 1959.

[112] J. Moore and R. Tedrake, "Powerline perching with a fixed-wing UAV," in *Proceedings of the AIAA Infotech@Aerospace Conference*, Apr. 2009.

[113] H. Huang, G. Hoffmann, S. Waslander, and C. Tomlin, "Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3277–3282, May 2009.

[114] P. Abbeel, A. Coates, and A. Y. Ng, "Autonomous helicopter aerobatics through apprenticeship learning," *International Journal of Robotics Research*, 2010.

[115] J. Gillula, H. Huang, M. Vitus, and C. Tomlin, "Design of guaranteed safe maneuvers using reachable sets: autonomous quadrotor aerobatics in theory and practice," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1649–1654, May 2010.

[116] S. Lupashin, A. Schöandllig, M. Sherback, and R. D'Andrea, "A simple learning strategy for high-speed quadrocopter multi-flips," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1642–1648, May 2010.

[117] A. F. Sorenson, "Autonomous control of a miniature quadrotor following fast trajectories," Master's thesis, Aalborg University, Aalborg, Denmark, 2010.

[118] A. Lussier Desbiens, A. T. Asbeck, and M. R. Cutkosky, "Landing, perching and taking off from vertical surfaces," *International Journal of Robotics Research*, vol. 30, no. 3, pp. 355–370, 2011.

[119] R. Cory and R. Tedrake, "Experiments in fixed-wing UAV perching," in *Proceedings of the AIAA Guidance, Navigation, and Control Conference (GNC)*, 2008.

[120] R. E. Cory, *Supermaneuverable Perching*. Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, MA, 2010.

[121] S. Bayrakar and E. Feron, "Experiments with small helicopter automated landings at unusual attitudes," in *arXiv, Georgia Institute of Technology*, Feb. 2008.

[122] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," in *Proceedings of the IFRR International Symposium on Experimental Robotics (ISER)*, 2010.

[123] Nintendo of America, Inc., "Wii Remote," [http://www.nintendo.com], 2011.

[124] Kako, "Interfacing Wii Camera over I^2C," [http://kako.com/neta/2008-009/2008-009.html], 2011.

[125] Philips Semiconductors, "The I^2C-Bus Specification," [http://www.nxp.com/acrobat_download2/literature/9398/39340011.pdf], 2000.

Appendix

Bill of Materials

Description	Brand	Part #	Qty	PPU	Notes
Motors and Propellors				$80.08	
Motor - Brushless DC, 11V	HobbyKing	KDA-20-22L	4	$12.27	
Propeller - Slow Flyer, Push 10x4.7"	APC	LP10047SFP	2	$4.39	
Propeller - Slow Flyer, Pull 10x4.7"	APC	LP10047SF	2	$2.93	
Prop Adapter w/ Collet, 1/8"	E-flite	EFLM1923	4	$3.95	
AL screws - 3mm x 1cm			8	$0.07	
Frame				$161.18	
AL Square Rods - 10mmx10mm, 10cm long - 3 Black, 1 Red	Mikrokopter	MK-40 Frameset	4	$13.50	
Fiberglass Square Plate	Mikrokopter	MK-40 Frameset	4	$8.20	
AL 50mils Structural Plate, 3.5"x5"			1	$0.50	
AL Battery Cage (25mils)			1	$0.50	
Landing Strut	Align	H60126T	2	$7.00	
Landing Skid	Align	H60137-84	2	$6.00	
Rubber Damper Standoffs (3mm)	Mikrokopter	MK-40 Frameset	4	$8.50	
AL screws - 3mm x 1.6cm w/ self locking nuts	Mikrokopter	MK-40 Frameset	8	$0.25	
Screws, AL, 4-40, 1/2", 3/4", 1" with nuts			8	$0.08	
Screws, AL, 6-32, 3/4"			2	$0.08	
Lego pieces	Lego		40	$0.10	
Plastic Nut - 3mm	Mikrokopter	MK-40 Frameset	7	$0.10	
Plastic Standoff (3mm x 1.5cm)	Mikrokopter	MK-40 Frameset	14	$0.42	
Power Board				$34.48	
Perforated Board - 10cm x 10cm Octagon			1	$2.50	
Perforated board - 1.25in x 1.75in			1	$0.25	
Heat sink, large			1	$1.25	
Heat sink, medium			3	$0.70	
Push Wire Connector, 3 and 4 pin			9	$0.10	
Plastic screws - 3mm x 8 mm	Mikrokopter	MK-40 Frameset	4	$0.20	
Gumstix screws, nuts	Gumstix	KIT0013	5	$4.00	
Power plug, elbow, 2.3mm			2	$1.25	
20ga wire, black, red			20	$0.05	per in
Wire wrap, various colors			24	$0.05	per in
Velcro			10	$0.10	per in
Electrical Tape			36	$0.01	per in
Header connectors and pins (2,3,4 sizes)			9	$0.05	
Dean's connectors (male and female)			1	$0.75	
Zipties, small			17	$0.01	
Processors				$299.16	
Gumstix, Verdex Pro XL6P COM	Gumstix	GUM270B-XL6P	1	$169.00	
Robostix - Servo controller	Gumstix	PKG00019_TH	1	$39.00	
Robostix - ESC output, sonar input	Gumstix	PKG00019_TH	1	$39.00	

ESC 30A	Turnigy Plush		4	$11.54	
14ga wire, black, red			120	$0.05	per in
Sensors				$188.95	
Sonar Sensor	MaxBotix	LV-EZ2	1	$27.95	
Webcam	Logitech Quickcam Pro 5000	961444-1403	1	$25.00	
Wide Angle Lens (2.1 mm M12)	Unibrain	2047	1	$25.00	
IR Blob Camera	Nintendo Wii-iRemote	Pixart	1	$30.00	
Plain PCB - 2in x 3in			1	$1.00	
25MHz crystal	Digikey	x-231-ND	1	$0.58	
IC Hex inverter	Digikey	296-4290-5-ND	1	$0.52	
Gyro sensor	Sparkfun	SEN-09801	1	$49.95	$10/chip
Accelerometer sensor	Sparkfun	SEN-09723	1	$29.95	$8/chip
Electronics				$232.61	
Mini Servo	Futaba	S3106	1	$14.00	
Regulator, 5V	Digikey	LM1084	2	$3.61	
Capacitor, 50uF, 10uF, 100uF			8	$0.20	
Wireless Receiver	Xbee	XBP24-ACI-001	1	$32.00	
Xbee to breakout board w/ headers	Sparkfun	BOB-08276	1	$2.95	
Battery - 11.1 V, 2600 mAh	Thunderpower	TP2600-3SPL2	1	$55.00	
Switch	Radioshack	275-014	1	$3.99	
Regulator, 3.3V	Sparkfun	LD1117V33	2	$1.95	
Wi-Fi board	Gumstix	PKG10080	1	$60.00	
Wi-Fi module	Gumstix	KIT150-US	1	$40.00	
Wi-Fi antenna for Gumstix	Gumstix	ANT006	1	$10.00	
I^2C Logic Converter	Sparkfun	BOB-08745	1	$1.95	
Total Cost				$996.46	