

UCLA

UCLA Electronic Theses and Dissertations

Title

Rotor Dynamics and Control Applications in a 6 Degree of Freedom Twist-Tilt Quadcopter

Permalink

<https://escholarship.org/uc/item/3gz0q463>

Author

Alawadhi, Abdulaziz M A A

Publication Date

2022

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
Los Angeles

Rotor Dynamics and Control Applications in a 6 Degree of Freedom Twist-Tilt Quadcopter

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Mechanical Engineering

by

Abdulaziz M A A Alawadhi

2022

© Copyright by
Abdulaziz M A A Alawadhi
2022

ABSTRACT OF THE DISSERTATION

Rotor Dynamics and Control Applications in a 6 Degree of Freedom Twist-Tilt Quadcopter

by

Abdulaziz M A A Alawadhi

Doctor of Philosophy in Mechanical Engineering

University of California, Los Angeles, 2022

Professor Tsu-Chin Tsao, Chair

One of the biggest shortcomings of traditional quadcopters is that they are underactuated with control over only 4 out of a possible 6 degrees of freedom. The user may control translations in all three axes as well as a single rotation in the Z axis, otherwise known as yaw. A handful of designs have been introduced in the field to gain control of the two remaining degrees of freedom, as gaining control of them would mean complete control over the quadcopter's attitude and position, a useful feature in certain applications. This work delves deeper into a design introduced in a previous work, of an overactuated quadcopter design with twelve system inputs, where each arm has 3 control inputs – propeller speed, a twist angle, and a tilt angle. A system identification of the quadcopter is completed using an experimental setup that dissects the quadcopter into a modular part – a single-arm. Furthermore, a multibody dynamic simulation of the complete quadcopter is created which accounts for the system's mass distribution, moments of inertia, and low level motor dynamics which were simplified or ignored in previous work. In addition to the previously proposed control scheme, a new scheme is also introduced which simplifies the system into a 6 input to 6 output system, thus creating a more simple and elegant control scheme. Both control schemes are tested in simulation and experiment with success.

The dissertation of Abdulaziz M A A Alawadhi is approved.

Jacob Rosen

Robert M'Closkey

Veronica Santos

Tsu-Chin Tsao, Committee Chair

University of California, Los Angeles

2022

To my family, for their never ending love and support.

To my mother, my biggest fan and greatest supporter.

To my sister, my safety net and trusted confidant.

And to my wife, my best friend and sanctuary.

TABLE OF CONTENTS

List of Figures	vii
List of Tables	ix
Acknowledgments	x
Vita	xi
1 Introduction	1
2 Mechatronic Design	7
2.1 Hardware	8
2.2 Electronics & Sensors	13
2.3 Communication	17
3 Kinematics	20
3.1 Thrust and Drag Vectoring	22
4 System Modeling & Identification	25
4.1 System Modeling	25
4.2 System Identification	29
4.2.1 High Level System Identification	30
4.2.2 Low Level System Identification	33
4.3 Validating System Identification with Single-Arm Setup	35
4.3.1 Simulation Setup	35
4.3.2 Experimental Setup	38

4.3.3	Results	39
5	System Control Scheme	47
5.1	12-Input Control Scheme	47
5.2	6-Input Control Scheme	54
5.2.1	6-Input Control Scheme Singularity Cases	56
6	Four-Arm Simulation & Experiments	59
6.1	Simulation Setup	59
6.2	Simulation Results	61
6.3	Experimental Setup	64
6.4	Experimental Results	66
6.4.1	Hover	67
6.4.2	Yaw	69
6.4.3	Roll & Pitch	71
6.4.4	Roll, Pitch, and Yaw Simultaneously	73
6.4.5	X Y Disturbance Force	75
6.5	Simulation vs Experimental Results	77
6.6	Energy Consumption	79
6.7	Experimental Repeatability Issues	81
7	Conclusion	86

LIST OF FIGURES

1.1	Drone with mounted ultrasonic thickness sensor by Tritex NDT.	2
2.1	CAD rendering of complete quadcopter assembly.	7
2.2	Part of the fabrication process of the <i>alpha</i> arc.	8
2.3	Old version of quadcopter body.	9
2.4	New version of quadcopter.	10
2.5	Exploded view of the single-arm setup.	12
2.6	Arm PCB board with motor drivers.	14
2.7	Birdseye view of body with two Crazyflies labeled.	15
2.8	Crazyflie with Lighthouse deck mounted (left) and Lighthouse beacon (right).	17
2.9	Communications flow and protocols.	18
3.1	Simplified quadcopter with 12-input actuations and inertial frames labeled.	21
3.2	<i>alpha</i> , <i>beta</i> , and <i>omega</i> actuations shown on a single-arm setup.	22
3.3	Quadcopter arm highlighting the achievable space for propeller direction.	23
4.1	Motor circuit diagram.	27
4.2	Experimental setup used to find the thrust coefficient using a fulcrum and scale.	31
4.3	Experimental setup used to find the drag coefficient.	32
4.4	Circuit used to measure the motor's inductance, L , using a square wave input.	34
4.5	Simulation model of the single-arm setup.	36
4.6	Propeller response to reference signal with ramp up and ramp down.	37
4.7	Physical model of the single-arm setup.	38
4.8	Position of α and β in experiment & simulation while propeller is off.	40

4.9	Position of α and β in experiment and simulation with propeller at 75% speed.	42
4.10	Single-arm with angular momentum, motor torque, and precession torque on α	42
4.11	Single-arm with angular momentum, motor torque, and precession torque on β	43
5.1	Control scheme for the 12-input quadcopter system.	48
5.2	Position and Attitude Control Law.	49
5.3	Visual of wrench mapper converting u_F and u_τ into 12 q vectors.	53
5.4	Visual of modified wrench mapper converting u_F and u_τ into ρ , ϕ , and θ	54
5.5	Control scheme for the 6-input system quadcopter.	56
5.6	Quadcopter configuration when $\phi = \frac{\pi}{2}$ and $\theta = 0$	57
6.1	Four-arm setup in simulation with MyRIO and platform shown.	60
6.2	Noise in CrazyFlie and Lighthouse sensors.	61
6.3	Position and attitude tracking results in simulation for both controllers.	62
6.4	Response of the 12 actuators in simulation for both controllers under pure yaw.	64
6.5	Picture of four-arm physical setup with safety enclosure.	65
6.6	Picture of copter while hovering.	67
6.7	Tracking results of both controllers while hovering.	68
6.8	Tracking results of both controllers while yawing.	70
6.9	Tracking results of both controllers while rolling.	71
6.10	Tracking results of both controllers while pitching.	72
6.11	Tracking results of both controllers with roll, pitch, and yaw.	74
6.12	Illustration of pulley experiment to apply force in X and Y directions.	76
6.13	Tracking results of both controllers with X Y force disturbance.	77
6.14	Tracking results of old and new simulation with hardware for 6-input controller.	78
6.15	Illustration of how the body bends causing the sensors to rotate.	83

LIST OF TABLES

1.1	Experimental results of different designs with variable numbers of system inputs.	3
2.1	List of components used with their masses and moments of inertia.	13
5.1	List of gains for position and attitude control laws.	51
5.2	List of gains for alpha and beta motors.	54
6.1	Charge consumption of 12-Input System vs 6-Input System.	80

ACKNOWLEDGMENTS

First and foremost, I would like to acknowledge my advisor, Professor Tsu-Chin Tsao for all that he has done for me. He is a truly remarkable man who always brings out the best in his students and is never at a loss for new unique ideas. It has been an honor to work with him throughout my studies at UCLA, where I could not have asked for a better advisor. From the MAE staff, I would like to thank Ben and Miguel for their guidance and dedication to the students in the department. They have always been dependable whenever I needed any kind of hardware work done. I would also like to thank Marla for all the administrative support she has given me over the years.

I would also like to acknowledge my friend and lab mate Omar. Without Omar, none of the work in this dissertation would have been possible, and I would pick him over an army of engineers for any project. I would also like to thank my mentors Martin Lee and Matt Gerber for always taking the time to teach me the skills necessary to complete this project. Also, I would like to thank Omar (again), Kevin, Han, and Edwin for always being great friends and companions in the lab.

And, of course, I would like to thank my parents for all their sacrifices in ensuring I received the best education. My sister, Eman, for always being there for me when I needed her. And to the rest of my family, for always caring and supporting me through my journey. And last but certainly not least, my wife Nouf. I do not know what I did to deserve someone so much better than me, but thank you for always standing by my side even in the most difficult times.

VITA

- 2013 B.S. (Mechanical Engineering), Boston University.
- 2013–2015 Field Engineer, Schlumberger.
- 2015–2017 Mechanical Integrity Engineer, EQUATE Petrochemical Company.
- 2017–2019 Master’s Student, Mechanical Engineering Department, UCLA.
- 2019–present Ph.D. Student, Mechanical Engineering Department, UCLA.

CHAPTER 1

Introduction

Interest in quadcopters, and multicopter drones in general, has grown rapidly over the last few years due to their simplicity and many potential applications. In this work, the terms quadcopter, drone, and copter will be used interchangeably. Traditional quadcopters are controlled by varying the propeller speeds of 4 propellers, which allow its center of mass to be translated in all three axes independently of each other. The drone is also able to rotate about its vertical axis (the Z axis) – an actuation commonly referred to as yaw. However, the system is underactuated as the user does not have control over the two rotations about the X and Y axes – actuations commonly referred to as roll and pitch respectively. The roll and pitch angles are uncontrollable because they are coupled with the X and Y axes translations. This means that, using high speed maneuvers, it would be possible to obtain a desired pitch or roll for a short period of time [KR13]. However, if the desired trajectory requires that the quadcopter hold a certain pitch or roll angle for an extended period of time while staying stationary with respect to X and Y, this would be impossible. Therefore, although the traditional quadcopter is an elegant and simple solution that works in many applications, it remains underactuated with only 4 system inputs and 4 degrees of freedom.

Applications that require control over roll and pitch include tasks that involve object manipulation, interaction with the environment, inspection, and navigation [RLO18]. Recently, many large factories are exploring such applications, with the goal of minimizing some of their costs. For instance, refineries have been attempting to use drones to reduce inspection costs by mounting ultrasonic thickness probes onto drones to inspect vessels and piping that are difficult to reach without scaffolding. Scaffolding has been reported to take up 20 to 40% of some maintenance budgets, resulting in millions of dollars of annual cost. If a drone can

be utilized to eliminate the need for scaffolding, this could result in large savings for these companies. Designs to undertake such a task were introduced by numerous companies, and one such design by Tritex NDT [NDT19] is shown in Figure 1.1. However, such a design would only be able to measure test pieces that are oriented perpendicular to the ground, as the orientation of the probe must match the orientation of the test piece. When dealing with piping and large vessels, many surfaces will not be oriented in this way, and this motivates the need for a design that has control over roll and pitch, as well as the payload carrying capacity to carry inspection tools.



Figure 1.1: Drone with mounted ultrasonic thickness sensor by Tritex NDT.

In pursuit of a truly decoupled system with 6 degrees of freedom, many designs have been introduced in the literature with the goal of controlling roll and pitch. Some designs introduced systems with 6 system inputs while others pushed for more overactuation with up to twelve system inputs. Many designs incorporated an actuated twist joint into the design [HHM15, NK14, OAN15, IL17, BTK20]. A few designs were selected from the literature as a representation and are shown in Table 1.1. The first example shown is a hexacopter with fixed tilt angles by Jiang [JV14] with a total of 6 system inputs - 6 propeller speeds. The idea of propellers with fixed tilt angles was explored in a number of designs with varying tracking results [CLM11, RRB15, NGK15, BD16, RMP17, PLA18, RCS19]. Another example of a 6

system input design is a tri-rotor design with tilt angles [ESG08, KKR15], which controls 3 propeller speeds and their 3 tilt angles. In their respective papers, both designs were shown to have at least some control over roll and pitch. But the experimental results also showed that the two designs' control over roll and pitch is quite limited, as one design only performs the experiment in a very controlled environment, peg-in-hole test, and the other achieves a twenty degree pitch angle but does so with very large errors in position. So, although control of roll and pitch is possible theoretically, they are not very controllable in a practical sense.

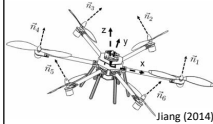
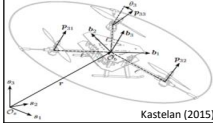
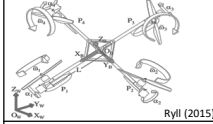
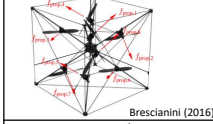

Drone Design	System Inputs	Commanded Roll	Commanded Pitch	Commanded Yaw	Commanded X Position	Commanded Y Position	Commanded Z Position
 Jiang (2014)	6	N/A: Peg-In-Hole Testing	20° Error: N/A	N/A: Peg-In-Hole Testing	N/A: Peg-In-Hole Testing	N/A: Peg-In-Hole Testing	N/A: Peg-In-Hole Testing
 Kastelan (2015)	6	20° Error: ~2°	2° Error: ~3°	8° Error: ~3°	0 m Error: ~0.8m	0 m Error: ~0.8m	1 m Error: ~0.3m
 Ryll (2015)	8	0° Error: ~10°	20° Error: ~6°	0° Error: ~13°	0 m Error: ~0.06m	0 m Error: ~0.03m	0 m Error: ~0.01m
 Brescianini (2016)	8	0° Error: ~7°	360° Error: ~3°	0° Error: ~7°	0 m Error: ~0.1m	0 m Error: ~0.1m	0 m Error: ~0.1m
 Elkhatib (2017)	12	0° Error: ~16°	180° Error: ~15°	0° Error: ~18°	0 m Error: ~0.2m	0 m Error: ~0.5m	0 m Error: ~0.3m

Table 1.1: Experimental results of different designs with variable numbers of system inputs.

Other designs have tried to introduce more system inputs to achieve better system maneuverability. For instance, an actuated tilt angle quadcopter by Ryll [RBG14] as seen in Table 1.1, which allows the angle of each propeller to be controlled in one axis, thus adding 4 system inputs for a total of 8 system inputs - 4 propeller speeds and 4 tilt angles. Experimentally, this design commanded a 20 degree pitch angle with less error in position when compared with the previous designs mentioned. Another design introduces the same actu-

ated tilt angle in one axis to a hexacopter by Elkhatib [Elk17], thus adding 6 system inputs for a total of 12 system inputs - 6 propeller speeds and 6 tilt angles. Experimentally, this design outperforms the other three designs mentioned previously. It achieves a 180 degree pitch angle, thus turning the drone upside down, with comparatively reasonable errors in the other 5 degrees of freedom. Finally, a design is introduced by Brescianini [BD16] with 8 systems inputs, where all of the inputs are propeller speeds. The design is similar to Jiang's design, except there are 2 more blades and they are oriented in more directions in 3 dimensional space. This system does very well in terms of tracking in 6 degrees of freedom. In fact, it performs the best out of all of the designs listed in Table 1.1. The issue with this design, however, is its payload carrying capacity. Because the propellers are oriented in numerous directions, the full thrust of the system can never be fully utilized. If the 8 motors in the design were oriented to all face upwards instead, the system would have about 5 times as much thrust as it does in its current orientation. In addition to this, the system's cube design make it difficult to mount tools onto the frame, making this design insufficient for the desired application.

It can be deduced from Table 1.1 that there is a correlation between the number of system inputs and the maneuverability of the system. This is ultimately due to thrust vectoring, where systems with more inputs tend to have more control over the direction in which each propeller may point in 3 dimensional space. Having the ability to point each propeller in more directions enables the system to generate thrust and torque in more directions thus making the system more controllable. It is important, however, to point out that more system inputs alone are not sufficient to increase the system's thrust vectoring authority when these inputs are redundant. Although some designs in Table 1.1 have greater thrust vectoring authority than others, none of the designs achieve full thrust vectoring authority for all propellers. This is due to joint angle limitations governed by design choices. Furthermore, an inability to achieve full thrust vectoring authority means that many orientations that are achievable with these designs will result in a large amount of wasted energy due to the need to subdue internal forces from thrust generated by one propeller with thrust from another propeller in

the X and Y directions. This creates the need for a new design which can do two things. First, the ability to track in 6 degrees of freedom, as some designs have shown. Second, maximize carrying capacity so that various tools can be carried for different applications.

A new design was introduced for a quadcopter with actuated tilt and twist angles in each arm by Gerber [GT18]. This design would result in a system with a total of 12 system inputs. Gerber argues that this system achieves omnidirectional control of both position and attitude - much like the other systems discussed previously. However, what separates this design from the others that were discussed is that it is also the only one that has full thrust vectoring authority over the propellers' directions in 3 dimensional space. This effectively means that it would be the most energy efficient design with the least wasted energy due to internal force cancellation in the X and Y directions needed to stabilize the system. In his paper, Gerber also goes on to discuss the reference frames, the control law, and the inverse kinematics of this system. This is followed by some simple simulation results which show that the system is indeed decoupled in each of its 6 degrees of freedom when the proposed control scheme is applied to the system. It should be noted that although the control scheme was tailored to this quadcopter specifically, other works have adopted similar hierarchical control schemes [RRB15, OAN15, ML12, KKR15, KVE18, CLM11, JV14, Elk17, PRY21].

Although Gerber's simulation model does help verify that the control scheme is effective, it simplifies the system quite extensively. For example, the system is simplified into a point mass with set moments of inertia in each axis. It also ignores the angular momentum of the rotating propellers. The biggest simplification made, however, is that the system's 12 actuators are able respond to the commanded reference signals seamlessly and immediately without any dynamic considerations. This reduces the fidelity of the simulation model and reduces confidence in its results when it is used as a test bench for the controller prior to migrating it onto the physical hardware. One of the goals of this work is to create a more complex and accurate simulation model for the system. This is done to get a better representation of the hardware and give greater confidence in the model's results. To do this, the system is first identified in more depth to include the low level dynamic response of the

12 actuators moving the system. The hardware is then tested extensively with experiments to determine the values of the system parameters that make up the new model.

The simulation model is also used to test a new control scheme that was developed with the goal of simplifying the system. This control scheme operates differently than the one introduced by Gerber. The control scheme proposed by Gerber controls 12 system inputs independently to produce 6 degrees of freedom in the overall system. It is thus redundant, as the desired system output can have multiple solutions, where a single solution is selected by minimizing propeller speeds. The new control scheme constrains 8 system inputs with only 2 commands, thus creating an overall system with only 6 system inputs and 6 degrees of freedom. This simplifies the system and gives a more elegant solution to the problem at hand, with only one possible solution to each desired system output.

Using the simulation model developed, both control schemes are thoroughly tested in simulation before migrating the controllers directly to the hardware for physical flight testing. The two control schemes are compared in simulation and in experiment using the same reference input and similar control inputs. The results are analyzed and discussed.

CHAPTER 2

Mechatronic Design

The most recent version of the quadcopter which will be the main discussion of this work is shown in Figure 2.1 along with some of its components. In this section, the hardware used to build the system will be clearly outlined. The electronics and sensors that are mounted to the system to monitor various parameters will also be covered, along with the communications used to control the quadcopter. Finally, the system's resulting kinematics and relevant inertial frames are discussed.



Figure 2.1: CAD rendering of complete quadcopter assembly.

2.1 Hardware

With a desired payload capacity of around 5 kg, keeping the structure weight down was vital to the copter's ability to fly and perform the desired maneuvers. Therefore, all structural elements were machined out of carbon fiber plate. Two plates of carbon fiber sandwich acetal plastic parts which act as standoff plates and are machined to be the joints that hold the alpha arm. Carbon fiber and acetal are selected because they are both light and have high structural rigidity. Many of these elements were initially fabricated by Gerber. A brief summary of the construction will be covered here.

Aside from the propellers, all structural components, other than the tubing and alpha arc, are machined from 2mm carbon fiber plate and 3/8" acetal plastic plate. The only exception was the alpha arc, which was the most difficult component to make. This was made by Gerber by using a custom-made mold and then applying layer by layer of carbon fiber with an epoxy to bind the layers together. An image of the mold with the arc in the drying process is shown in Figure 2.2.

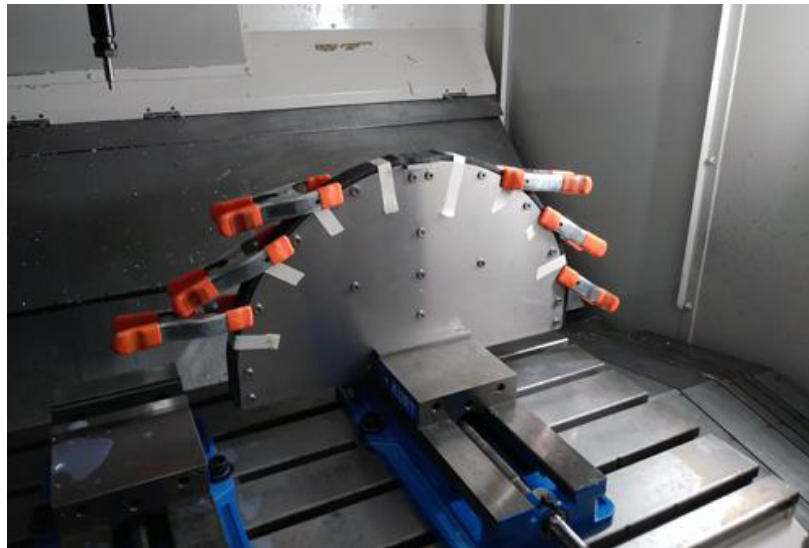


Figure 2.2: Part of the fabrication process of the *alpha* arc.

Gerber also created the beta arm and all the needed joints and couplings to mount these

onto the body. For the most part, the elements from the alpha arm down to the propeller have been kept the same in the new design. The only change made is the bearing within the joints. In the alpha joint specifically, the amount of friction in the joint was quite high and this was affecting the position tracking performance in the alpha joint. These joints' passive sleeve bearings were thus replaced with needle bearings which made the joints a lot smoother. The beta bearings are unchanged. The biggest change made to the design comes with respect to the body itself. The original body designed and fabricated by Gerber is shown in Figure 2.3.

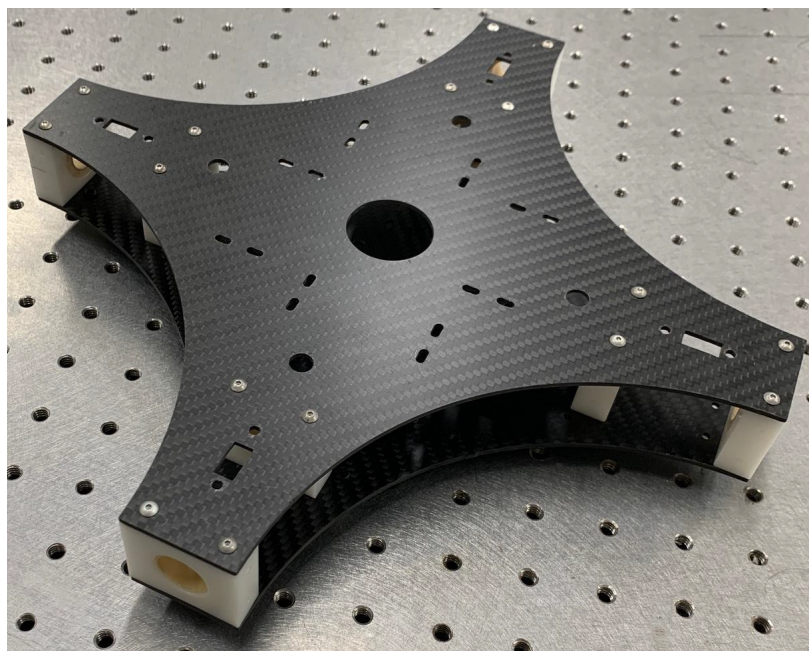


Figure 2.3: Old version of quadcopter body.

The old version of the body is compact in size, and a decision was made to increase the size substantially for two main reasons. First, the goal of this project was to maximize payload carry capacity to mount various tools onto the copter. It would thus follow that space would be needed to mount said tools, preferably in the middle of the structure. Second, one of the discoveries from a similar project showed that bringing the thrust sources too close together could lead to unexpected behavior due to turbulent effects and so the rods were

sized accordingly [PRY 9]. By making the body larger, the propellers are spread out more, allowing them to pull from a larger pool of air and reducing the turbulent effects from one propeller to the next.

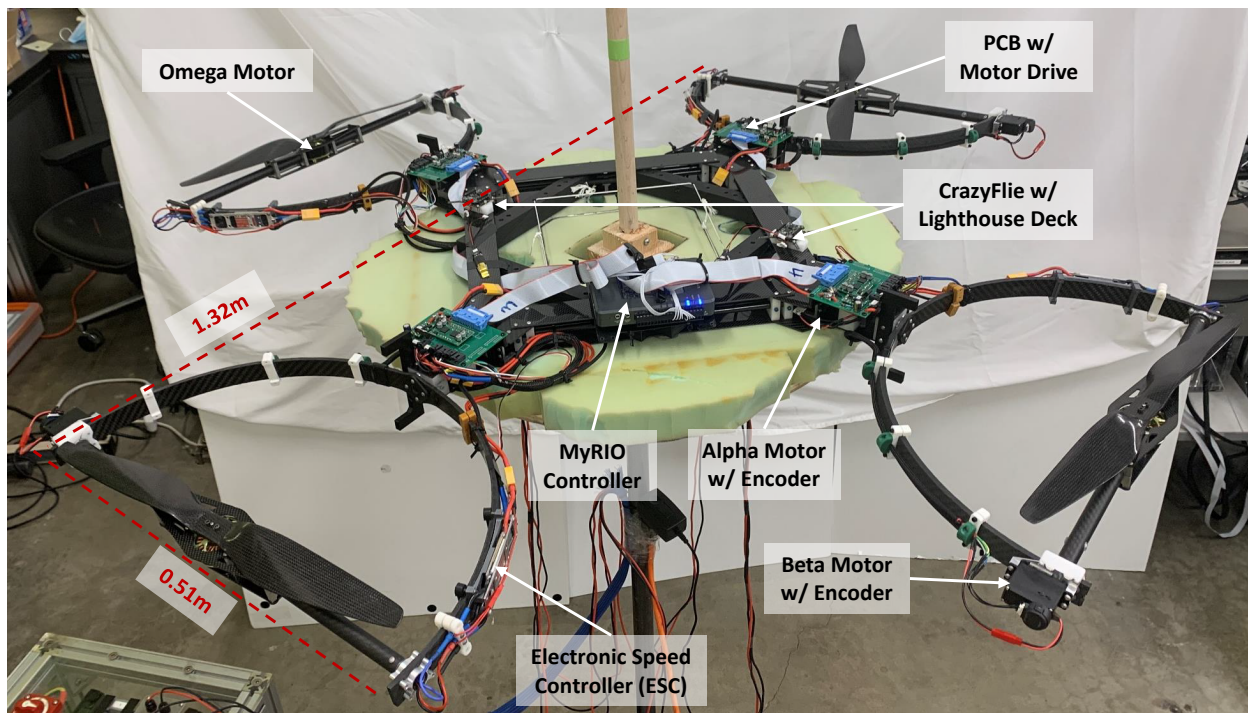


Figure 2.4: New version of quadcopter.

A larger body is designed with independent arm housings interconnected by two 10mm carbon fiber tubes. The distance from one propeller to the opposite propeller is now 1.32 meters compared to only 0.82 meters previously. Initially, the tubes were meant to be the only connection from arm to arm. However, after extensive testing, it was found that the structure was not stiff enough, and the arms were rotating with respect to one another about the tubing joints. Although the rotation angles were small, the relative displacement from one arm to the next at the end of the tubes was sizable. So, carbon fiber stiffening plates were added above and below each set of tubes to increase the stiffness of the connection. This worked well to stiffen the structure, and this is the final version of the copter. The final product can be seen in Figure 2.4. This design leaves ample room in the middle of the

copter to mount any kind of tool that the user deems necessary. Also, mounting bars, which are made of 3D printed PLA, are added to the system. These do not add any structural support to the system and are only added to give the user space to mount tools.

The motors used in the design are all labeled in Figure 2.4 to show exactly where they are housed. The alpha motor sits inside the newly designed arm housing, the beta motor sits on the alpha arc, and the omega motor sits inside the beta arm. There is a dual motor drive soldered onto each arm's printed circuit board (PCB) to drive the alpha and beta motors. To drive the omega motor, an electronic speed controller (ESC) is mounted onto each alpha arm's arc. This approach relies on direct control of the revolute joints unlike other indirect methods that rely on propeller rotors [PRY 9, NPP18].

When fully assembled with the body parts, motors, batteries, and the microcontroller, the copter weighs around 5.5kg. The propellers are 18 inches in diameter and, when coupled with the brushless DC motor, can spin at a speed of around 550 rad/s. At this speed, each propeller can produce a little over 25 Newtons of thrust, with the complete system producing over 100 Newtons in thrust, or about 10.5kg in total carry capacity. The microcontroller utilized for the control of the system is National Instrument's MyRIO, a real time target that performs all computations on a synchronized timed loop, preventing any asynchronous time varying sampling found in traditional nested control systems [SYG10]. The microcontroller is mounted onto the body's frame with ribbon cables running to each arm's PCB to run the signals from the controller to each board and back.

In its simplest form, the design of the quadcopter incorporates 4 unique parts: the body, alpha arm, beta arm, and propeller. These are outlined in Figure 2.5 with an exploded view of the modular arm. Note that the alpha arm has its motor mounted within the copter arm body, the beta arm has its motor mounted onto one side of the alpha arm, and the propeller has its motor mounted on the beta arm. This means that these bodies are connected to one another in a serial fashion and that each propeller is allowed to tilt and twist through these serial connections. It is worth pointing out that, since the beta motor sits on one side of the alpha arc, it inflicts a constant torque bias due to its mass on the alpha joint that the alpha

motor must continuously compensate for. Since there are 3 motors per arm and 4 arms, the final system has 12 inputs in total.

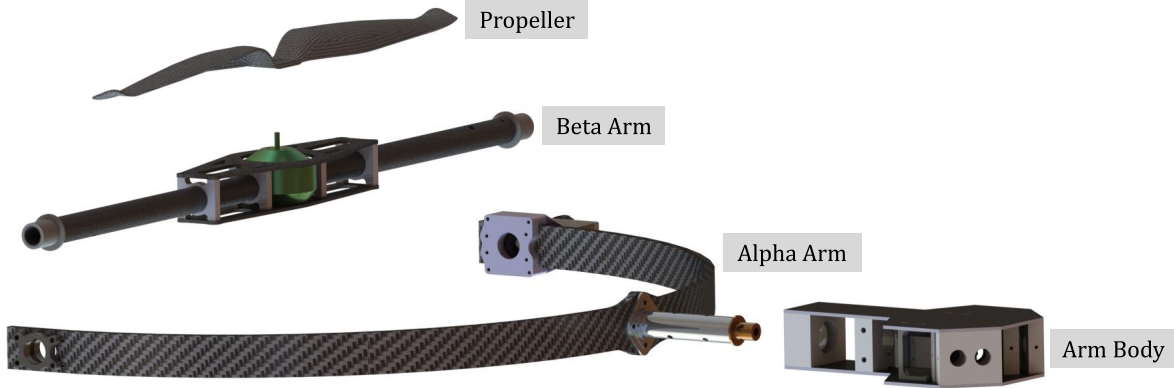


Figure 2.5: Exploded view of the single-arm setup.

To connect the body to the alpha arm, a revolute joint is used. Previously, a sleeve bearing was used in this joint to minimize friction. However, the sleeve bearing was found to have too little an effect on friction and so it was replaced with a needle bearing. The needle bearing reduces the amount of friction in the joint more effectively as the alpha angle is being actuated. Similarly, the alpha arm is connected to the beta arm using another revolute joint. This joint also has a sleeve bearing but was found to be sufficient in dealing with the friction in this joint and so it was not changed. Finally, the propeller is mounted directly onto a brushless DC motor which is mounted onto the beta arm, this is yet another revolute joint. Therefore, in total, there are 3 motors and 3 revolute joints actuating each arm. The alpha and beta arms are actuated using DC motors of similar size, while the propeller is rotated by a brushless DC motor.

A list of the components used along with their mass and moment of inertia are listed in Table 2.1 for reference. Some of the smaller components are ignored here, as their effect is small on the system's dynamics. Note that all 3 moments of inertia are listed for the body, but for the alpha arm, beta arm, and propeller, only 1 moment of inertia about the axis of the revolute joint is listed. The alpha joint's moment of inertia is very large compared to the

other 2 joints because of the beta motor that sits on one side of the alpha arc. The MyRIO's moment of inertia is not listed because it is lumped into the body's moment of inertia.

	Mass (kg)	I (kg-mm ²)
Body	3.459	133430, 133430, 266050
Alpha Arm	0.251	9368.3
Beta Arm	0.227	42.4
Propeller	0.023	244.9
MyRIO	0.225	N/A

Table 2.1: List of components used with their masses and moments of inertia.

2.2 Electronics & Sensors

To sufficiently sense and control the drone, several electronic components and sensors are needed. Firstly, a microcontroller is required to implement the control scheme. For this, National Instrument's MyRIO is used and mounted directly onto the quadcopter. It was chosen due to its high processing power, as well as its ability to implement FPGA code. To write the code onto the MyRIO and view the output results in real-time, a computer with LabVIEW installed is also required.

The aforementioned PCB, shown in Figure 2.6, is another vital electronic part made up of numerous components. It serves two purposes. The first is routing the digital signals back and forth from the microcontroller via a ribbon cable to the appropriate actuator or sensor. The second is receiving the input from the power source and stepping down the voltage to get the voltage necessary for each component in the system. The PCB has 2 voltage regulators two step down the power source which comes in at a voltage of 24 volts. The power source's voltage is only suitable for the omega motor which runs at 24 volts and has a Kv rating of

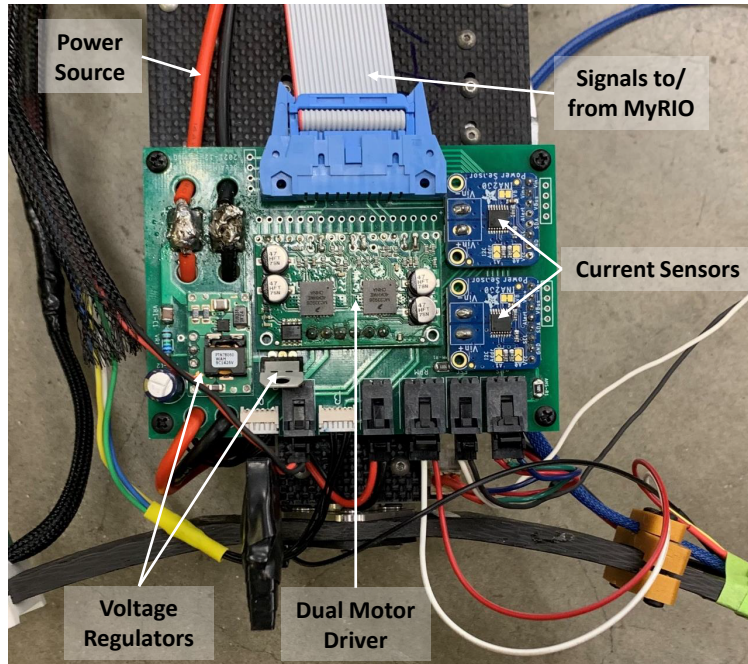


Figure 2.6: Arm PCB board with motor drivers.

320 RPM per volt. Therefore, the first voltage regulator on the PCB steps down the voltage from 24 volts to 7.4 volts. The 7.4 voltage is used to supply power to the alpha and beta motors but is still too high for the rest of the components in the system, namely the sensors. Thus, the second voltage regulator further steps down the voltage from 7.4 volts to 3.3 volts, which is the voltage that the encoders, RPM sensors, current sensors, and homing sensors require.

The encoders are necessary to close the position control loop on the alpha and beta motors, and so there is an encoder mounted on each of these motors. The RPM sensor is used to measure the actual speed of the propellers. The propeller's speed control loop is already closed by a controller that is built into the ESC and cannot be altered. However, the actual RPM is still useful to have and so it is measured for modeling and troubleshooting purposes. The homing sensor is used on the alpha joint to ensure that the 0 position established is the same each time the system is run. The two current sensors that are integrated into the circuit, one for the alpha motor and one for the beta motor, are used to measure the amount of current running through each alpha and beta motor. The current reading from

these motors is useful when trying to control the motor as it can be used to measure torque directly, as current is proportional to the torque produced, whereas voltage is not. Although it was part of the initial plan, the current reading is currently unused. But this feature is still available for future control designs.

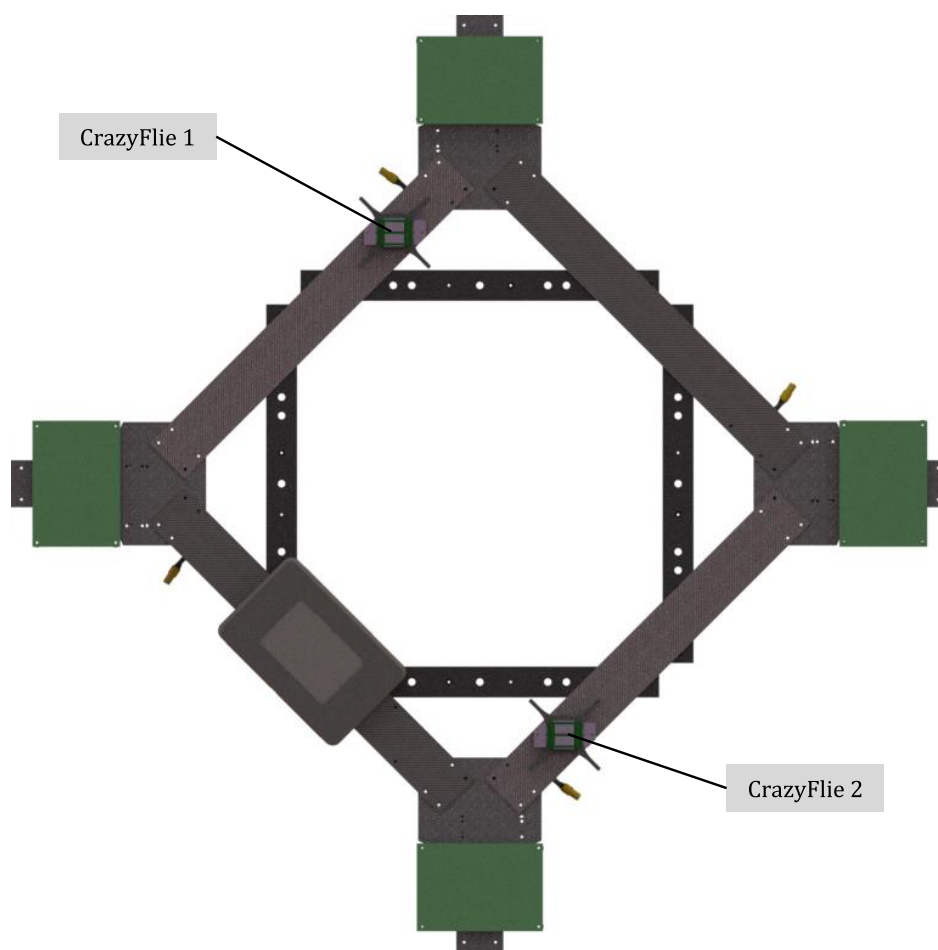


Figure 2.7: Birdseye view of body with two Crazyflies labeled.

At a high level, feedback of the quadcopter body's position and orientation are required for control purposes. In most of the other projects mentioned, motion capture is achieved using infrared camera technology systems such as Optitrack or Vicon. However, these systems can be quite expensive and so a cheaper alternative was considered. In this project, the copter's position and orientation are captured using Bitcraze's Crazyflie 2.1 [Bit14] which is mounted

onto the body of the quadcopter coupled with Bitcraze's Lighthouse technology [Bit19] as shown in Figure 2.7. It should be noted that this is a standalone mini drone but has had its motors removed so that its built-in sensors can be used. The Crazyflie has an IMU and gyroscope onboard, which account for orientation feedback. Also, two Crazyflies are used instead of one because the body of the quadcopter was found to be somewhat flexible and bends slightly about the X and Y axes. This means that the attitude at the point at which the Crazyflie is mounted will not necessarily match the attitude at the quadcopter's center of mass when bent - which could be very disruptive to the controller. So, by using two of them and averaging the roll, pitch, and yaw values across both sensors, a more accurate measurement of attitude can be obtained at the center of mass of the body. To obtain position data, Bitcraze's Lighthouse technology is utilized [Bit19]. This consists of a laser emitting beacon and a receiving deck with markers that is mounted onto the Crazyflie's body, both shown in Figure 2.8. For better visibility of the platform and to ensure minimal physical blockage from the beacon to the sensors, two beacons are utilized. Using this setup takes care of all of the sensing required for the quadcopter body's position and attitude at a high level. However, it should be noted that the Lighthouse signal can only transmit the required data at a maximum frequency of 100Hz as it does so through a radio signal to the ground computer. The high level controller also runs at this frequency, and so any drops in signal will hinder the high level controller's performance. It should be noted that the infrared camera technologies that were previously mentioned run at 250Hz, and their signals are transmitted via Ethernet cable to the ground computer. They are thus more reliable, but the Lighthouse system is still utilized regardless.

On a lower level, there are additional parameters that require measurement. Because their positioning is of utmost importance, encoders were retrofitted onto all alpha and beta motors. This allows each of these motors to have its own feedback control loop. These motors also each have a motor drive. In the case of the brushless DC motor that runs the propeller, an electronic speed controller (ESC) is needed to drive the motor. This is a 40-amp ESC with internal speed control with gains predetermined by the manufacturer. This means that

the speed control loop cannot be modified to enhance the speed response or reshape it in any way. To monitor the speed of the propeller while the quadcopter is running, an RPM sensor is also incorporated even though this is not used in the control loop. This sensor is used for modeling purposes and to monitor the performance in case troubleshooting is needed. Finally, a homing sensor is used to ensure that the 0 set point position is achieved every run in the alpha arm. The beta arm is more balanced and will hold its position, and so this can be done manually for beta.

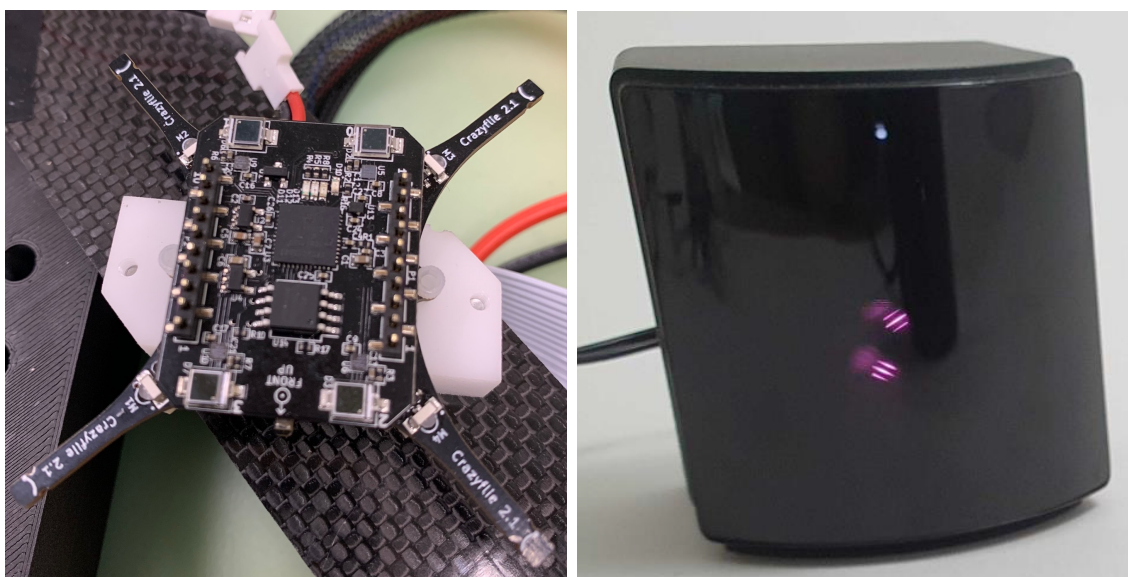


Figure 2.8: Crazyflie with Lighthouse deck mounted (left) and Lighthouse beacon (right).

2.3 Communication

The communication methods with which the hardware components communicate must be established, as not all commands and feedback are generated and performed on the same piece of equipment. Figure 2.9 shows an outline of the communication flow and protocols used to control the quadcopter with reference to the hardware mentioned in the previous section. There are 3 important devices within the communication flow: a ground computer,

the MyRIO microcontroller, and the CrazyFlie sensors coupled with the Lighthouse system [Bit19].

A principal idea here is that all parts of the control scheme run on the MyRIO and the MyRIO only. No part of the control scheme runs on any other hardware, and so all feedback signals must return to the MyRIO, and all command signals must originate at the MyRIO. The MyRIO is physically mounted onto the copter’s frame, and so it can be wired directly to the actuators and their sensors. It can thus send commands and collect feedback data directly through this wired connection for all 12 motors.

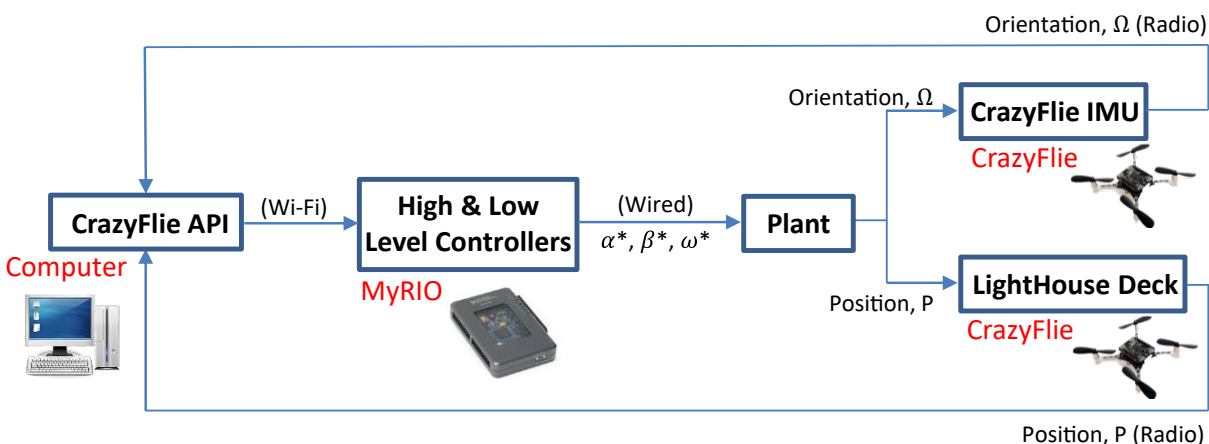


Figure 2.9: Communications flow and protocols.

However, as previously mentioned, the quadcopter body position and attitude data are collected by the Crazyflies coupled with the Lighthouse system which uses the IMU, gyroscope, and Lighthouse laser-emitting beacons. Once the data is collected by the CrazyFlie sensors, it is sent over a radio signal to a computer running BitCraze’s native API specific to these mini drones. It is worth mentioning that the radio signal used has an upper frequency limit of 100Hz, as this is the minimum frequency needed to run the position and attitude controller, any unexpected drops in signal can thus be destabilizing to the system. The API also filters the collected data, which is convenient, as the data can be directly relayed to the controller in this form. The next step is to extract the data from the API and stream it into

LabVIEW; note that both programs are running on the ground computer. Once the data is streamed into LabVIEW, it must be sent to the MyRIO microcontroller via a Wi-Fi signal. The data is sent through a Wi-Fi dongle on the computer and is received by the MyRIO, which is capable of receiving the Wi-Fi signal. The controller now has all the data it needs to execute one step of its control loop and the cycle can continue.

CHAPTER 3

Kinematics

In addition to covering the hardware, it is also important to look at how each part of the quadcopter moves and how it is constrained in 3 dimensional space, as well as what frames are necessary to represent the system accurately. In this section, a detailed description of the system's actuatable joints and nomenclature will be covered. To avoid confusion, the motor actuations that move the system will be referred to as system inputs and the body's overall ability to translate and rotate in 3 dimensions will be referred to as degrees of freedom. The 12 system input quadcopter is shown in Figure 3.1 with its various system inputs, shown with blue arrows, and all of the frames applied onto the system, shown with green coordinate systems.

For the purposes of this work, the tilt angles in the arms will be referred to as the alpha angles, the twist angles will be referred to as the beta angles, and the spinning speed of the propeller will be referred to as omega. This nomenclature will be used throughout this dissertation. Note that only omega is a speed measurement, while alpha and beta are position measurements as these are the parameters that are most relevant to the system. The parameters are labeled accordingly in Figure 3.1 and Figure 3.2. It is also important to note that there are a total of 6 inertial frames involved in the control scheme that actuates the quadcopter; a global inertial frame fixed in space and 5 additional inertial frames fixed to the body. These frames can be seen in Figure 3.1.

Each of these inertial frames is used for different computations required to control the

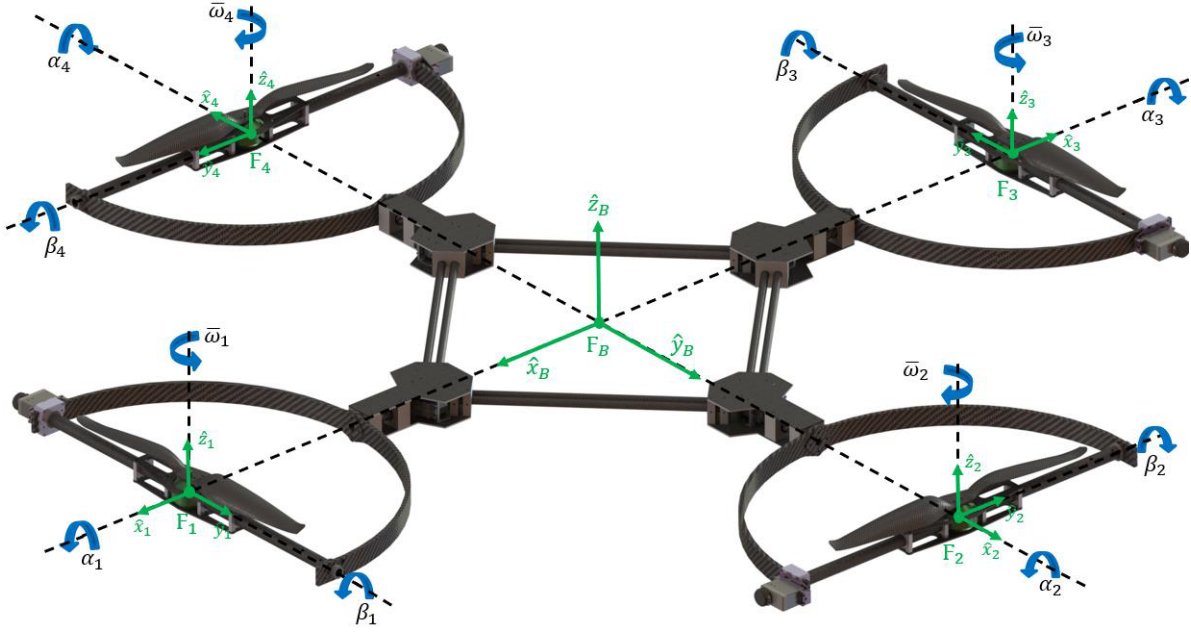


Figure 3.1: Simplified quadcopter with 12-input actuations and inertial frames labeled.

system. The first frame is the global world frame that is fixed in 3 dimensional space with respect to the quadcopter. All 3 translational degrees of freedom of the quadcopter are measured against this frame. There is also the quadcopter body's inertial frame which is fixed onto the quadcopter's center of mass. All three rotational degrees of freedom of the quadcopter are measured against this frame as measuring these against the world frame means that, for example, a pure yaw may induce motion about all three axes in the body frame depending on the current orientation of the quadcopter - this is confusing and difficult to follow. And so, the frame chosen to measure the rotations with respect to is the body frame, as it is more intuitive [BM99]. Finally, there is an inertial frame fixed on each of the arms on the intersection of the alpha and beta angles. All frames are oriented so that the positive X direction always points away from the system's center of mass and the positive Z direction is pointing up. These frames are used to command the individual actuators on each arm. This is done so that a rotation of some value in the alpha joint remains the same regardless of what the rest of the quadcopter is doing. Similarly, commanding a beta angle or omega speed is done with respect to each actuator's own axis of motion.

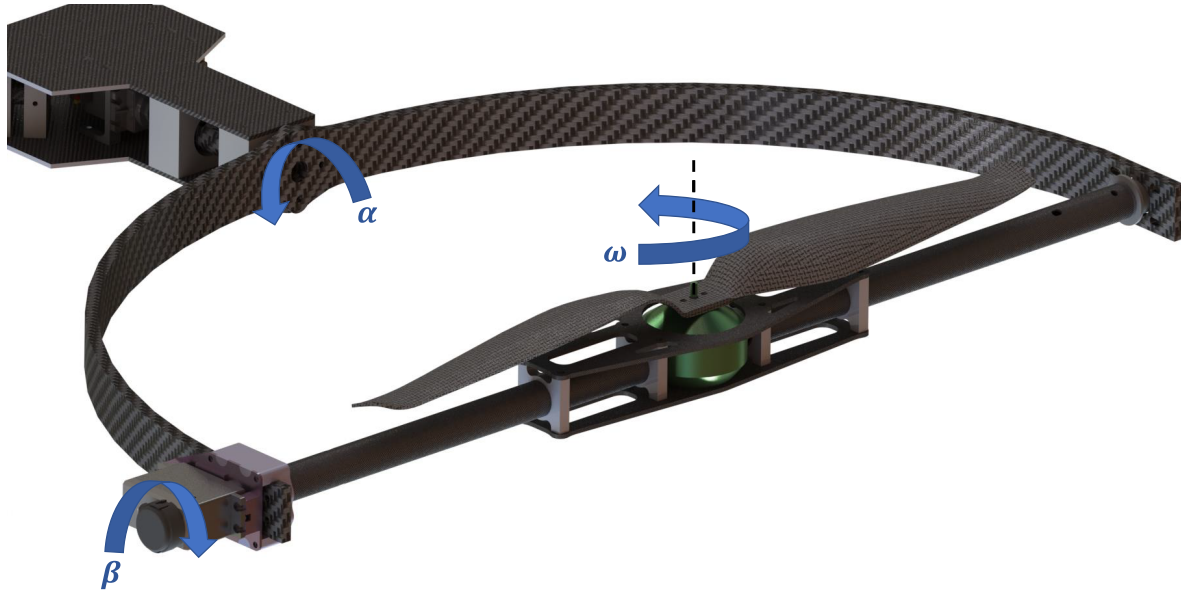


Figure 3.2: *alpha*, *beta*, and *omega* actuations shown on a single-arm setup.

3.1 Thrust and Drag Vectoring

To highlight the significance of how the kinematics of this system make it unique, it is important to point out the system’s vectoring authority. The propellers are the only components within the quadcopter that create sizable forces and torques capable of moving the body. They create a thrust force aligned with their spin direction due to the reactionary force from the air being pushed down. But the propellers also experience torque in the direction opposite to their spin due to the air resisting the blade’s spin. Although there are other aerodynamic forces at play when the quadcopter is running, propeller thrust and drag are the primary means of moving the system’s center of mass. For the purposes of this work, other effects such as blowback will be ignored since the quadcopter is not meant to fly close to structures or in narrow spaces [GG17]. Downwash is also ignored as the integral action in the controller is to compensate for its effects [YSG21].

The significance of having tilt and twist angles that intersect at a point in space on each propeller means that this design kinematically gives full thrust and drag vectoring

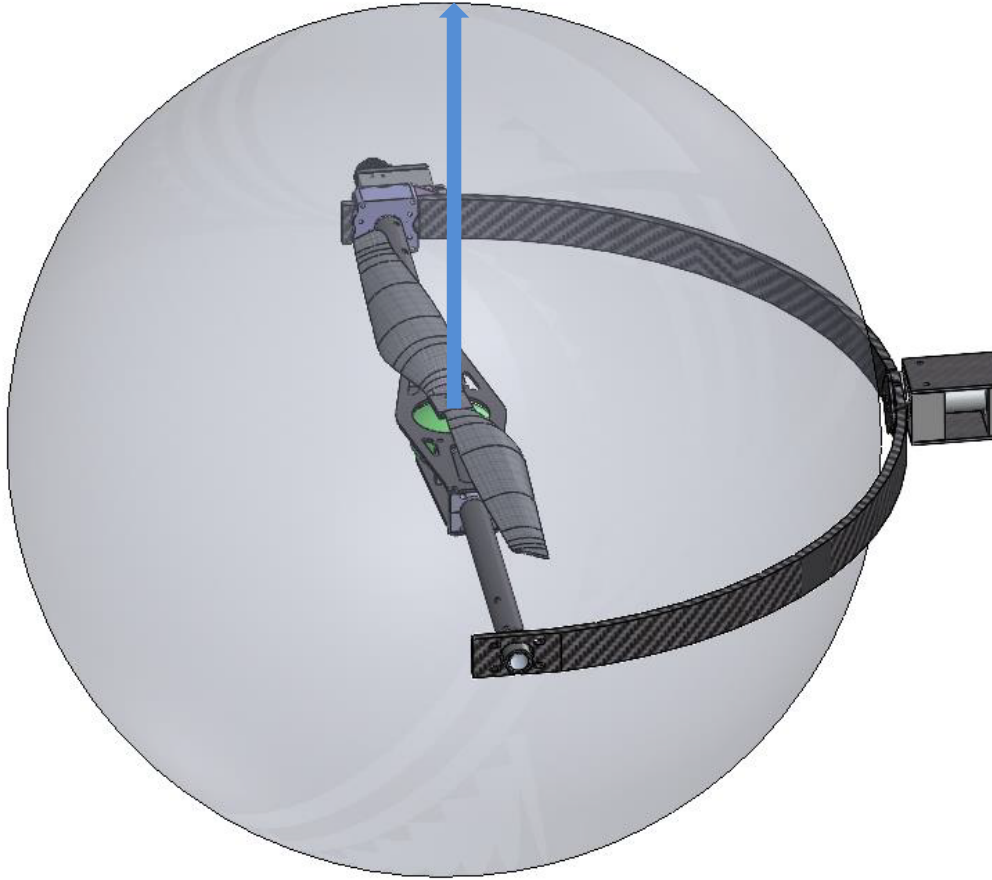


Figure 3.3: Quadcopter arm highlighting the achievable space for propeller direction.

authority over each propeller. In other words, each propeller, and consequently its thrust and drag vectors, can be pointed in any direction in 3 dimensional space as shown in Figure 3.3 which highlights the achievable space as a full sphere. This is a feature that no other design in the field has introduced yet. Having full vectoring authority over each propeller allows the design to access forces and torques in any axis necessary which results in a more maneuverable drone. This is evident in the literature, as outlined in the previous section, as designs with more vectoring authority have more control over roll and pitch actuations. The feature allows the design to eliminate the need to cancel X and Y forces internally to hold certain positions, especially while hovering. For instance, with any other design, if the drone is to hover while holding any attitude other than a flat configuration, it would need to cancel force components in the X and Y directions created by one propeller with

another propeller. This means that energy must constantly be used to compensate for joint limitations imposed on the system by design choice. In the case of the tilt-twist quadcopter that is being presented here, such joint limitations do not exist. The quadcopter can be commanded to hold any attitude while holding its position and it will always be able to point its propellers straight up to counter gravity without any need to compensate for force components in X and Y directions.

CHAPTER 4

System Modeling & Identification

4.1 System Modeling

In this section, the overall multibody dynamics of the quadcopter body and the low level actuators will be derived. The dynamic derivation of how each propeller's drag torque and thrust actuate the overall quadcopter body were derived by Gerber [GT18]. The key elements of the derivation are highlighted here since they are relevant, but a more thorough derivation can be found in Gerber's publication. First, the propellers' thrust and drag are considered, as they are the main sources of force and torque in the system:

$$t_i = C_p \bar{\omega}_i^2 \hat{t}_i \quad (4.1)$$

$$\bar{\tau}_i = (-1)^i C_\tau \bar{\omega}_i^2 \hat{t}_i \quad (4.2)$$

Where C_p and C_τ are the coefficients of thrust and drag respectively. Note that both are proportional to the propeller speed squared. Also, the thrust vectors of all propellers are always pointing in the same direction, while the drag vector alternates in direction depending on which blade is being considered. This is because the blades alternate in spin direction, so does the drag torque as it will always act in the direction opposite the blade's spin. Next, the vehicle dynamics of the quadcopter are derived using Newton-Euler to balance forces and torques:

$$m\ddot{\mathbf{p}} = m\mathbf{g}_0 + \mathbf{R} \sum_{i=1}^4 \bar{\mathbf{R}}_i \mathbf{t}_i + \mathbf{F}_d \quad (4.3)$$

$$\mathbb{I}\dot{\boldsymbol{\Omega}} + \mathbf{h} = \boldsymbol{\tau}_g + \sum_{i=1}^4 [\mathbf{d}_i \times (\bar{\mathbf{R}}_i \mathbf{t}_i) + \bar{\mathbf{R}}_i \bar{\boldsymbol{\tau}}_i] + \boldsymbol{\tau}_d \quad (4.4)$$

In these equations, m is the mass of the quadcopter, \mathbf{p} is the position matrix, \mathbf{g}_0 is the gravity vector, \mathbf{R} is the body attitude, \mathbf{F}_d is an external disturbance force, \mathbb{I} is a lumped inertia matrix of all the components of the quadcopter, \mathbf{h} is the change in the body's angular momentum due to rotation, $\boldsymbol{\tau}_g$ is the moment inflicted on the body due to gravity, \mathbf{d}_i is the constant distance from the propeller's axis of spin to the body's center of mass, and $\bar{\mathbf{R}}_i$ is each arm's rotation matrix.

By substituting Equation 4.1 into Equation 4.3 and Equation 4.2 into Equation 4.4, the following equations are obtained:

$$m\ddot{\mathbf{p}} = m\mathbf{g}_0 + \mathbf{R}\mathbf{u}_F \quad (4.5)$$

$$\mathbb{I}\dot{\boldsymbol{\Omega}} + \mathbf{h} = \boldsymbol{\tau}_g + \mathbf{u}_\tau \quad (4.6)$$

where,

$$\mathbf{u}_F = C_p \sum_{i=1}^4 \mathbf{q}_i \quad (4.7)$$

$$\mathbf{u}_\tau = C_p \sum_{i=1}^4 \mathbf{d}_i \times \mathbf{q}_i + C_\tau \sum_{i=1}^4 (-1)^i \mathbf{q}_i \quad (4.8)$$

$$\mathbf{q}_i = \bar{\omega}_i^2 \bar{\mathbf{R}}_i \hat{\mathbf{t}}_i \quad (4.9)$$

The rearranged form in Equations 4.5 and 4.6 is more useful from a controls perspective since \mathbf{u}_F and \mathbf{u}_τ are control outputs which will later be used in the control scheme. It's

important to realize that the multi-body dynamics of the system have been simplified quite a bit here. For instance, it is assumed that the system's moment of inertia matrix, \mathbb{I} , has a constant value and does not change with changes in the system's orientation. Physically, this is not true, however the changes in \mathbb{I} are quite small when the alpha and beta angles are actuated since the distance from the quadcopter's center of mass to each arm's point of rotation is fixed, and so the change is negligible. Other effects that are ignored are mainly aerodynamic effects such as the downwash effects in certain configurations. Naturally, the real dynamics of the system are far more complex than this simple derivation. However, a simplified system model, such as the one that appears in Equations 4.5 and 4.6, is needed for control design purposes. Once the controller is designed based on the simple model, a more complex simulation model can be built, and the controller can be applied to the simulation to verify that the control design is robust enough to stabilize the system. Finally, with the confidence gained by this process, a hardware test may be conducted more safely.

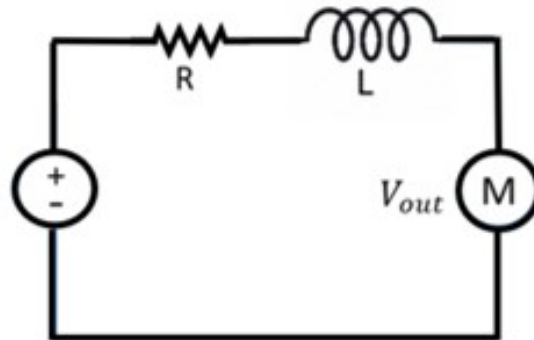


Figure 4.1: Motor circuit diagram.

Although this derivation captures the quadcopter's vehicle dynamics at a high level, it does not consider the low level dynamics of the 12 individual actuators which actually move the system. These low level dynamics may be neglected for the model used to design the controller. However, they are important and needed, as the speed at which the magnitude and direction of u_F and u_τ can be changed is dependent on the dynamics involved in these

low level actuators. Thus, failing to accurately model these dynamics in the simulation model used to verify the control design could lead to inaccurate system responses when compared to the physical hardware. This could lead to a controller that succeeds in controlling the system in simulation but fails to control the physical system.

To model the actuators, a simple circuit diagram is considered in Figure 4.1. From this diagram, the following equation is derived for the current, i , in the circuit:

$$i = \frac{V - K_m \omega_{in}}{R} \quad (4.10)$$

Note that the motor inductance, L , does not appear in this equation as it is ignored. This is because its value was found to be small experimentally and therefore its effect on this equation is negligible. The experiment used to find the value of the motor inductance will be discussed later in the System Identification section. Next, the actuator's inner and outer shafts need to be considered separately since the gear ratio is another parameter that needs to be captured. The inner and outer shafts are represented by Equations 4.11 and 4.12 respectively.

$$\tau_{in} = K_i i - J_{in} \dot{\omega}_{in} \quad (4.11)$$

$$J_{out} \dot{\omega}_{out} = \tau_{out} - b_{out} \omega_{out} \quad (4.12)$$

Note that the friction which acts on the outer shaft, τ_{out} , is also considered here. Also, the gear ratio that ties the inner shaft to the outer shaft is also included:

$$\omega_{in} = N \omega_{out} \quad (4.13)$$

$$\tau_{in} = N \tau_{out} \quad (4.14)$$

Using Equations 4.10 through 4.14, a series of equation substitutions can be performed to arrive at a transfer function from input voltage, V , to output shaft speed, ω_{out} . However, in the case of the alpha and beta motors, the main interest lies in tracking the position and not the speed of the motor. So, the transfer function is simply multiplied by an integrator to obtain a transfer function from motor input voltage to the motor output shaft position, P_{out} . These are shown in Equations 4.15 and 4.16.

$$\frac{\hat{\omega}_{out}}{\hat{V}} = \frac{\frac{Nk_i}{R}}{(N^2 J_{in} + J_{out})s + (\frac{N^2 k_i^2}{R} + b_{out})} \quad (4.15)$$

$$\frac{\hat{P}_{out}}{\hat{V}} = \frac{\frac{Nk_i}{R}}{(N^2 J_{in} + J_{out})s^2 + (\frac{N^2 k_i^2}{R} + b_{out})s} \quad (4.16)$$

The final form of the transfer function shown in Equation 4.16 now gives the position of the alpha and beta actuators given an input voltage, and so the low level dynamics of each motor is captured. However, there is still a need to find the values of the parameters used in the transfer function and this must be done experimentally through proper system identification techniques.

4.2 System Identification

In the derivations performed in the System Modeling section, many system parameters were introduced. To model the physical system accurately, a thorough system identification must be conducted so that all these parameters can be found experimentally. Although finding some of these parameters might seem trivial, the methods used will be outlined here for completeness since these parameters' values are vital in ensuring that the results obtained by the physical system will match the results of the simulation model that will be developed to test the control design.

4.2.1 High Level System Identification

The high level vehicle dynamics derivation in Equations 4.5 and 4.6 introduces the following physical parameters: \mathbb{I} , m , d_i , C_p , C_τ . Some of these are easy to obtain via a CAD model or by performing a direct physical measurement such as the moment of inertia matrix (\mathbb{I}), quadcopter mass (m), and the distance from the alpha rotation axis to the propeller (d_i). However, the coefficients of thrust (C_p) and drag (C_τ), which account for aerodynamic effects in the model, must be obtained with experimental setups. Both parameters are based on the blade's geometry, and so different blades will have different coefficients of thrust and drag.

The thrust coefficient, C_p , is determined using a simple fulcrum setup where the force generated by the propeller is measured by a scale on the other side of the fulcrum at different RPM values, an RPM is utilized here. A picture of the simple setup can be found in Figure 4.2. By plotting the propeller speed squared, w^2 , versus the force generated in Newtons, the slope of the line can be used to find the thrust coefficient of the propeller. Note from Equation 4.1 that, for the purposes of control design, the relationship between the propeller speed squared, w^2 , and the thrust force, t_i , is assumed to be linear. This assumption holds true in experiment, as the relationship between these two parameters is linear in the propeller's operating range.

The drag coefficient, C_τ , is traditionally found using some variation of a pressure sensor in a wind tunnel with the propeller mounted with the desired angle of attack [MC17]. Alternatively, some have found the drag coefficient using a force sensor with the torque from the propeller pushing against this sensor with some lever arm [CFC14]. Another method that could be used is mounting the motor that is running the propeller directly to a torque sensor. However, in this case, the drag associated with the propeller is found using a less conventional method since access to specialized equipment such as a wind tunnel or torque sensor was limited. The method proposed utilizes equipment that was already available in the setup such as a current sensor, RPM sensor, a joint that is aligned with the drag vector produced, and control over the motor actuating the joint about which the drag vector acts.

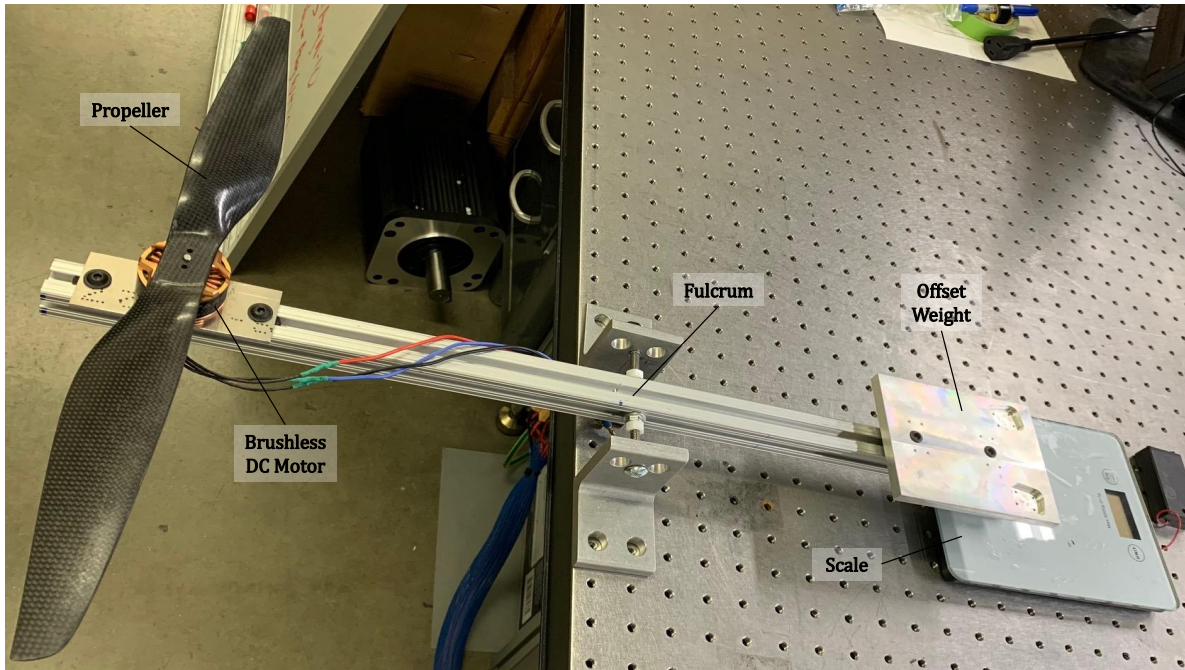


Figure 4.2: Experimental setup used to find the thrust coefficient using a fulcrum and scale.

The beta joint is set to align the propeller's spin axis with the alpha joint axis of rotation as shown in Figure 4.3.

The alpha and beta motors are commanded to hold the joint at this configuration using simple PID controllers. The alpha motor's current is monitored using a current sensor. It should be noted that the static moment about the alpha axis is non-zero due to the imbalance in weight introduced by the beta motor hanging on one side of the arm. This will induce a bias in current in the alpha controller must always compensate for the imbalance. It is important to quantify exactly how much current is running through the alpha motor due to this bias using the current sensor. This can be done by simply commanding the alpha to hold the angle shown without actuating the propeller and recording the amount of current running through the motor at the nominal position so that the current due to the bias can later be accounted for when the propeller is running. It is thus assumed that this bias is constant, even though it would change if the alpha angle were to change substantially. However, if the angle is held relatively close to its starting point throughout the run, then

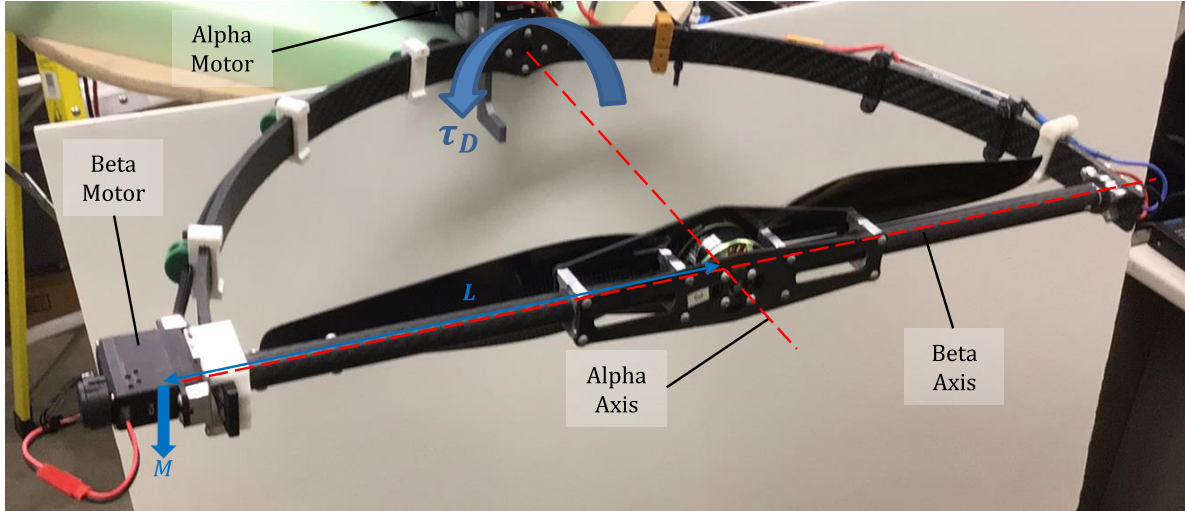


Figure 4.3: Experimental setup used to find the drag coefficient.

the assumption is reasonable.

Now that the bias is accounted for, the propeller is run at increasing propeller speeds while monitoring the amount of current running through the alpha motor at steady state. As the propeller speed increases, the magnitude of current running through the alpha motor should also increase, less the bias current, as it must now compensate for more drag torque introduced by the propeller with each run to keep the alpha joint at the intended position. Assuming that the motor constant is known, the drag coefficient, k_r , can now be found by using a simple moment balance about the alpha joint's axis or rotation:

$$\tau_{mass\beta} + \tau_{motor} = \tau_D \quad (4.17)$$

$$MLg + K_{motor}I = \omega^2 k_r \quad (4.18)$$

$$k_r = \frac{MLg + K_{motor}I}{\omega^2} \quad (4.19)$$

Since measurements of current were taken at many propeller speeds, a more accurate fit can be obtained for the drag coefficient, k_r , using Equation 4.19. One of the most obvious

shortcomings of this method is that it ignores the effects of friction in the alpha joint. Thus, this approach may not be as accurate as the other methods discussed, but given the restriction in available equipment, it gives a reasonable approximation.

4.2.2 Low Level System Identification

The low level dynamic derivation performed for the actuators in Equations 4.10 through 4.16 introduce the following physical parameters: N , k_i , R , J_{in} , J_{out} , b_{out} , and L . Note that each of these parameters must be found twice, once for the alpha motor and once for the beta motor as each motor has its own unique values. Furthermore, it is assumed that all 4 alpha motors are perfectly identical, and all 4 beta motors are perfectly identical. Much like the high level derivation, the gear ratio (N), motor resistance (R), input shaft inertia (J_{in}) and output shaft inertia (J_{out}) of each motor are easy to obtain by direct measurement or a CAD tool, while the rest of the parameters are obtained experimentally.

The motor constant, k_i , is obtained by rotating the motor arm at varying voltages and measuring the steady state rotation speed at each voltage. Using a simple linear fit to the data, the slope of the line will give the motor constant as a value of voltage over rotation speed. Another parameter of interest here is the quantization count in the encoders of the alpha and beta motors as this can be added to the simulation model. This is measured by simply rotating the motor arm a full revolution and measuring the count output from the encoder.

The motor inductance, L , is obtained by feeding a square wave voltage input into the circuit shown in Figure 4.4. A resistor, R_1 , is added to the motor circuit and the square wave voltage, V_{in} , is applied across the circuit. The voltage across the added resistor, R_1 , is measured using an oscilloscope. The voltage across the added resistor can be written in the form shown in Equation 4.20, where R_1 can be chosen to be significantly large compared to R_m to simplify the equation to the form shown in Equation 4.21.

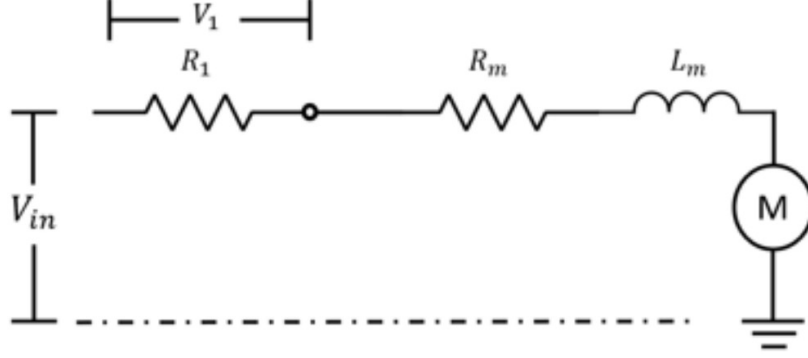


Figure 4.4: Circuit used to measure the motor's inductance, L , using a square wave input.

$$V_1 = \frac{V R_1}{R_1 + R_m} (1 - e^{-\frac{(R_1 + R_m)}{L} t}) \quad (4.20)$$

$$V_1 = V (1 - e^{-\frac{R_1}{L} t}) \quad (4.21)$$

The value of the inductance (L) can now be calculated by obtaining the time constant at 37% of the rated value since R_1 is known:

$$\tau = \frac{L}{R_1} \quad (4.22)$$

$$L = \tau R_1 \quad (4.23)$$

Finally, friction in the joint, b_{out} , must be identified. This is perhaps the most difficult parameter to measure experimentally as it requires very specialized equipment. Instead, this the friction is estimated using a grey-box approach and is assumed to follow a Stribeck friction model. First, various input and output data is collected from the physical model. This data is then fed into a simulation model where all other parameters are known and accounted for using the identification methods that were conducted. The simulation model is then only allowed to fit the given inputs to the correct output response by varying the friction parameters in the Stribeck model. The iterative process is run until the error between the model and the actual data is reduced to a satisfactory level.

4.3 Validating System Identification with Single-Arm Setup

Using the modeling and parameterization techniques discussed in the previous section, experiments are run to obtain the values of the parameters that are relevant to the simulation model, and a multi-body dynamics simulation of the system is created. The focus here is on the single-arm setup, which is a modular part of the quadcopter and can serve as a test of how well the system identification was done. This setup can be experimented upon more easily and eliminates the complexity of dealing with the translations and rotations of the overall body, thus making it easier to test and replicate the effects of each individual actuator. This also means that the high level controller is not in effect, and each of the 3 motor reference signals is created manually.

4.3.1 Simulation Setup

It is important to understand how the multibody dynamic simulation model is set up and what parameters are accounted for before looking at the results. Figure 4.5 shows a visualization of the simulation setup. In this multibody dynamic model, each independent body has a different color to indicate that it is actuated separately. The only exception is the body of the arm, in black, which is grounded so that there are no changes in the body's position or orientation. However, the alpha arm, in red, and the beta arm, in green, are allowed to freely rotate about their respective rotary joints. The propeller, in red, is allowed to rotate in only 1 direction, depending on the arm that is examined, the one shown rotates in the counterclockwise direction.

There are numerous parameters modeled in alpha and beta. To start, each motor's resistance, motor constant, and friction are considered. In addition to this, the gear ratio of each motor is added to the model with the moment of inertia of both the input and output shafts. Furthermore, the inductance that was ignored in the previous model is added. The controller duty cycle to voltage output to the motor is mapped from the physical hardware

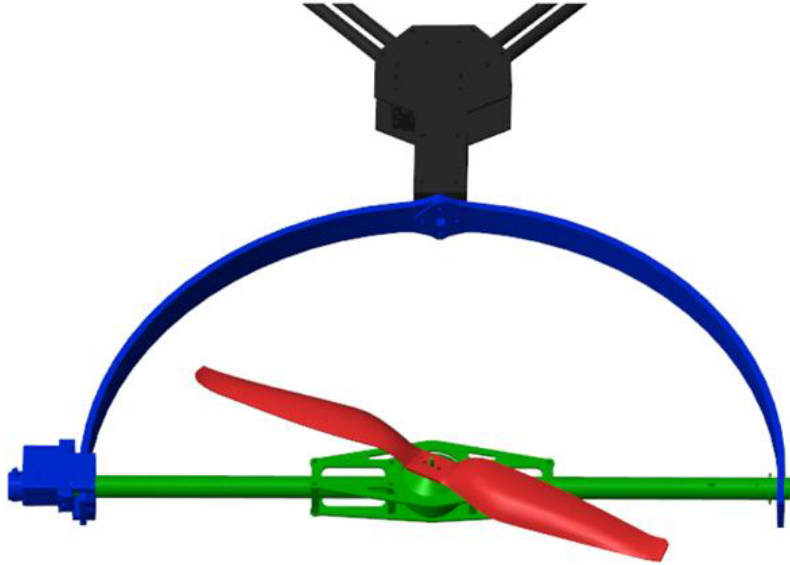


Figure 4.5: Simulation model of the single-arm setup.

and added. The encoders that track the position of each joint have inherent quantization values as well, these are measured physically and added. Finally, saturation in voltage is added and thus the model has an upper torque limit for each motor. After accounting for all the mentioned parameters, the position loops alpha and beta are each wrapped in their own controller with independent gains.

The propeller motor which actuates omega works differently than the alpha and beta motors. It is a brushless DC motor, whereas the other two motors were brushed DC motors. This motor is paired with an ESC as previously mentioned, with inherent gains that cannot be modified or read. This makes the modeling process a little more difficult, as the input to the ESC is a PWM signal and the output is a propeller's speed, and it is difficult to measure the voltage in between. Thus, to match the physical output of the ESC in simulation, the PWM signal to propeller speed relationship is mapped and added to the model. However, it is important to realize that the dynamics of the speed ramp up is not the same as the speed ramp down. The physical response of the propeller to an alternating step is shown in Figure 4.6 for reference. Notice how much faster the system can ramp up versus ramp down. This is because the ramp up is achieved by the ESC actively putting energy into the

system, but the ramp down is dependent on passive energy dissipation through air resistance and the friction inside the motor housing. Finally, it should be noted that the magnitude of thrust and drag from the propellers are dependent on this speed, and so, in simulation, this dynamic effects influences both.

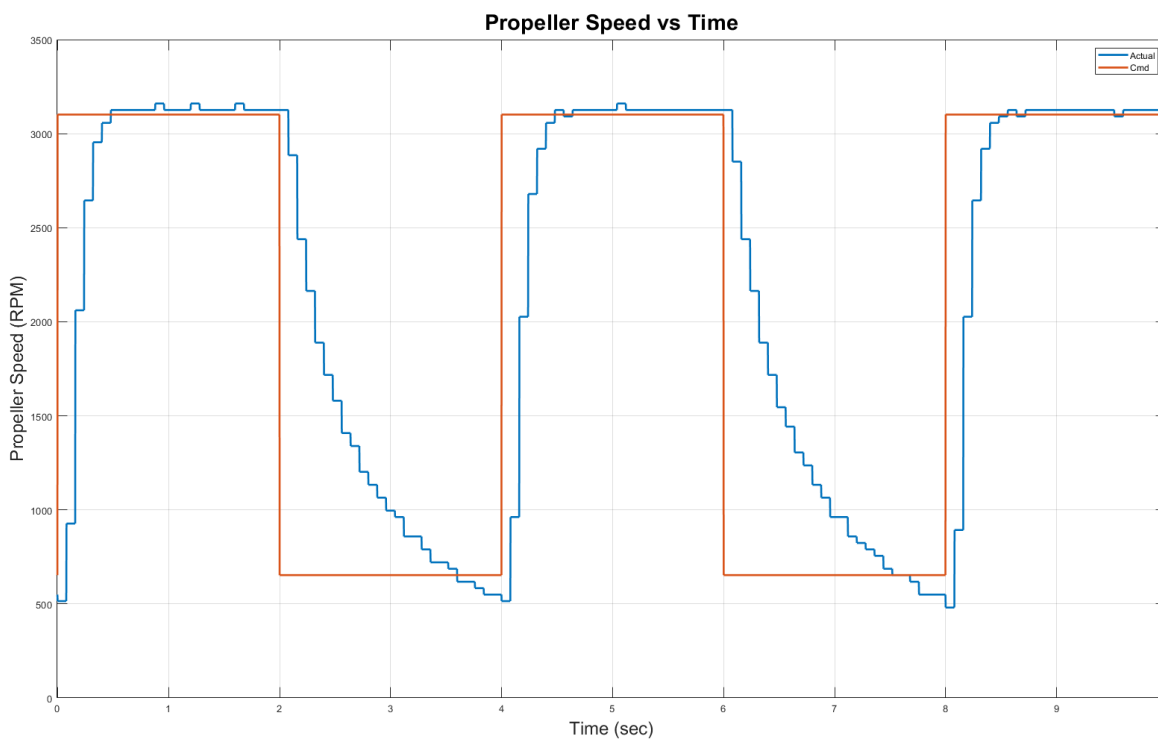


Figure 4.6: Propeller response to reference signal with ramp up and ramp down.

In the case of all three motors, notice that the complexity of the simulation model goes beyond what is discussed in the modeling section - the white box model of the system. The simulation model is intentionally made more complex to serve as a test bench for the controller that is developed using the white box model. This is done to ensure that the controller, which was designed using the less complex model, is robust enough to handle these parameters that were not modeled. This gives greater confidence in the controller's ability to stabilize the physical system, where even more un-modeled parameters are involved.

4.3.2 Experimental Setup

Just as it is important to understand how the simulation model is set up, it is equally important to understand how the physical model is set up. Figure 4.7 shows a picture of the single-arm experimental setup. The blade is placed within a cage for safety and is supplied with an external power source. The PWM commands needed to actuate the motors come from the MyRIO which is mounted on the table behind the body. The setup is very similar to the simulation model, as it should be. The arm body is grounded to the optical table and is not allowed to translate or rotate in any direction. Aside from the arm body, there are 3 actuatable rigid bodies which are allowed to rotate about their respective rotary joints, each with its own motor and controller.

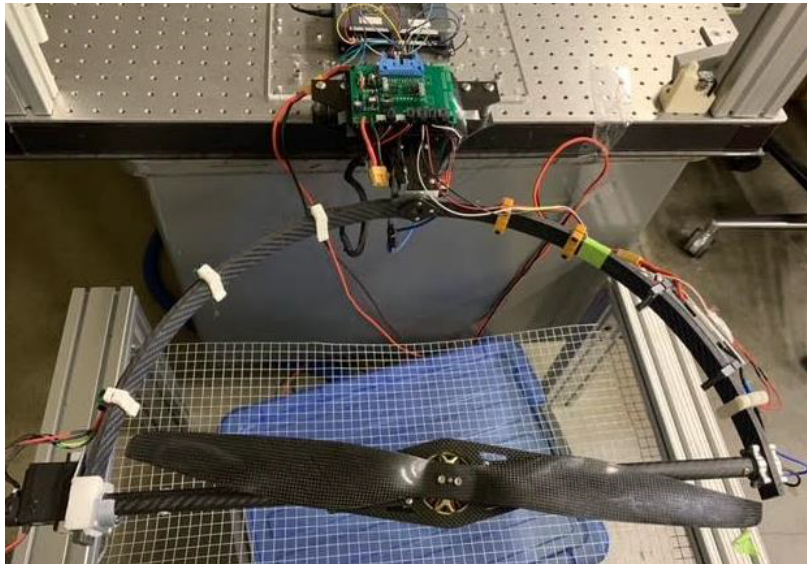


Figure 4.7: Physical model of the single-arm setup.

Unlike the simulation model, there is no need for any modeling here. The alpha and beta motors are controlled with position and gains selected by the user, while the omega motor is controlled with speed and gains that are inherent to the ESC used. The controller simply injects a PWM signal into each of the 3 motors, and the motors are actuated accordingly. One difficulty in dealing with the hardware when compared with simulation is the ability

to achieve true 0 position in the alpha and beta joint. To achieve this in the alpha joint, a homing sensor is tripped, and the arm is adjusted to its 0 position. This position, however, may still not be true 0. In the beta joint, no homing sensor is used since the joint has more friction and can hold its position during the initiation sequence. Another reason why a homing sensor isn't used here is to reduce the number of wires and wire management needed. However, this can be an issue, as it may be even more difficult to achieve true 0 position in beta.

4.3.3 Results

It is important to emphasize that the high level controller which commands the quadcopter body's 6 degrees of freedom is not in effect here. Instead, the α , β , and ω parameters are commanded directly using 3 user-defined reference signals. This setup was used to characterize the majority of the system's parameters and compare the results obtained with the results obtained in simulation. This allowed for close examination of how well the simulation was able to replicate the physical experiment's results. Although this setup only involves only 1 arm of the quadcopter, it must be stressed that this step is the key to obtaining an accurate simulation of the overall system as this step ensures that the 12 actuators that move the system are modeled correctly.

In completing this exercise, it is important to ensure that all control variables are kept constant when comparing the physical system to the simulation model. There is an individual controller in the loop of each motor, and so the gains on these controllers in the simulation and the physical setup must be selected to be identical when comparing the two. Each system is then fed the same reference signal for the angles in α and β , while choosing one of two fixed speeds for the propeller speed, ω . In the first experiment, the propeller speed is kept at 0 while the two actuation angle reference signals are changed dynamically. The results from the experiment and simulation are plotted along with the reference signal in Figure 4.8.

In Figure 4.8, the simulation shows a good overall ability of replicating the data obtained

by the experimental setup. All the major trends are captured including the rise time slope and slight overshoot. Although these results are promising, the system also needs to be tested in a more dynamic setting to give higher confidence in the results obtained by the simulation. Also, note that there is a small difference between the initial conditions of the simulation compared to the hardware. This is because it is possible to start the simulation with the motors in any position desired; however, in the physical hardware, the motors must be started at the same position every time to ensure that proper homing is established. To compensate for this, both systems are given some time to stabilize around the initial reference signals to ensure the rest of the run would be comparable.

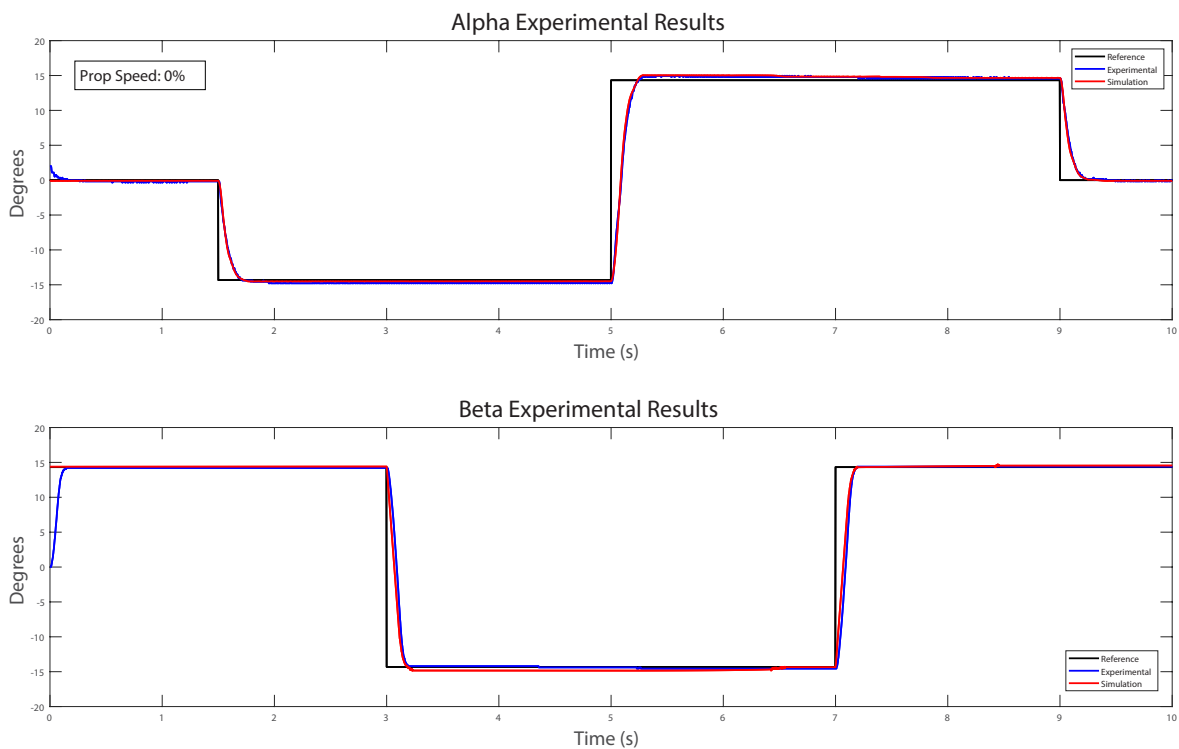


Figure 4.8: Position of α and β in experiment & simulation while propeller is off.

In the second experiment, the propeller speed is run at 75% of its maximum speed while feeding the same reference signal to the two motors angles as was done before. The objective here is to observe any changes in the simulation's behavior when there are more dynamics at

play due to the momentum of the propeller. If the moment of the propeller does introduce any deviations from the first experiment, it is valuable to see whether the simulation is able to replicate the experimental results in these conditions. The results of this run are plotted in Figure 4.9.

In this experiment, the propeller is running at a high speed and both plots now show some coupling between the alpha and beta joint actuations. It is evident that whenever one of the two joints is actuated while the other joint is commanded to stay in place, the joint that is commanded to stay in place experiences an external force that moves it away from its commanded reference. It is important to highlight that the simulation was able to predict the phenomenon that was observed in the physical hardware and that it captures the forces experienced due to this coupling. Although the exact motion due to this force is not captured exactly, the magnitude of motion is quite close to what it should be and the time it takes to get back to the reference is also close. The deviations seen here can be attributed to slight variations in how the static friction reacts in reality versus simulation, a parameter that can be difficult to model accurately. Nevertheless, this gives higher confidence in the fidelity of the model and its ability to capture the behavior of the physical system.

4.3.3.1 Coupling Effect Between Alpha & Beta

The observed occurrence that is causing the coupling between the alpha and beta joints in the Figure 4.9 is precession, a consequence of the conservation of momentum [HCZ18]. When the blade is spinning, it has angular momentum in either the positive or negative direction of thrust, depending on the propeller's spin direction. Whenever an external torque is imposed on the system in a direction which is normal to the angular momentum, a torque appears in the third axis that is normal to the plane that is created by the angular momentum and external torque applied. The direction in which the precession torque will act can be determined by using the right-hand rule. A visual representation of how this force acts on the single-arm system in each axis can be seen in Figures 4.10 and 4.11.

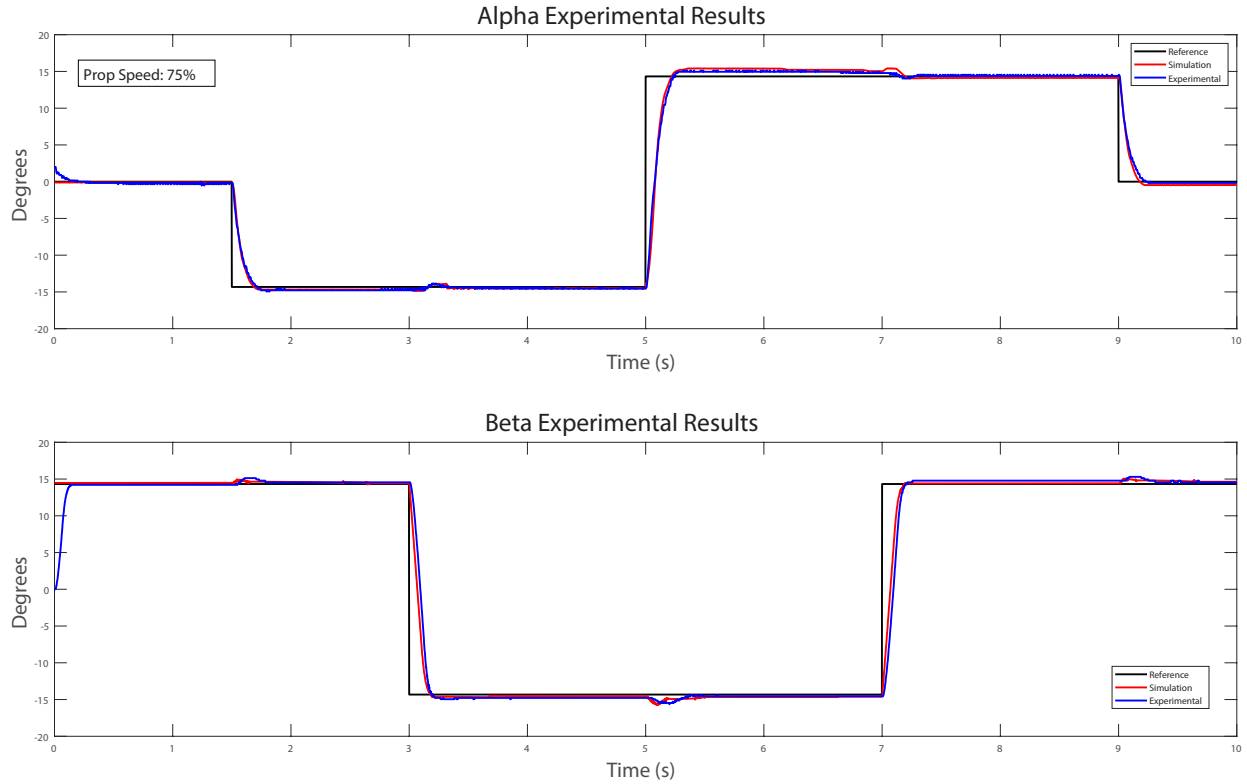


Figure 4.9: Position of α and β in experiment and simulation with propeller at 75% speed.

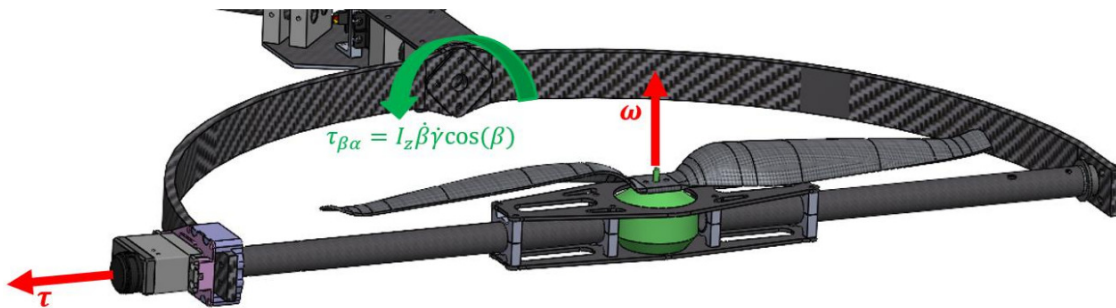


Figure 4.10: Single-arm with angular momentum, motor torque, and precession torque on α .

The equations that govern the coupling between the two axes were derived using the Energy Method and are shown in Equations 4.24 and 4.25. The equations also show that the precession torque acts on both axes, so that whenever one joint is actuated and the

propeller is spinning, the second joint experiences the precession torque. This is ultimately due to the design of the actuating arm - as there is a perpendicular intersection between the alpha and beta rotation axes, with an angular momentum vector perpendicular to the plane formed by the intersecting joints. Essentially, the design of the single-arm setup is equivalent to a gyroscope.

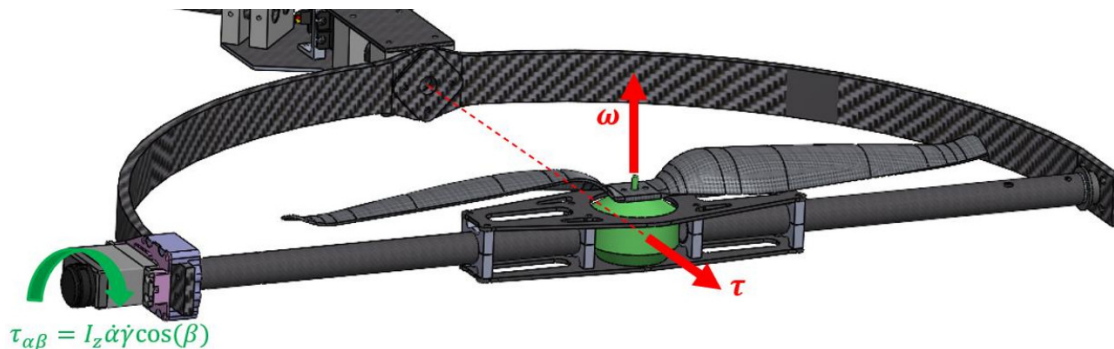


Figure 4.11: Single-arm with angular momentum, motor torque, and precession torque on β .

The equations also show that the precession torque is proportional to the propeller's moment of inertia, I_z , its speed, $\dot{\gamma}$, and the speed at which the external torque rotates the joint, $\dot{\alpha}$ or $\dot{\beta}$. There is also a geometric element in the equations which is dependent on the angle of β . The 0 position in this joint is defined to be the vertical, and is depicted in Figures 4.8 and 4.9. This is true for different reasons in each case. In the case where the external torque is applied about the alpha angle, as the β angle skews from the 0 position, it will put more of the resulting precession torque in a direction that cannot be actuated, and so the structure will take the force internally in this case. In the case where the external torque is applied about the beta angle, as the β angle skews from the 0 position, the angular momentum vector is aligned with the torque vector and so the precession effect is lost. Another noteworthy result in Figure 4.9 is that the precession's effect on alpha's position is smaller and less pronounced than the effect on beta's position even though beta has faster rotational velocity. This is because the alpha arm has a much larger moment of inertia and

so the same amount of torque in both axes will result in a smaller position change in alpha.

$$\tau_{\beta\alpha} = I_Z \dot{\beta} \dot{\gamma} \cos(\beta) \quad (4.24)$$

$$\tau_{\alpha\beta} = I_Z \dot{\alpha} \dot{\gamma} \cos(\beta) \quad (4.25)$$

In the literature, the same occurrence of precession is well researched in control moment gyroscopes (CMGs) as they have similar joint orientation and so they behave very similarly. CMGs are most commonly used on space vessels, where they are mounted onto spacecrafts to make small, controlled changes to the vessel's orientation while consuming only small amounts of energy. CMGs are well investigated and complex derivations of the dynamics involved in these systems can be found quite readily [Bau02, TYJ10]. However, there is a fundamental difference between CMGs and the quadcopter at hand. In the case of CMGs, the resultant precession torque is a desired outcome that is meant to move the vessel as planned. In the case of the quadcopter, the precession torque is an unwanted disruption to the system's tracking ability. Initially, it was suspected that this could be an issue while operating the quadcopter as these disruptions could mean sudden changes in the direction in which the propeller is facing, resulting in forces and torques in directions that are unaccounted for. Thus, depending on the magnitude of these disruptions, the precession effect has the potential to destabilize the system.

The phenomenon was thus investigated more closely both in simulation and experiment on the single-arm setup. The propeller would be spun at a constant high speed and a step reference command would be fed into one of the two actuators while the other actuator was commanded to stay at its 0 position. The displacement in the joint that was commanded to stay at its 0 position would then be examined to see how far it moved from its commanded reference. It was found that, although the effect was not negligible, the disruptions were not likely to destabilize the overall system. This is because major disruptions were only seen when the commanded angles were sudden and large in magnitude. Whereas the actuations in alpha and beta are expected to be small in magnitude and slow when the high level controller

is engaged. As Equations 4.24 and 4.25 show, low rotation speed in the actuated joint will ensure that the torque due to precession will remain relatively low. Thus, the precession torque will not disrupt the quadcopter during flight.

Nonetheless, the investigation was extended further, and an attempt was made to cancel the effects of precession for the sake of increased stability in the overall system. To do this, a feed-forward command was added to each motor to subdue the precession torque in each joint. The idea is simple, since the magnitude of the alpha command and when it will be given by the controller are known at each time step, then, theoretically, the resulting precession torque in the beta joint is also known in each time step. Thus, it should be possible to cancel the precession effect due to *alpha's* known commanded motion by commanding a torque in beta and vice versa. Of course, this is only true if the dynamics between the two axes closely resembles the form in Equations 4.24 and 4.25.

However, in practice, the experiments with the feed-forward signal showed no improvement over the ones without the feed-forward signal. These experiments failed for three main reasons, and all of them were hardware related. First, in the case of canceling the effects in the alpha axis, the alpha motor was found to be undersized for the task of compensating for the sudden torques imposed due to the beta actuations. Second, in the case of the beta joint, the magnitude of motion due to the precession torque seen was small in magnitude because the joint had considerably more friction when compared with the alpha joint and so the effects were suppressed. Third, the brushless DC motor powering the propeller would stall if the step command to either the alpha or beta motors was too large in magnitude. It is suspected that this is due to the motor becoming momentarily misaligned with the housing as the sudden rotation occurs. This meant that the precession effect at speeds beyond 75% of the propellers maximum capacity could not be investigated.

Even so, this is a very interesting potential avenue for new research that emerged from this work. By choosing a bigger alpha motor to help counter the precession torque, making the beta joint smoother to reduce friction, and choosing a sturdier motor to actuate the propeller, more reliable results may be yielded, and better testing can be conducted. Also,

to maximize the torque observed, the moment of inertia of the blade, I_Z , can be increased by replacing it with a flywheel of greater mass. Such a replacement could also allow for use of a motor other than the brushless DC which is less likely to stall during sudden rotations.

CHAPTER 5

System Control Scheme

In this chapter, two control schemes will be discussed. The first control scheme was introduced by Gerber [GT18] and takes 6 desired transitional and rotational commands on the quadcopter's body and outputs 12 individual motor commands. The second control scheme is proposed in this paper and takes 6 desired transitional and rotational commands on the quadcopter's body and outputs only 6 motor commands by constraining two sets of motors to move in a certain way. Note that in both control schemes, there are two levels of control needed to control the quadcopter. The first, and higher level, controller commands the quadcopter body's position and attitude with respect to the global frame and the body frame respectively based on the desired reference. This control loop runs at a frequency of 100Hz. The second, and lower level, controller commands the 12 system inputs which actuate the 12 motors that move the system to achieve the desired position and attitude of the overall quadcopter. This low level control loop is run at a frequency of 500Hz, 5 times what the high level controller runs at, to ensure that the low level actuators have time to attain the commands fed by the high level controller. Of the 12 motors, position control is implemented on the 8 motors controlling the alpha and beta arm angles, and angular velocity control is implemented on the other 4 motors which control the angular speed of the propeller.

5.1 12-Input Control Scheme

The first control scheme that was proposed to control the system was introduced in [GT18] by Gerber and a modified version of it is depicted in Figure 5.1. This control scheme will be called the 12-Input Control Scheme to distinguish it from the other control scheme that

will be discussed. A change is made to the control scheme diagram to include the low level controller explicitly as it was previously ignored. It is important to include this within the control scheme as there are dynamics at play in this step and ignoring them would mean that the system is assumed to achieve all desired alpha, beta, and omega commands immediately. Also, the nomenclature is slightly modified here so that the thrust and torque vectors, u_1 and u_2 , are renamed to u_F and u_τ respectively as this naming is more intuitive.

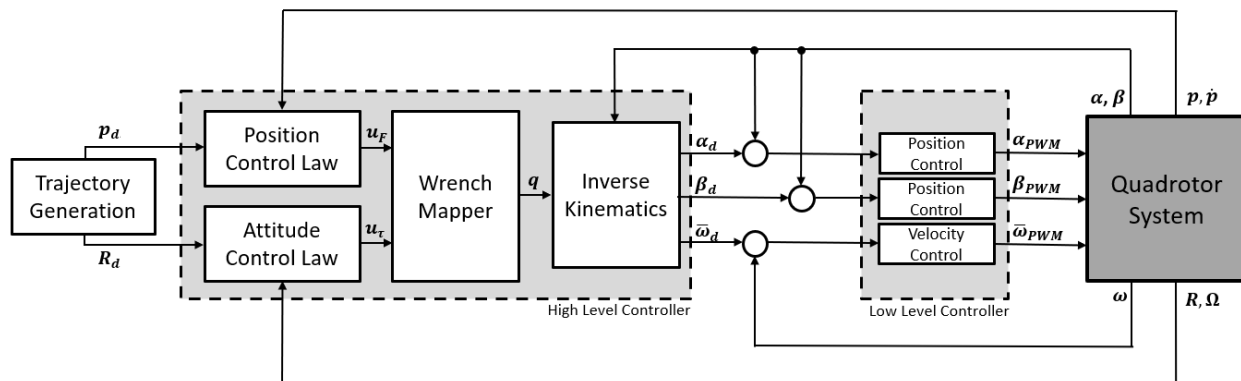


Figure 5.1: Control scheme for the 12-input quadcopter system.

The first step in the control process is to generate a desired reference trajectory for the quadcopter’s position and attitude. This step can be done manually, autonomously, or with a case structure dependant on sensor feedback. How this reference signal is designed or obtained is not the focus of this work, and so the reference signal will be chosen depending on the system capability that needs to be portrayed. Once a reference is established, the force and torque vectors at the quadcopter center of mass, u_F and u_τ , that are required to achieve the desired trajectory are generated by the position and attitude control laws respectively. Note that each of these vectors has 3 elements, one in each axis. Also, the position and attitude control laws are each internally controlled by their own controllers. The overall system can therefore be designed to be more aggressive in tracking position versus attitude or vice versa. However, balance is important here as making one loop too aggressive compared to the other can easily destabilize the system.

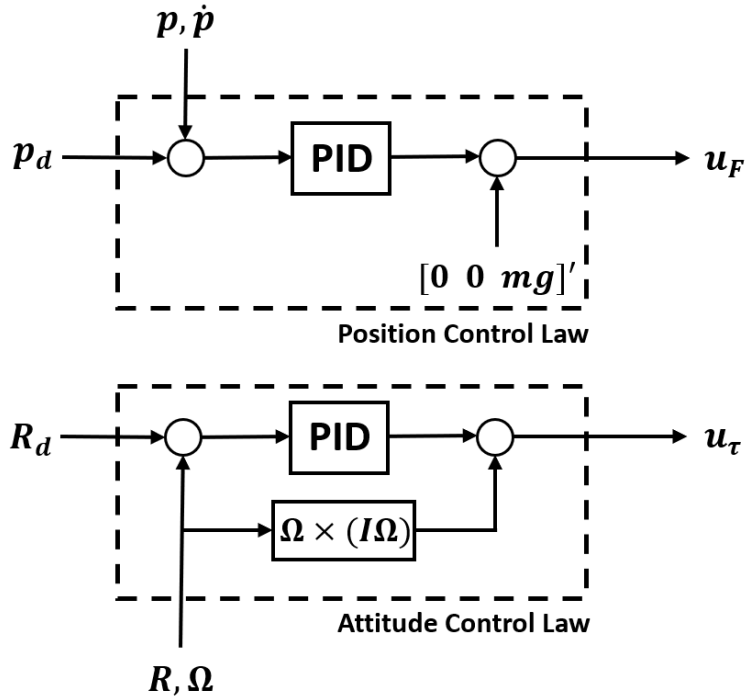


Figure 5.2: Position and Attitude Control Law.

The equations used to compute the force and torque vectors, u_F and u_τ , are shown in Equations 5.1 and 5.2 [GT18]. The controllers used in the position and attitude control laws both involve a series of 3 simple PID controllers in each axis, as shown in Figure 5.2. In the position control law, the desired position in each axis is fed into the controller from the trajectory generation step and an error signal is computed versus the actual position of the copter's center of mass. This error then goes into 3 simple PID controllers, one in each axis, and a desired force vector in 3 dimensions, u_F , is computed. Before the u_F is sent to the wrench mapper, a force equal to the copter's mass multiplied by the acceleration of gravity is added in the world frame's positive Z direction to offset the force due to the copter's mass. Because there are 3 axes and 3 gains for each axis's controller, there are a total of 9 gains to consider here - these gain values are listed in Table 5.1. It should also be noted that this computation is made in the world frame, and so the force vector is finally transformed into the body frame.

$$u_f = m\mathbf{R}^T(\mathbf{g}_0 - \mathbf{K}_p\mathbf{e}_p - \mathbf{K}_v\mathbf{e}_v) \quad (5.1)$$

$$u_\tau = \mathbf{h} - \boldsymbol{\tau}_g - \mathbf{K}_R\mathbf{e}_R - \mathbf{K}_\Omega\mathbf{e}_\Omega \quad (5.2)$$

where,

$$\mathbf{e}_p = \mathbf{p} - \mathbf{p}_d \quad (5.3)$$

$$\mathbf{e}_v = \dot{\mathbf{p}} - \dot{\mathbf{p}}_d \quad (5.4)$$

$$\mathbf{e}_R = (\mathbf{R}_d^T\mathbf{R} - \mathbf{R}^T\mathbf{R}_d), \quad \mathbf{e}_R \in \mathbb{R}^3 \quad (5.5)$$

$$\mathbf{e}_\Omega = \boldsymbol{\Omega} - \mathbf{R}\mathbf{R}_d^T\boldsymbol{\Omega}_d, \quad \mathbf{e}_\Omega \in \mathbb{R}^3 \quad (5.6)$$

Similar to the position control law, the attitude control law, also shown in Figure 5.2 is fed the desired attitude in each axis from the trajectory generation step and an error signal is computed versus the actual attitude of the copter. This error goes into 3 simple PID controllers with 9 gains which are independent from those seen in the position control law. These gain values are also listed in Table 5.1 for reference. This results in a desired torque vector in 3 dimensions, and a term, h , is finally added to this to account for change in the body's angular momentum due to the rotation of the inertia tensor, resulting in the desired torque vector in 3 dimensions, u_τ . It should be noted that the 3 integral gains are set to 0 and so, technically, this is currently a PD controller. Also, in the position control law, direct feedback of position and velocity are available, and in the attitude control law, direct feedback of attitude and attitude change rate are available. Therefore, full state feedback of the system is available here, as all states are measured directly, and no states are inferred from other states.

	P	I	D
X Position	5.55	0.12	1.665
Y Position	5.55	0.12	1.665
Z Position	6.5	0.5443	3.2
X Rotation – Roll	8.0	0	3.0
Y Rotation – Pitch	8.0	0	3.0
Z Rotation – Yaw	5.0	0	0.1

Table 5.1: List of gains for position and attitude control laws.

Using the u_F and u_τ vectors generated in the previous step, the wrench mapper can now map the force and torque needed at the quadcopter center of mass to the alpha and beta angles as well as the propeller speed needed on each of the 4 arms via a vector of 12 q values. The goal here is to create a force and torque on each arm which collectively create the force and torque needed at the center of mass. With 3 q 's per arm, the propeller on each arm can be pointed in the desired direction with the desired magnitude by manipulating alpha and beta angles and the propeller speed. Collectively, the 4 arms should now give a resultant force and torque on the quadcopter body equivalent to u_F and u_τ . Since there are only 6 elements in the U vector, made up of u_F and u_τ , and 12 elements in the q vector, the wrench mapper matrix cannot be square and must have 6 rows and 12 columns. The wrench mapper equation from [GT18] is shown in Equation 5.7, where W is the wrench mapper.

$$\mathbf{U} = \mathbf{W}\mathbf{q} \tag{5.7}$$

$$\begin{bmatrix} u_{F,x} \\ u_{F,y} \\ u_{F,z} \\ u_{\tau,x} \\ u_{\tau,y} \\ u_{\tau,z} \end{bmatrix} = \begin{bmatrix} c_p \mathbf{I}_3 & \dots & c_p \mathbf{I}_3 \\ \mathbf{J}_1 & \dots & \mathbf{J}_4 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_{12} \end{bmatrix} \quad (5.8)$$

where,

$$\mathbf{J}_i \equiv c_p \mathbf{d}_i + (-1)^i c_\tau \mathbf{I}_3, \quad \mathbf{J}_i \in \mathbb{R}^{3 \times 3} \quad (5.9)$$

To find the q vector required to achieve the desired U vector, the inverse of the wrench mapper matrix, W , must be taken and multiplied from the left. Since this matrix is a 6 by 12 matrix, no inverse exists. Instead, a pseudo-inverse of the matrix, W^+ , must be found and used. Note in Figure 5.1 that the wrench mapper comes after the position and attitude control laws. A visual representation of how the wrench mapper converts u_F and u_τ to the q 's on each arm is shown in Figure 5.3.

$$\mathbf{q} = \mathbf{W}^+ \mathbf{U} \quad (5.10)$$

where,

$$\mathbf{W}^+ = \mathbf{W}^T (\mathbf{W} \mathbf{W}^T)^{-1} \quad (5.11)$$

Once the q vector is found, the 3 q values on each arm still need to be transformed into alpha, beta and omega positions and speed values as these are the actuatable parameters. They are mapped onto each arm via inverse kinematics, this can also be seen in Figure 5.1. At this stage, the alpha, beta, and omega desired commands needed to achieve the desired U vector are known. The final step is to feed these commands into the low level controller which will send the pulse width modulation (PWM) signal to a PID controller which is

paired with each individual motor to achieve the desired position or speed. The gains used in the PID loop of each alpha and beta motor is listed in Table 5.2 for reference. Note that omega is not included here as its control loop is done internally by the ESC and so these values are unknown. Also, all alpha and beta motors are assumed to be identical and so one set of gains are used for all alpha motors and another set is used for all beta motors even though there are slight differences in the hardware. Finally, the process is then reiterated with feedback from the quadcopter body’s position and attitude, along with the derivative of these values, and the cycle continues.

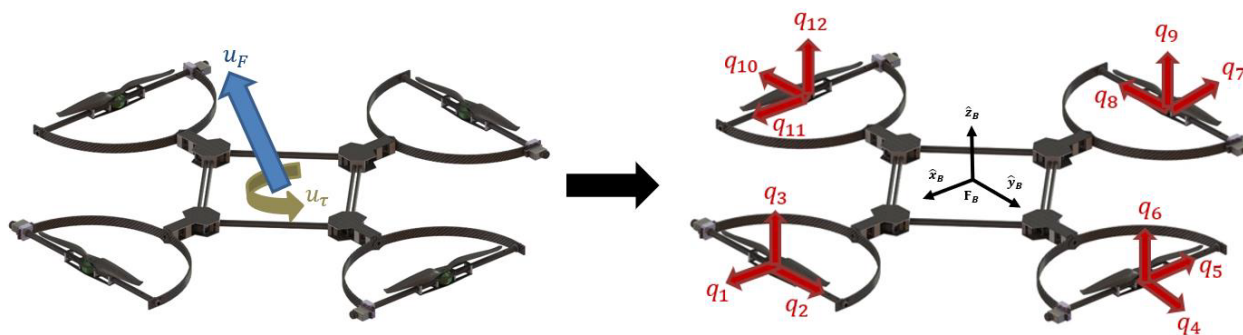


Figure 5.3: Visual of wrench mapper converting u_F and u_τ into 12 q vectors.

As discussed, in this system, there are 12 input parameters, but only 6 controllable variables – the 6 translational and rotational degrees of freedom. This means that by design, the system has a high degree of redundancy and for any given set of u_F and u_τ there are multiple solutions. This means that the controller must now decide which solution to select based on some chosen criteria. And so, the controller chooses the solution that minimizes the sum of the propeller speeds squared by using the Moore-Penrose right pseudo-inverse to invert the wrench mapper matrix [GT18]. Gerber argues that doing this minimizes the propeller speeds and so this will automatically select the most energy efficient solution available to the controller since the propellers account for most of the energy drawn by the system.

	P	I	D
Alpha Motor	6.5	6.5	0.09
Beta Motor	18	4	0.3

Table 5.2: List of gains for alpha and beta motors.

5.2 6-Input Control Scheme

Although the 12-Input Control Scheme should work well, the scheme is overactuated and so, as discussed, a given set of u_F and u_τ will have numerous solutions to the 12 motor outputs. If there were only 6 controllable variables instead of 12, a more simple and elegant solution could be attained since the wrench mapper, W , would be a square matrix of 6 by 6. In such a configuration, there would only be one possible solution for each set of u_F and u_τ values and so the redundancy of solutions is also eliminated.

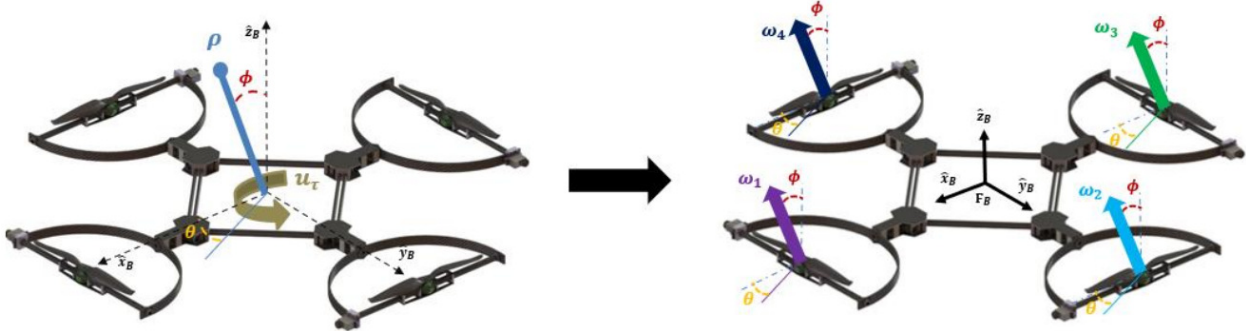


Figure 5.4: Visual of modified wrench mapper converting u_F and u_τ into ρ , ϕ , and θ .

Such a system is possible by imposing a constraint upon the wrench mapper to constrain the directions that the propellers are allowed to face. The constraint imposed is simple, all 4 propellers must always point in the same direction in 3 dimensional space. Naturally, this direction will be chosen to be in the direction that u_F is pointing. Effectively, this constraint results in two sets of 4 motors being constrained to move together. The first set of motors is α_1 , α_3 , β_2 , and β_4 . The second set of motors is α_2 , α_4 , β_1 , and β_3 . The 6-Input Control

Scheme will thus give the first set of motors a single position command for all 4 motors to track and give the second set of motors a single position command for all 4 motors to track. In addition to the ϕ and θ commands, the controller still has control over the 4 propeller speeds, ω_i , leading to a total of 6 system inputs and 6 degrees of freedom. For the purposes of this controller, the direction and magnitude of the u_F vector are more conveniently defined in spherical coordinates, and so the u_F vector is converted into a magnitude, ρ , and two angles, ϕ and θ . A visual representation of this is shown in Figure 5.4, and each of these three parameters are defined in Equations 5.12 through 5.14.

$$\rho = \sqrt{u_{F,x}^2 + u_{F,y}^2 + u_{F,z}^2} \quad (5.12)$$

$$\phi = \cos^{-1}\left(\frac{u_{F,z}}{\sqrt{u_{F,x}^2 + u_{F,y}^2 + u_{F,z}^2}}\right) \quad (5.13)$$

$$\theta = \tan^{-1}\left(\frac{u_{F,y}}{u_{F,x}}\right) \quad (5.14)$$

$$\begin{bmatrix} C_p & C_p & C_p & C_p \\ -C_\tau c\theta s\phi & C_p D_{Arm} c\phi + C_\tau c\theta s\phi & -C_\tau c\theta s\phi & -C_p D_{Arm} c\phi + C_\tau c\theta s\phi \\ -C_p D_{Arm} c\phi - C_\tau s\theta s\phi & C_\tau s\theta s\phi & C_p D_{Arm} c\phi - C_\tau s\theta s\phi & C_\tau s\theta s\phi \\ C_p D_{Arm} s\phi s\theta - C_\tau c\theta & -C_p D_{Arm} s\phi c\theta + C_\tau c\phi & -C_p D_{Arm} s\phi s\theta - C_\tau c\phi & C_p D_{Arm} s\phi c\theta + C_\tau c\phi \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \begin{bmatrix} \rho \\ u_{\tau,x} \\ u_{\tau,y} \\ u_{\tau,z} \end{bmatrix} \quad (5.15)$$

With the change specified, a 6 degree of freedom system with 6 system inputs is created, and so the system is simplified. However, the output from the Position Control Law to the wrench mapper is no longer u_F , but ρ , θ , and ϕ . Also, the q vector that is output from the wrench mapper now has only 6 output parameters and so it is renamed to F to differentiate it from its previous nomenclature when it had 12 output parameters and avoid confusion with the convention commonly used in the field. The changes discussed here are reflected in Figure 5.5.

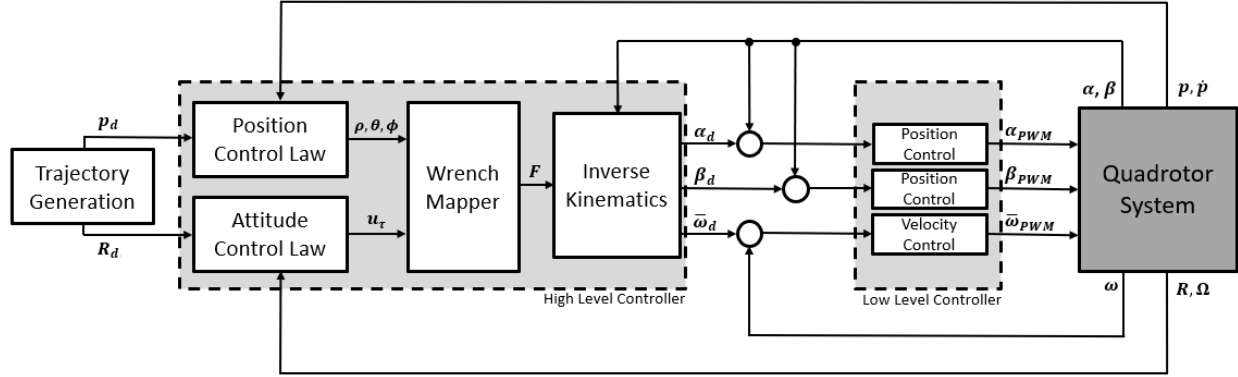


Figure 5.5: Control scheme for the 6-input system quadcopter.

With the exception of the wrench mapper, the two control schemes are quite similar. By comparing Figures 5.1 and 5.5, it is clear that the high level architecture of the controllers is still similar for the 6-Input and 12-Input Control Schemes.

5.2.1 6-Input Control Scheme Singularity Cases

When the system is altered with the 6-Input Control Scheme, a singularity that is not present in the 12-Input Control Scheme is introduced into the system. This singularity is strictly due to the restriction imposed in the wrench mapper, and occurs when an angle of $\phi = \frac{\pi}{2}$ is chosen, regardless of the θ angle chosen. An angle of $\phi = \frac{\pi}{2}$ puts the thrust vector, u_F , purely on the body frame's X-Y plane. This results in the wrench mapper taking on the form shown in Equation 5.16.

$$W(\phi = \frac{\pi}{2}) = \begin{bmatrix} C_p & C_p & C_p & C_p \\ -C_\tau c\theta & C_\tau c\theta & -C_\tau c\theta & C_\tau c\theta \\ -C_\tau s\theta & C_\tau s\theta & -C_\tau s\theta & C_\tau s\theta \\ C_p D_{Arm} s\theta & -C_p D_{Arm} c\theta & -C_p D_{Arm} s\theta & C_p D_{Arm} c\theta \end{bmatrix} \quad (5.16)$$

Although it may not be immediately obvious, this is a linearly dependent matrix of rank 3, and so this is the source of the singularity introduced to the system whenever $\phi = \frac{\pi}{2}$.

Whenever all thrust vectors sit on the quadcopter body frame's X-Y plane and are forced to point in the same direction, a restriction the modified wrench mapper imposes, the singularity occurs because the torque vector in one of the two directions normal to the thrust direction is no longer attainable by differentiating the propeller speeds. For example, in Figure 5.6 the system is constrained with $\phi = \frac{\pi}{2}$ and $\theta = 0$.

$$\begin{bmatrix} C_p & C_p & C_p & C_p \\ -C_\tau & C_\tau & -C_\tau & C_\tau \\ 0 & 0 & 0 & 0 \\ 0 & -C_p D_{arm} & 0 & C_p D_{arm} \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} = \begin{bmatrix} \rho \\ u_{\tau,x} \\ u_{\tau,y} \\ u_{\tau,z} \end{bmatrix} \quad (5.17)$$

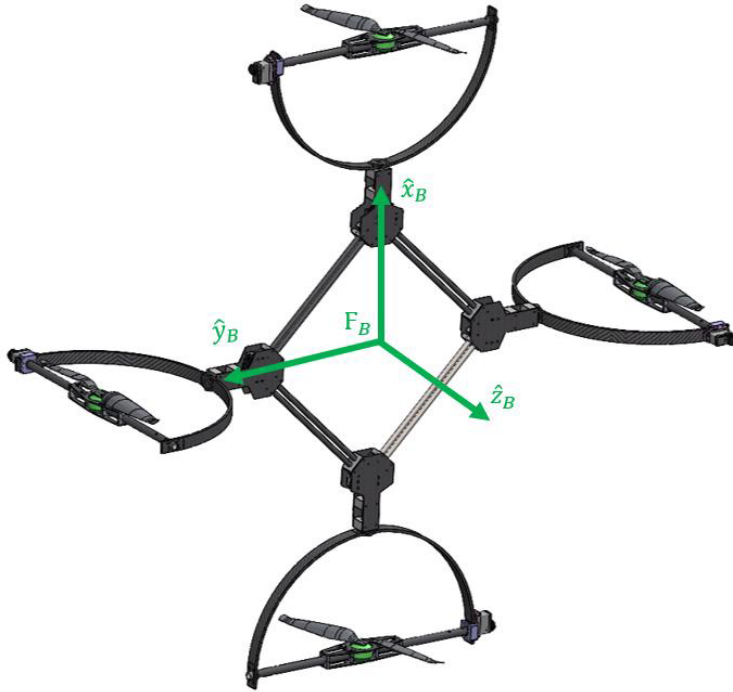


Figure 5.6: Quadcopter configuration when $\phi = \frac{\pi}{2}$ and $\theta = 0$.

The wrench mapper matrix for this specific orientation is shown in Equation 5.17 with the values of ϕ and θ plugged in. A row of zeros appears in the 3rd row of matrix which

means that control over torque in the Y direction of the quadcopter body frame has been lost. Once again, it should be noted that this is not the only orientation in which a singularity exists, as any orientation with $\phi = \frac{\pi}{2}$ with any angle of θ will also present a singularity. The main aim of this section was to identify the presence of the singularity in the system; but no method of dealing with the singularity is discussed here. However, a solution to similar singularity has been proposed in the literature [YSG21].

CHAPTER 6

Four-Arm Simulation & Experiments

In this chapter, the simulations and experiments conducted on the complete copter and their results will be discussed. The physical parameters that were obtained from the single-arm setup are reused when creating the simulation for the complete system. There are several differences when compiling the four-arm system. Obviously, instead of just 1 arm, there are now 4 arms connected to the central body. Also, the quadcopter body is no longer grounded as it was in the single-arm experiment, and so the structure is allowed to translate and rotate freely in space as a result of the forces and torques generated by the propellers. Furthermore, unlike the single-arm setup, the high level controller is now in full effect and so there the 6 translational and rotational degrees of freedom are being actively regulated. This also means that the individual input commands to each arm's α , β , and ω are no longer controlled explicitly as they were in the single-arm system. Instead, a desired quadcopter body position and attitude reference is passed through the high level controller and the low level inputs are calculated by the controller automatically.

6.1 Simulation Setup

The four-arm quadcopter in simulation is shown in Figure 6.1 for reference. As was the case with the single-arm system, each color change extending from the central body represents an independent and rigid individually actuatable body. In addition, the MyRIO and the takeoff platform are both modeled and added to the system. The MyRIO is important to model as it introduces an offset mass that will affect the outputs of the system. This will

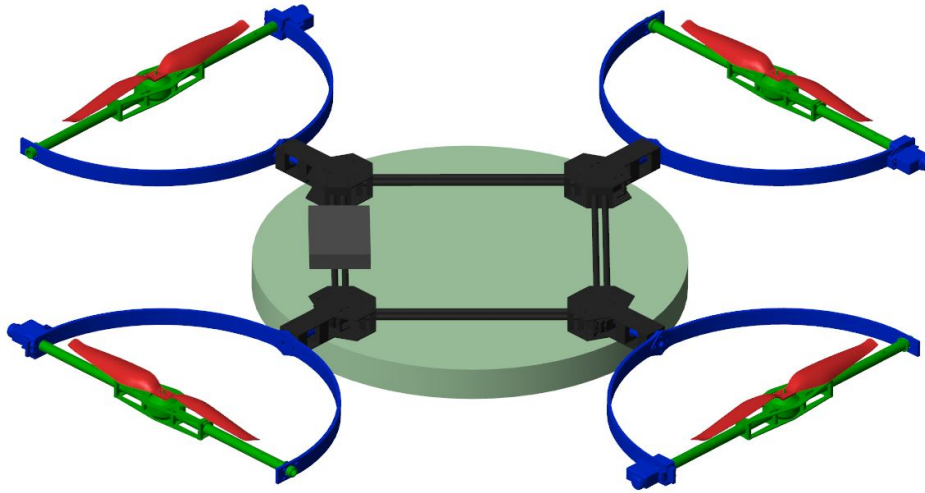


Figure 6.1: Four-arm setup in simulation with MyRIO and platform shown.

result in a force bias in Z translation, roll, and pitch. The takeoff platform is also important to model as it is the initial condition that the quadcopter starts from physically and so this should be kept constant for a fair comparison to hardware results.

The model includes the high level controller running at frequency of 100Hz and the low level controller running at 500Hz, to match the hardware setup. Also, the position and attitude data obtained from the simulation can be instantaneous and perfect if desired. However, this would not match the hardware, as the sensors have variable amounts of noise in their position and attitude readings. This noise is thus added to the simulation with the goal of mimicking the noise in the sensors that are used on the actual hardware. The noise is recorded on the hardware by keeping the sensor stationary and recording the output of the sensors, the result of this is shown in Figure 6.2 for reference. This noise is considered and added in simulation to ensure an accurate representation.

It is important to model relevant aspects of aerodynamics as these will be the drivers of the system in space. The aspects of aerodynamics deemed the most relevant are the thrust and drag forces of the propeller. These are now more relevant, as they will have direct effects on the body's position and attitude since the body is no longer grounded. However, this

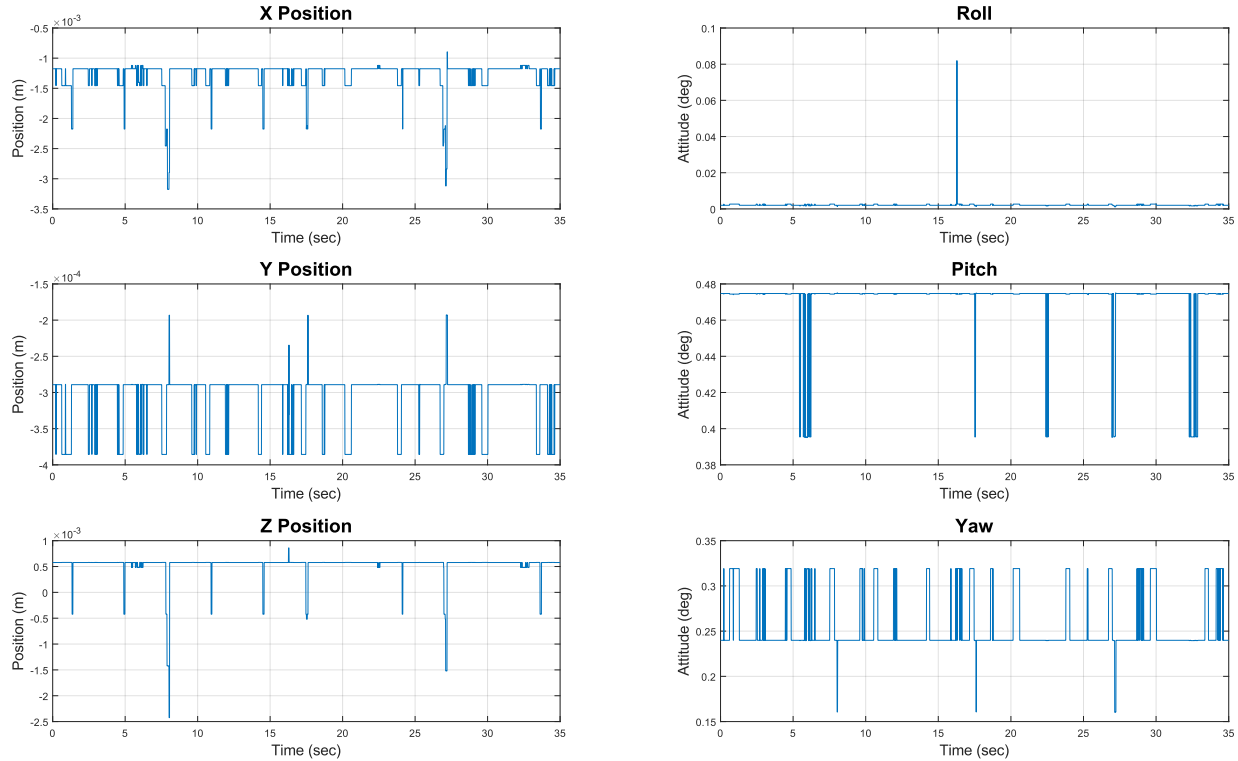


Figure 6.2: Noise in CrazyFlie and Lighthouse sensors.

does not mean that there aren't other aspects of aerodynamics that are important in certain situations; for instance, when flying close to walls or when stacking two blades on top of each other. In the experiment, there is a large amount of air moving around the system, known as blowback, this is not considered either. However, these effects are considered less essential to the model in most situations when compared with thrust and drag, and so they are not modeled for this reason. Furthermore, an assumption of rigid bodies is made for all the parts that are included in the model.

6.2 Simulation Results

The differences in the control scheme and, more specifically, the wrench mapper between the 12-input and 6-input control schemes have already been explained. In this section, the two schemes' simulation tracking results are compared by giving both systems the same reference

signal to track in position and attitude over a 40 second run in simulation to shed light on how the controllers work at a low level. The reference signal given here allows for large position changes in the X, Y, and Z axes as there is no space restriction here. The reference is chosen to command a positive translation in the X, Y, and Z axes to leave the platform and move in space, and then it commands all 3 attitude changes in succession to ensure the system can achieve all 3 attitude degrees of freedom independently. In these simulation runs, all the parameters and controller gains at the high and low levels are kept constant across the 12-input and 6-input systems to ensure a fair comparison. The tracking results of both simulation runs are plotted in Figure 6.3 with 3 plots for position, on the left, and 3 plots for attitude, on the right. This format of plot will continue to be used to report results of the copter's position and attitude.

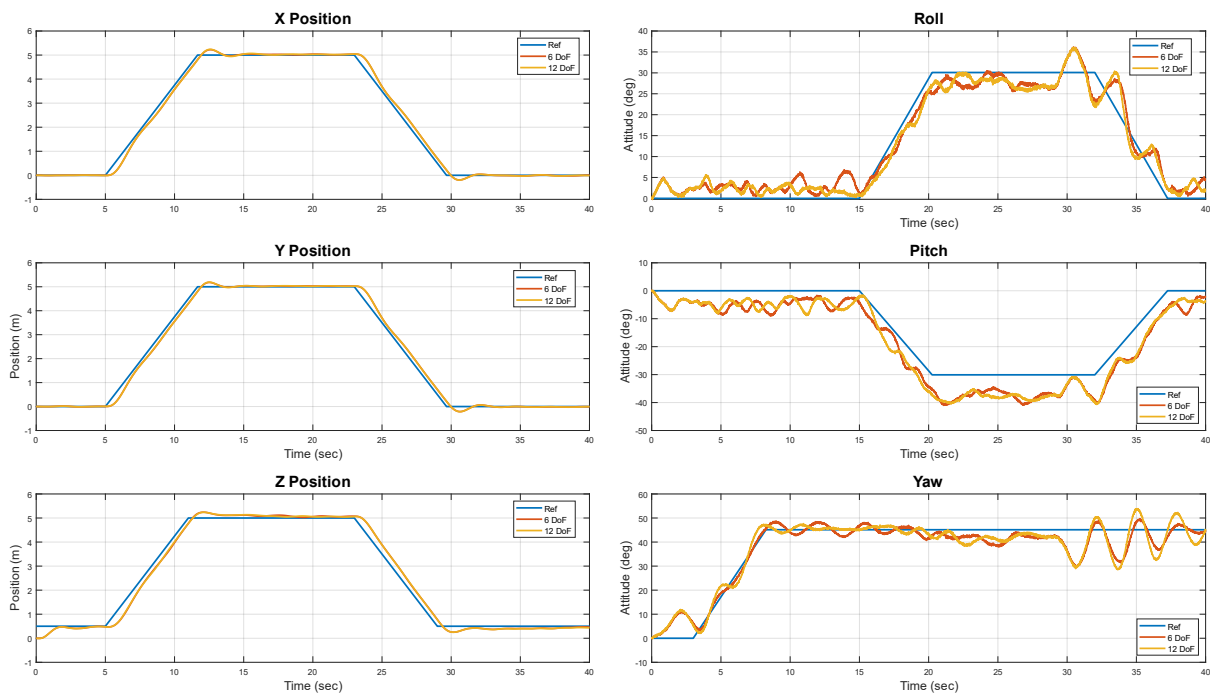


Figure 6.3: Position and attitude tracking results in simulation for both controllers.

Overall, the two controllers in simulation perform very comparably when considering the tracking results in Figure 6.3. The responses of the two systems in X, Y, and Z position are

almost identical. The differences come about in the 3 attitude responses. Each controller does slightly better in some degrees of freedom and worse in others. The reason the difference is seen primarily in the attitude tracking is because the changes made to the wrench mapper for the 6-input controller force the system to only produce yaw torque by creating a differential speed between clockwise and counterclockwise blades. The 12-input controller, on the other hand, is allowed to create a yaw torque by pointing the blades onto an axis that would give thrust in the desired yaw direction. This will influence the 6-input controller system's yaw results compared to the 12-input controller system. As a consequence of the propeller dynamic shown in Figure 4.6, the 3 attitudes becomes very coupled. For instance, if a pure positive pitch is commanded, both controllers will command the omega 3 to speed up and omega 1 to slow down by the same amount and at the same rate. However, due to the propeller dynamic discussed, the increase rate will be higher than the decrease rate. And this will cause disturbance to the Z axis position. This will in turn, slow down and speed up more propellers which further exacerbates the coupling in the 3 attitudes. However, all the positions will react the same in both systems because the change made in the wrench mapper does not change the method with which these 3 actuations are performed.

Although the tracking responses of the two systems are quite similar, there are big differences in how these tracking results are achieved, specifically in yaw. The two control systems' low level alpha, beta, and omega responses for each arm are plotted in Figure 6.4 for a simulation run that commands a pure yaw at 2.5 seconds. The 12-input system actuations are shown on the left and the 6-input system actuations are shown on the right for comparison.

Vast differences between the two controllers are evident when examining Figure 6.4. Both systems deal with an initial negative yaw that is induced onto the system on takeoff. And at 2.5 seconds, both systems track a commanded positive yaw. This plot makes it clear that the 12-input controller is indeed using part of its thrust vectors in four arms to produce a yaw torque by actuating the alpha arms instead of using the drag torque like a

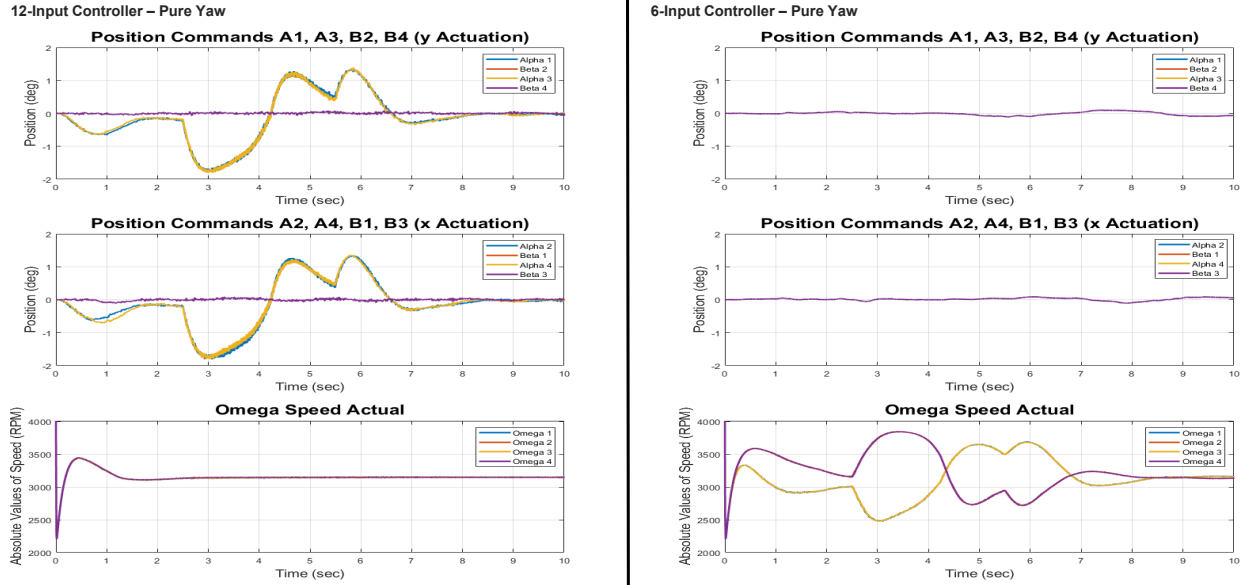


Figure 6.4: Response of the 12 actuators in simulation for both controllers under pure yaw.

traditional quadcopter would. While doing this, the controller keeps all propeller speeds across all 4 blades identical. On the other hand, the 6-input controller keeps all alpha and beta angles very close to their 0 positions, meaning the propellers are pointing straight up and the thrust is only used to counter the effect of gravity. When a change in yaw is commanded, the controller simply increases the difference in speed between the clockwise spinning blades, omega 2 and 4, and the counterclockwise blades, omega 1 and 3. Ideally, this does not change the average thrust produced by the system, and so no change should be seen in the Z position. However, the difference in speed between the two pairs of propellers creates a torque about the Z axis due to the difference in drag torque generated. Although there were many instances of this happening in the 40 second run shown previously, showing a pure yaw command showed the phenomenon more clearly.

6.3 Experimental Setup

The four-arm experimental setup is shown in Figure 6.5. The copter is built to be as symmetric as possible to reduce the number of biases that the motors must deal with.

The biggest exceptions are the beta motor bias imposed on the alpha arm, and the MyRIO microcontroller, which is mounted on one of the connector plates and so it sits off the copter's center of mass. Due to where it is mounted, the MyRIO's mass will inflict a negative bias force on the Z axis, a positive bias torque on roll, and a negative bias torque on pitch at all times. It's worth taking note of these biases when looking at the experimental results to come. The high and low level controllers both run on the MyRIO microcontroller at frequencies of 100Hz and 500Hz respectively.

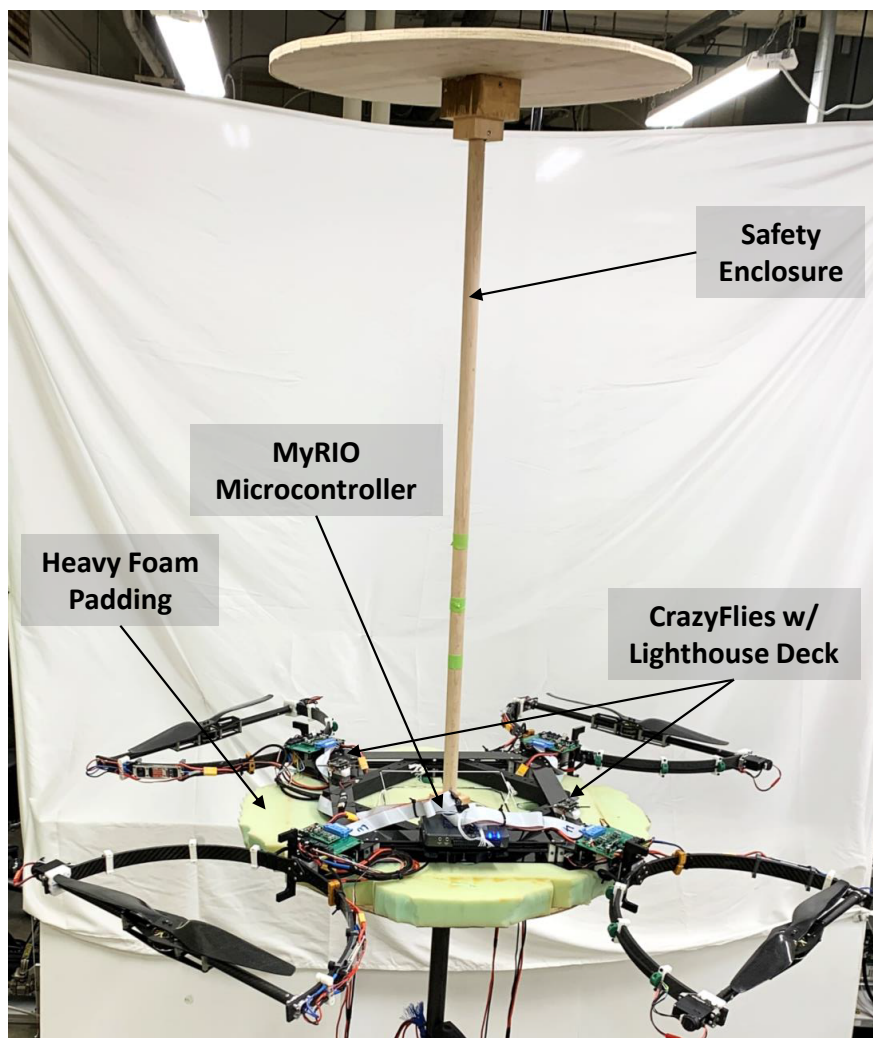


Figure 6.5: Picture of four-arm physical setup with safety enclosure.

As stated before, the high level controller is fully operational here, unlike the single-arm

setup, and the copter is now allowed to fly within the confines of a safety enclosure. The safety enclosure is used because the copter is set up to fly in a tight, indoor space. The safety enclosure that the copter sits in allows the copter to perform attitude changes, up to a certain limit, but restricts its translations in the X, Y, and Z axes. The enclosure allows the copter to move 12cm in the positive and negative directions of the X and Y axes, after which a physical limit is reached, and the copter is stopped from moving any further in these directions. The copter also has limits in the Z axis where it has 1 meter of space to move within. The bottom limit is padded with heavy foam to ensure that the copter does not get damaged on impact or heavy landing from high altitude and the top limit is un-padded but would simply stop the copter from moving any further in the vertical direction in case of instability. If a limit is hit in any direction, the experiment is considered a failure and redone, as hitting the limits alters the system being controlled and so the outcome is no longer relevant.

The CrazyFlies with a Lighthouse Deck are the markers used to determine the copter's overall position in the X, Y, and Z position. It is thus important that they are always visible to the Lighthouse beacons. If either of the CrazyFlies lose line of sight to both beacons at the same time, then the position is temporarily lost in this CrazyFlie and there is thus a high risk of the system becoming unstable due to this, unless the signal is regained quickly. Therefore, the positioning of the Lighthouse beacons is very critical when considering the actuation that is to be conducted on the system.

6.4 Experimental Results

The performance of the complete four-arm physical system will be tested extensively in this section. Several commands will be applied ranging from a simple hover to a combination of roll, pitch, and yaw. The same controllers used in the simulation will be reused here to see if they are robust enough to stabilize the system. The performance of the two controllers will be compared to one another to see if the differences seen in simulation still hold on the physical hardware.

6.4.1 Hover



Figure 6.6: Picture of copter while hovering.

The first command given to the system was a slightly positive Z position reference while commanding 0 on all the other degrees of freedom. In short, the copter was commanded to hover at low altitude which is the most manageable command when gauging the performance and determining whether the controller is working as intended. This reference command was fed to the system numerous times while fixing bugs in the controller's code. Finally, the copter was successfully commanded to hover with both controllers, it is shown hovering in 6.6 and the results of the run are shown in Figure 6.7.

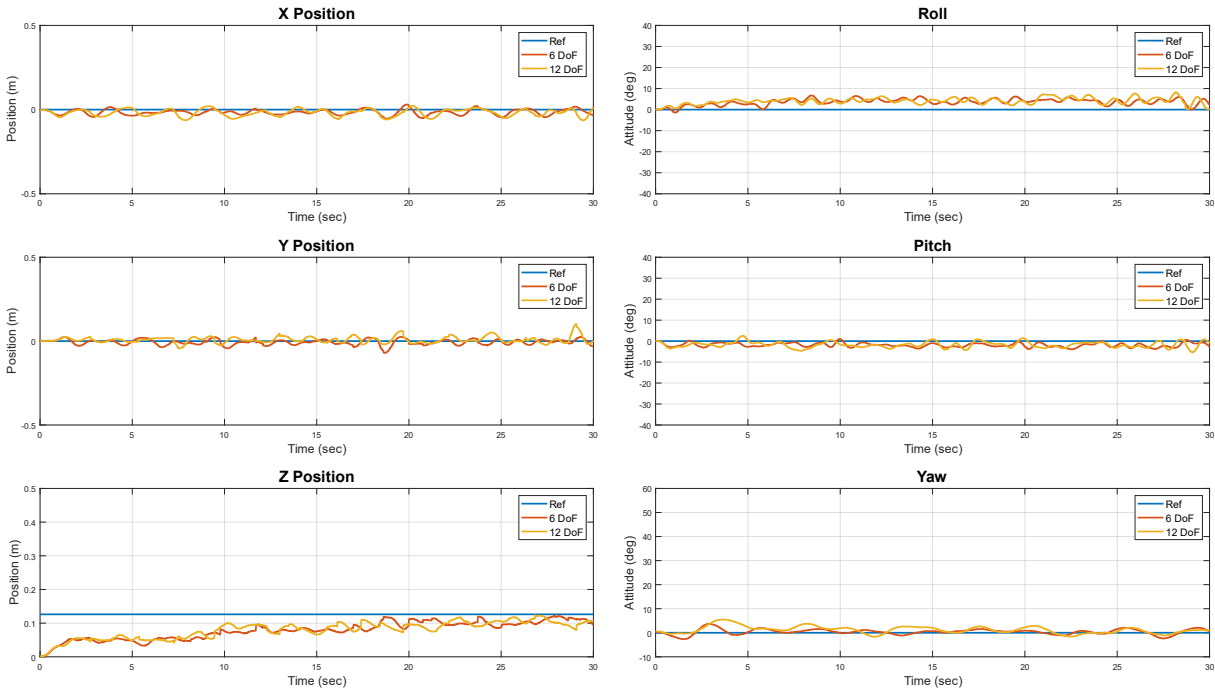


Figure 6.7: Tracking results of both controllers while hovering.

Although the system was not commanded to do much here, the hover data reveals a few things about the system. First, although the overall tracking is acceptable, the oscillations seen in the simulation data are indeed present in the response. Second, a positive bias is seen in roll, and a negative bias is seen in pitch. This was expected and is caused by the MyRIO's mass since it is not mounted at the copter's center of mass. Third, the two controllers perform very similarly, as predicted by the simulation results. The 6-input controller may have performed slightly better, but this can probably be attributed to the repeatability of the experiment and not the performance of the actual controller. Most importantly, the data shows that the controller developed using the white-box model, and tested using the more complex simulation model, is robust enough to stabilize the system. One deviation from the simulation that should be pointed out is that the Z position response reaches the commanded level much faster in simulation than it does in reality. This is because the battery was worn out when this experiment was conducted, where it was found that when the batteries were freshly charged, the response would vary quite a bit. It's also worth noting

that the hover height, Z position, was chosen to be quite small to ensure that the copter has enough space to hover but not too high to ensure that the copter would not be damaged in case of instability and would land on the foam pad that is quite close.

6.4.2 Yaw

The command attempted after the simple hover was achieved is the yaw command. This is because the yaw command poses low risk of damage to the hardware with almost no chance of hitting the safety enclosure in any way. This is first attempted on the 6-input controller, which would mimic a traditional drone when yawing, and so its outputs are more easily deciphered to understand what the controller is trying to do. In the case of a positive yaw, the 6-input controller will increase the speeds of the clockwise spinning propellers, omega 2 and 4, and slow down the counterclockwise spinning propellers, omega 1 and 3. The 12-input controller yaws by actuating the 4 alpha arms and putting some portion of thrust in the direction that would achieve the yaw command. The results of the yaw command for both controllers can be seen in Figure 6.8. The yaw command is given after about 7 seconds when the system has achieved a stable hover. A yaw of 45 degrees is commanded here, but up to 90 degrees was attempted with success. The reason 45 degrees was finally selected as a limit is due to the power cables becoming more taut at actuations greater than this number. The copter does a good job of achieving this degree of yaw without tensing up the power cables. Overall, the two systems perform very similarly here in terms of position and attitude, even though the yaw is achieved differently on a low level.

Another feature that stands out in the command signal is that it is rate limited with a slew rate of 0.15 radians per second, or roughly 8.6 degrees per second, in either direction. This is done to protect the system from becoming unstable due to sudden, large changes in error which was first noticed in simulation. The yaw command showed the most resilience out of the attitude commands in simulation and so its slew rate can probably be pushed further, but this limit was not tested on the hardware. Also, notice that the system's reaction

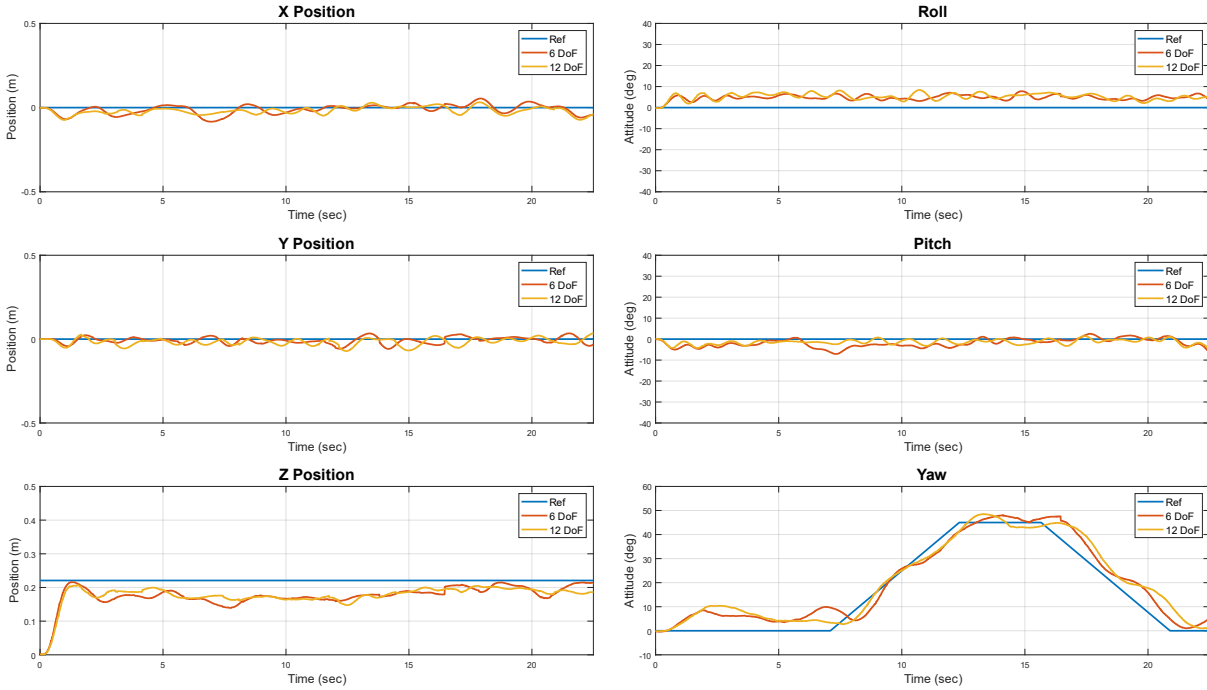


Figure 6.8: Tracking results of both controllers while yawing.

to the Z position command is very different than the response in hover. This is because this run was done after the batteries being used were charged, and so the propellers can muster more thrust at the same PWM command. This is due to the map from PWM command to propeller speed, which was done somewhere closer to the full charge. So, as the battery is used a few times after being charged, the mapping slowly starts to move away from the mapping made. This can be rectified by ensuring the batteries are always well charged, or by accounting for the battery's voltage and performing the mapping from PWM to propeller speed at numerous voltages. The voltage would then need to be measured and recorded before each physical run so that the effect of variable voltage can be accounted for in the simulation model. Finally, the Z position is still chosen to be relatively small, as a yaw actuation does not require clearance and a small Z position is sufficient to complete the commanded actuation while minimizing the risk of damaging to the copter.

6.4.3 Roll & Pitch

The individual actuations of roll and pitch are lumped into one subsection as they are basically the same actuation using different sets of omega motors. To achieve a positive roll, the controllers would speed up the omega 2 propeller while slowing down the omega 4 propeller by the same amount and, ideally, at the same rate. To achieve a positive pitch, the omega 3 propeller would be commanded to speed up while the omega 1 propeller would be commanded to slow down. As previously discussed, this reaction is expected in both controllers as they react identically in roll and pitch.

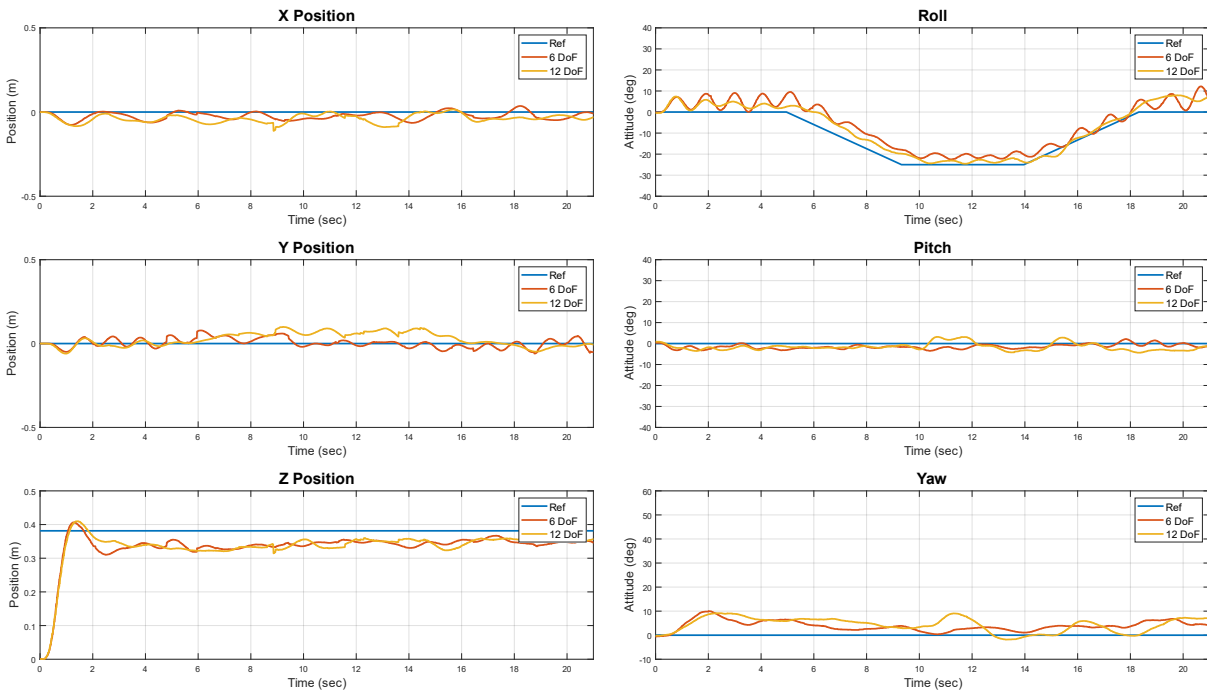


Figure 6.9: Tracking results of both controllers while rolling.

The results of the roll actuation command at 25 degrees can be seen in Figure 6.9. It should be noted that 30 degrees was also attempted with some success, but this would bring the blades and the drone body way too close to the safety enclosure. So, although 30 degrees are attainable, 25 degrees is selected since it was a safer option. The two systems perform

similarly, as expected, with the 6-input controller outperforming the 12-input controller slightly in the unactuated degrees of freedom but showing more oscillation in roll. This difference is still quite small, and as previously stated, is probably due to how repeatable the experiment is rather than how well one controller performs compared to the other. The repeatability issues will be more thoroughly addressed in a following section. As seen before in the yaw command, a slew rate is added to the roll command to prevent sudden, large changes in the error signal. The slew rate here is smaller than the one used in yaw however, at only 0.1 radians per second, or roughly 5.7 degrees per second. The reason the slew rate here is smaller is because the roll and pitch degrees of freedom were found to be less stable in simulation and so this value was selected because of this. However, as is the case with the yaw slew rate, the limit of how large this number can be made has not been tested.

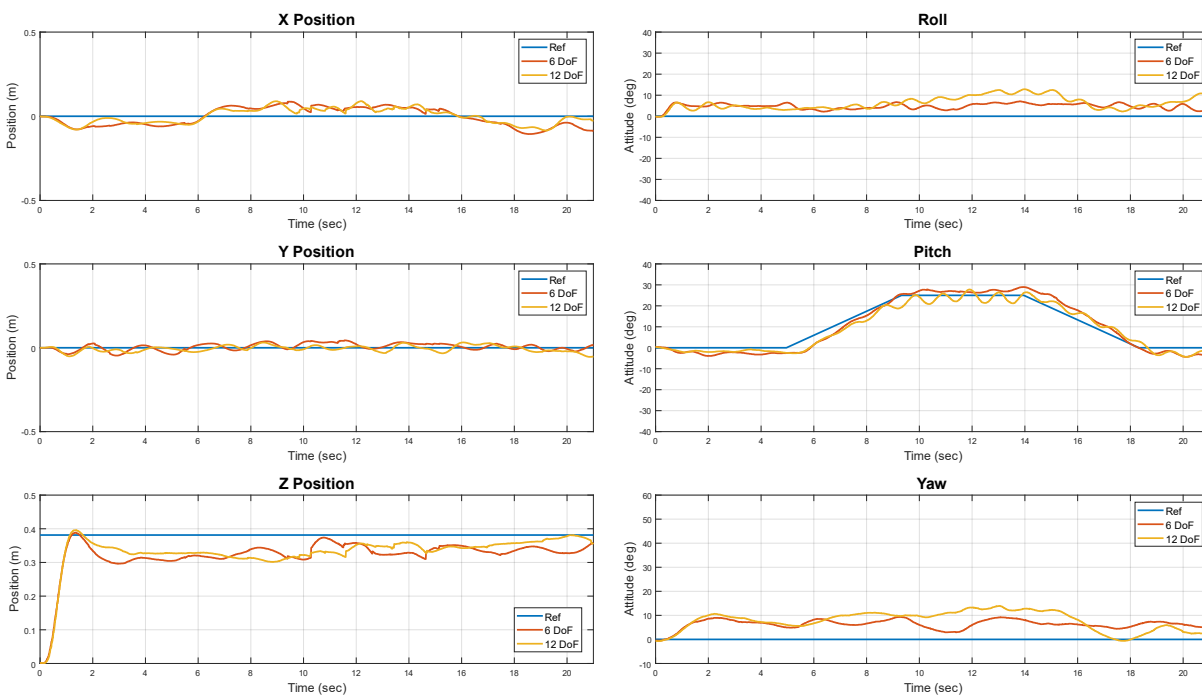


Figure 6.10: Tracking results of both controllers while pitching.

The results of the pitch actuations can be seen in Figure 6.10. The same slew rate applied roll is also applied to pitch, as they are technically identical. Notice that, this time, it is

the 12-input controller that oscillates in the commanded degree of freedom, and the 6-input controller does not. Once again, the difference in performance between the two controllers is not very significant. One thing to notice in both plots and in both controllers is that the system, although commanded to stay at a yaw position of 0, has a predisposition to yaw in the positive direction on takeoff. This was not noticed when the copter was commanded with a simple hover but is probably due to the set of omega 1 and omega 3 motors being slower on average than omega 2 and 4 when given the same PWM commands. Since the Z position command is now made larger, the effect becomes more evident. However, it is worth noting that the system does slowly correct for this over time due to integral action in the controller. The reason why the Z position command is increased almost twofold when compared with the yaw command is because the roll and pitch commands bring the propeller blades very close to the bottom platform. A large Z position command is thus necessary to ensure that no contact is made.

6.4.4 Roll, Pitch, and Yaw Simultaneously

In the literature, the reference command most commonly chosen to showcase the system's ability to attain 6 degrees of freedom is a pure pitch [RBG14, BD16, Elk17]. This is chosen as it is one of the 2 degrees of freedom that a traditional quadcopter cannot attain. And so, by performing a pure pitch, they should the system is able to perform one of degrees of freedom that was previously unattainable. By symmetry, if the pitch can be achieved, then so can the roll. And so, this covers both degrees of freedom that are traditionally unattainable. However, this does not show well the system performs in all degrees of motion when multiple commands in different degrees of motion are made all at once. So, in this section an attempt will be made to hover the copter and then yaw, pitch, and roll in succession. This has not been attempted in the literature except with relatively very small actuations [KKR15].

The actuation planned is basically a superposition of all of actuations shown up to this point. First, the copter is yawed to 45 degrees and given a few seconds to stabilize. It is then commanded to pitch at 25 degrees, followed by a roll of 25 degrees once the system

has stabilized. The system stays at this orientation for a few seconds and then is rolled and pitched back respectively back to its hover orientation. The results of this actuation are shown in Figure 6.11.

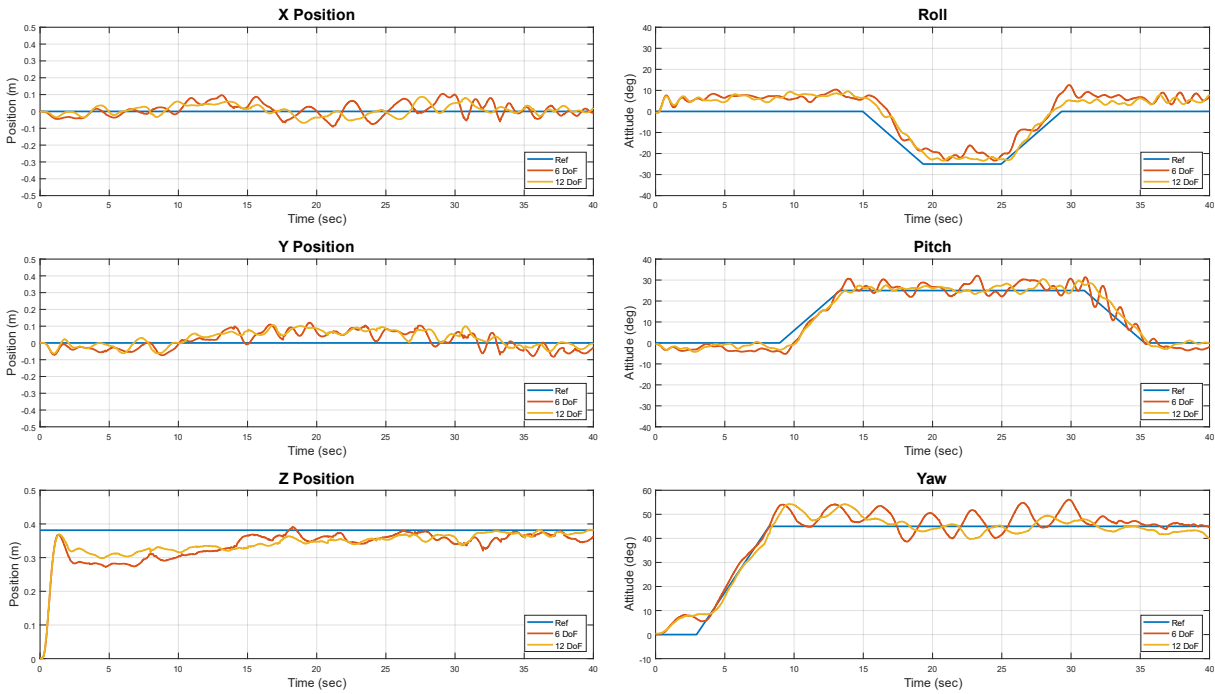


Figure 6.11: Tracking results of both controllers with roll, pitch, and yaw.

The data shows that the system is still stable and can complete the actuation command. However, it becomes immediately apparent that superimposing the attitude commands in this fashion increases the magnitude and frequency of oscillation in all degrees of freedom. One of the degrees of freedom most affected by this is yaw, which would only oscillate in a range of 5 degrees above and below the reference when commanded separately but now oscillates closer to a range of 10 degrees above and below the reference command. The X and Y position commands go from an oscillation range of positive to negative 0.05m to a range closer to positive to negative 0.1m. Another coupling that is apparent here is when pitch is commanded, the Y position experiences a shift in mean. When the roll is commanded, the X position experiences a shift in mean. Both return to normal after the roll and pitch are

restored to their original orientation. The range of oscillation in roll and pitch also increased from only 3 to 4 degrees in its most stable runs to over 10 degrees when all three actuations are superimposed.

The two controllers perform similarly in most degrees of freedom apart from yaw. In yaw, the 12-input controller does a better job of dampening out the oscillation. Overall, although both systems are stable, the amount of oscillatory noise has almost doubled as a result of superimposing the attitude commands. This is probably why this is not done in the literature. However, if the system were truly decoupled, this would not be the case, and the superimposing of commands would make no difference in the error. This suggests that a more sophisticated controller is needed here, as the current one assumes that each degree of freedom is decoupled from the rest. The results suggest otherwise, and if the coupling were accounted for in the controller's design, then it is likely that these results would also improve.

6.4.5 X Y Disturbance Force

Up to this point, actuations have been made in 4 of the 6 possible degrees of freedom. No actuations have been made in the X and Y positions due to the restriction imposed by the safety enclosure. However, this means that the system's ability to track these degrees of freedom is untested, except for their ability to regulate around their origin. So, to paint a more complete picture of the system's capabilities, the copter will be tested with a disturbance force in the X and Y directions while hovering. The idea is, if the system can withstand a force in these two directions and rejects the disturbance while hovering, then it can supply a force equivalent to the disturbance force in these directions. This thus proves, perhaps indirectly, that the system would be able to actuate in these degrees of freedom if allowed to do so.

The simple experimental setup of how this is achieved is shown in Figure 6.12. A string is attached to the system while attempting to keep it perpendicular to the connector arm to



Figure 6.12: Illustration of pulley experiment to apply force in X and Y directions.

ensure equal amounts of force are felt in the X and Y direction. The string is then extended to a pulley where a mass would be dropped down to exert the force onto the system. In the experiment shown, this mass weighed 0.5kg. This experiment was also conducted with a 1kg mass, which the system was able to withstand. However, the limits were hit in X and Y before the system could regulate itself, and so it was considered a failed run even though the system showed an ability to reject the disturbance force. A more conservative weight of 0.5kg is thus used. The results of the experiment are shown in Figure 6.13.

Initially, the copter is allowed to hover and stabilize itself. Then, a mass is lowered to exert the force onto the system. This is done by slowly achieving tension in the line, and then dropping the mass from there, but this can be tricky to do as the drone is moving at the same time. It is difficult to determine exactly when the mass was dropped as this was done manually, which also means that the time at which the mass is released is not the same for both controllers in the plot. But the effects of the force on both systems start somewhere between the 12 and 17 second mark, as highlighted. When the force is applied both X and Y positions start to shift due to the force exerted by the mass. However, both controllers

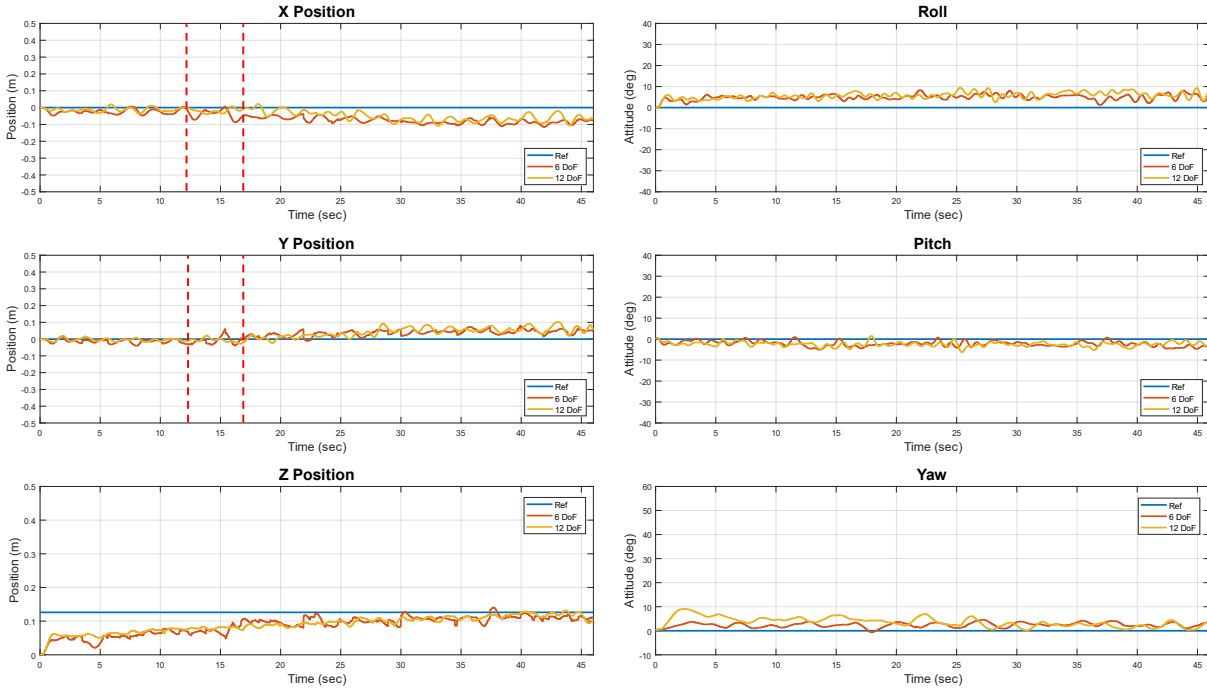


Figure 6.13: Tracking results of both controllers with X Y force disturbance.

can withstand this force and they do so without hitting the X and Y limits imposed by the safety enclosure. It is also possible to see that the system slowly compensates for the bias and moves closer to the X and Y origin, but the experiment was not run long enough for this to complete. Nonetheless, this shows that the system can indeed supply forces in X and Y that would actuate the system in these degrees of freedom. When added to the experiments showing the system’s ability to attain the other 4 degrees of freedom, the copter’s full range of capabilities in 6 degrees of freedom are shown.

6.5 Simulation vs Experimental Results

Before moving on, a comparison of how well the old and new simulation models predict the physical system’s results is made. To do this, only the 6-input controller will be considered as the outcome is similar in both controllers. The high level controller applied will be identical in all 3 cases. The 3 attitude, superimposed reference command will be used for

the comparison as it is the most complex case presented and will help discern differences between the two simulations and the experimental results. The results are shown in Figure 6.14.

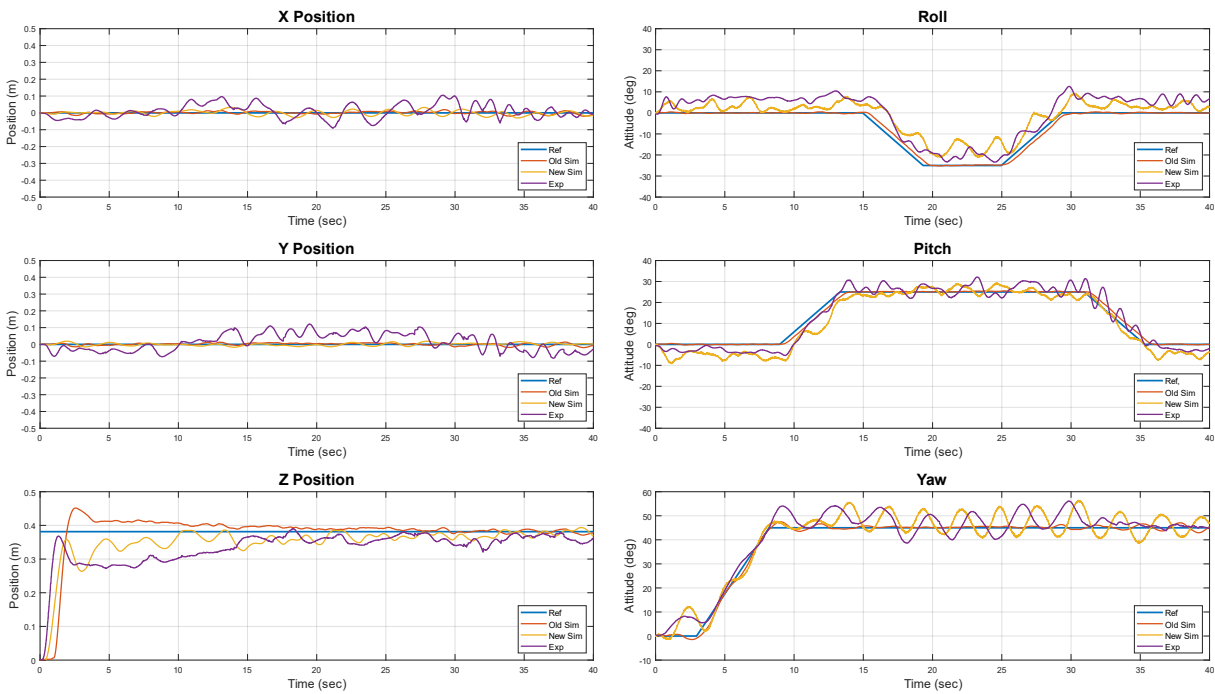


Figure 6.14: Tracking results of old and new simulation with hardware for 6-input controller.

The old simulation is hard to see on the plot in most degrees of freedom. This is because it sits right on top of the reference signal most of the time. It basically predicts that the controller is extremely stable and has little to no oscillation most of the time. There is a slight phase shift in roll and pitch tracking, and a little bit of oscillation in yaw tracking when the reference is changed. However, aside from these small deviations, the simulation predicts that the controller is practically perfect. This is clearly not the case when compared with the experimental data.

When the new simulation's data is superimposed on top of the old simulation, it is clear that the new simulation predicts that the controller will not be as stable as the old simulation predicts. Oscillation with sizable magnitude is seen throughout the response,

especially in the roll, pitch, and yaw responses. The new simulation does a good job here, as the oscillation is seen in the experimental data and the magnitude matches well. However, the new simulation model falls short when predicting the magnitude of oscillations in the X and Y position responses. Although it does predict oscillation, the magnitude captured by the new simulation is roughly one third of what is seen in the experimental data. Overall, the new simulation brings us much closer to the actual response of the system, although some improvement is still possible here.

A final note can be made here about the two controllers' performance in general over all the preceding experiments shown. Neither controller showed significant improvements over the other when comparing the copter body's tracking results. In some experiments, the 12-input controller outperformed the 6-input controller, and in other experiments the 6-input controller outperformed the 12-input controller. However, in all the experiments, the difference in results was too small to determine which controller produced better results. During testing and debugging, the 6-input controller was primarily used because it is easier to understand what the controller is trying to do due to there being less system outputs. It was thus used for most of the testing and debugging phase of the project, but this is not because its performance was better in any way. To determine with certainty which controller is more stable and has better tracking ability, the controllers must each be optimized separately as the dynamics by which each controller yaws is different. Only once both controllers are optimized can one controller be claimed to be better than the other. At this time, both perform very similarly and there is no clear distinction between the tracking of the two systems when considering all the experiments that were conducted.

6.6 Energy Consumption

The tracking performance and methods with which the tracking is achieved for the 12-input and 6-input systems have been discussed, but it is also important to discuss the difference in energy consumption between the two systems over the previously discussed 40 second simulation run. To perform the calculation, it will be assumed that the consumption of

the controller, sensors, and actuation motors other than the propellers is negligible. This assumption is reasonable, because the propellers take up more than 95% of the energy consumed by the overall system. Numerous experiments were conducted on the hardware to determine how much current is consumed at different propeller speeds using a current sensor. Since the propeller speeds are always known, the amount of charge that is being pulled by each propeller can be calculated. Charge is a very relevant measure of efficiency here as this will ultimately determine the maximum flight time of the system given a certain battery. In Table 6.1, the charge consumption is calculated for each arm over the 40 second run and then summed up to determine the system's total consumption.

	12-Input System	6-Input System
Omega 1	575.12 A·s	593.17 A·s
Omega 2	574.66 A·s	566.99 A·s
Omega 3	583.12 A·s	615.55 A·s
Omega 4	582.92 A·s	588.30 A·s
Total	2316.28 A·s	2364.01 A·s

Table 6.1: Charge consumption of 12-Input System vs 6-Input System.

The results show that the 12-input controller is indeed more energy efficient than the 6-input controller, but only by roughly 2% - a negligible difference. Because the 12-input controller minimizes the summed squares of the 4 propeller speeds, this solution minimizes the system's kinetic energy. The 12-input controller is thus the one that finds the solution with maximum efficiency. However, the 6-input controller performs only 2% worse than this when fed the same reference signal, and if the 12-input controller is assumed to give the most optimal solution, then the 6-input controller is also extremely efficient. Another trend that is visible in Table 6.1, is the spread in energy consumption among the arms in the 6-input system versus the 12-input system. The 12-input system's arm consumptions are

much more closely grouped with a spread of roughly 9 A·s, while the 6-input system has a spread of roughly 48 A·s from its lowest to highest consuming arm. This is expected, as the 6-input system yaws by varying the speeds of opposite spin propellers, while the 12-input system yaws by actuating the alpha joints and getting a portion of thrust in the desired yaw's direction while keeping the propeller speeds relatively steady.

6.7 Experimental Repeatability Issues

In some cases, the experiment was rerun multiple times to obtain a successful run. In these cases, it was clear that there were certain aspects of the experiment that were not very repeatable, leading to some runs being successful and others failing even though the reference command and initial conditions were kept constant between these runs. In this section, some of the repeatability issues will be addressed and possible fixes will be proposed.

As previously mentioned, maintaining line of sight between the beacons and the CrazyFlies is vital to maintaining the position and attitude feedback of the drone body. When this signal is lost due to blockage of line of sight, the copter quickly becomes unstable. This becomes a challenge for two reasons. First, the rod in the middle of the safety enclosure can block the line of sight as the copter is being actuated. And so, the initial condition of the copter and the path it moves along as the actuation is completed must be carefully considered so that no blockage occurs due to the safety enclosure's rod. Second, when the copter starts off at a level hover and then rolls and pitches, it is likely that the copter's own body will block the sensors. It is very difficult to position the beacons in positions where the CrazyFlie will be seen throughout the motion, and the experiment must thus be redone multiple times in some cases. This is quite different from the simulation, where the position readings are always available, and no blockage needs to be considered. To tackle this issue, 4 to 8 Lighthouse beacons can be used instead of just 2 to increase the visibility of the markers from various angles.

Another issue with respect to the CrazyFlies' sensor measurements is where the sensors

are mounted physically within the system. Ideally, these sensors would be placed at the copter's center of mass, as this is the point of interest where the measurements are needed. However, this is clearly not possible, as the center of the copter was kept open intentionally for possible tools that may be mounted onto the frame. Therefore, the 2 CrazyFlies were placed on the connector plates on either side of the copter. The position and attitude of the copter at the center of mass is then calculated via a transformation matrix that maps the displacements and rotations at the sensor's mounted location to the body's center of mass. Of course, in doing this, it is assumed that the body is completely rigid, and so the transformation is valid. However, in practice, it was found that this is not the case, as the body was quite flexible at its joints. In the first few trials experiments, this would lead to the sensor picking up rotations that only happen at the extremities where the sensors are mounted but not at the center of mass. An illustration of this is shown in Figure 6.15. In the figure, the green dashed line shows an exaggeration of what the body would look like when it bends. In such a case, the body's center of mass is still flat (depicted in black), but because the sensors are not mounted at the center of mass, the sensors pick up a false rotation (depicted in red). To fix this, two things were done. First, a second sensor was mounted on the exact opposite side from the first so that an average of the two would more accurately convey the position and rotation of the center of mass. Second, the body was made stiffer by adding numerous plates to the design. It is also worth mentioning that if the sensors were not mounted perfectly flush to the copter's body, then there would always be an error in the drone's actual roll and pitch when compared with the sensor readings. This means that the drone will attempt to track the sensor's roll and pitch, and not its own body's. This, of course, will lead to undesirable outcomes such as translations in the X and Y directions when the copter should just be hovering. To solve this issue, adapters are machined and used to mount the CrazyFlies more precisely onto the copter's body.

Powering the copter for long periods of time was another challenge that was encountered, as it consumes a large amount of charge while running. If this were to be done with the

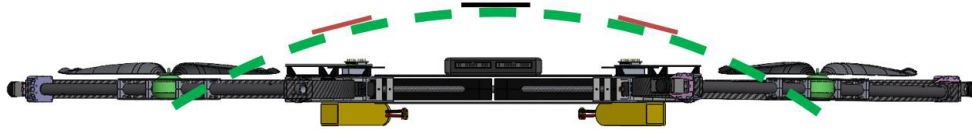


Figure 6.15: Illustration of how the body bends causing the sensors to rotate.

onboard batteries that were planned for the system, the batteries would need to be recharged every 3-4 runs. This would be fine for normal operation; however, it is not sufficient for the testing phase where the copter is running consistently for long periods of time. So, instead of using the onboard batteries, the system was tethered to four power lines that come from a pair of 12V car batteries connected in series. This setup works well, as it allows the system to be run for 70-80 runs before the batteries need to be recharged. However, caution must be exercised with the 4 power cables that are extending down from the system as they could get caught in the propellers. Also, the starting position of these cables must be constant from run to run to ensure that the cables are not taut or interfering with the copter's movement in any way.

However, it must be noted that the propeller's PWM signal to propeller speed mapping was done with a battery that was freshly charged. Therefore, as the batteries are depleted by consecutive runs, the mapping no longer matches reality, which thus leads to deviations in the system's performance depending on how well charged the battery is. The charge level of the battery thus introduces another factor when considering the repeatability of the experiment. The effects of low charge can be mitigated in the Z position by increasing the mass of the drone in the controller which would give more nominal thrust in this direction as the controller now compensates for mass and increases the nominal PWM signal input needed to carry the system. However, this is not a very good solution, as it will only compensate for the thrust lost due to the low battery voltage in the Z direction but will do nothing to compensate for the changes that the system experiences in any of the other degrees of freedom - a better solution is thus needed. This issue can be resolved by eventually moving away from the current battery setup. It would be best to start using multiple sets of the onboard batteries which would ideally always be freshly charged by alternating sets as used

sets charged. This gets rid of the cabling issue and allows for a true free flying drone, and it may reduce the variation in charge from run to run as the range of voltage from a fully charged to depleted battery is reduced.

Another source of poor repeatability is how the alpha and beta motors' initial 0 position was established manually at the beginning of each experiment. In the case of the alpha joint, the arm is pushed all the way down so that it sits on a physical limit and then it is commanded to move a certain number of encoder counts to its 0 position. However, it is possible that the alpha arm is not sitting perfectly on the limit every time, or that it moves between the initial setup and when the code starts to run. In this case, the alpha's initial 0 position may slightly shift from one run to the next. It should be noted that alpha has a homing sensor, but this is not used as the number of digital pins on the controller are maxed out, and so this sensor is forfeited in favor of more vital sensors which the system cannot run without. In the beta joint, there is no physical limit, and so the zeroing is done purely by eye. The arm is lined up against a flat edge that sits behind it, and this determines how straight the arm is. However, since it is done by eye, there are bound to be slight deviations in each arm from one run to the next. To solve this issue, absolute encoders should be used instead of normal ones in both the alpha and beta motors. The issue is, these will introduce the need for more digital pins on the controller, which may mean that the controller would have to be replaced with a different one altogether. Instead, an analog absolute encoder could be introduced to eliminate the need to change out the controller due to the unavailability of digital pins.

Finally, as the arms are moving about, the wires need to cross the revolute joints, and this always poses a threat of tangling or snagging as the joints move. To avoid this, ample slack was given in these wires, but this issue was still observed periodically as the system was operated, and in such cases the experiment's repeatability was affected. These wires would also hinder the system from being able to make large angle attitude actuations in the future, as the joints would have to make very large rotations in such a case. This would require more slack in the wires which would increase the risk of tangling. A possible solution

to this problem is slip rings. By using a slip ring at each rotary joint, the system would be able to rotate freely in these joints with no risk of tangling or obstruction. However, due to the large amount of current that needs to pass from the drone's body to the omega motor, such a slip ring would have to be custom made for this system and would thus be expensive.

CHAPTER 7

Conclusion

This work developed and tested a quadcopter with 12 degrees of freedom that was previously introduced. A better simulation of the quadcopter that more accurately represents the hardware system was created via more rigorous system modeling and identification. The simulation now considers the system to have distributed mass, varying moments of inertia, and motor dynamics. The improved simulation provides a more accurate test bench for control design. Furthermore, a new control scheme with 6 system inputs and 6 system outputs was introduced and compared to the existing control scheme with 12 system inputs and 6 system outputs. Both control schemes were proven to be stable with testing in simulation and experiment, and both showed similar results when considering the tracking response of the copter's body. Overall, the copter demonstrates the ability to achieve all 6 degrees of freedom with results similar to those found in the literature.

Possible avenues of future work include improving the controller to account for the coupling observed in the system, with the goal of reducing the oscillation seen in the results. Furthermore, the safety enclosure that is currently in use, although great in initial testing, restricts the system from making large magnitude rotations and translations. To better demonstrate the full range of the copter's capabilities, a new experimental setup needs to be made to allow the copter to fly freely in space. Finally, testing the copter with a tool or payload mounted onto the frame is needed to experimentally assess its ability to achieve 6 degrees of freedom while carrying a payload and performing a task where accurate position and attitude tracking are required.

BIBLIOGRAPHY

- [Bau02] Robert J Bauer. “Kinematics and dynamics of a double-gimbaled control moment gyroscope.” *Mechanism and machine theory*, **37**(12):1513–1529, 2002.
- [BD16] Dario Brescianini and Raffaello D’Andrea. “Design, modeling and control of an omni-directional aerial vehicle.” In *2016 IEEE international conference on robotics and automation (ICRA)*, pp. 3261–3266. IEEE, 2016.
- [Bit14] Bitcraze. “Crazyflie 2.1: A versatile open source flying development platform.”, 2014.
- [Bit19] BitCraze. “Getting started with the Lighthouse System: a positioning system which can estimate its own X, Y and Z in a global coordinate system.”, 2019.
- [BM99] Francesco Bullo and Richard M Murray. “Tracking for fully actuated mechanical systems: a geometric framework.” *Automatica*, **35**(1):17–34, 1999.
- [BTK20] Karen Bodie, Zachary Taylor, Mina Kamel, and Roland Siegwart. “Towards efficient full pose omnidirectionality with overactuated mavs.” In *International Symposium on Experimental Robotics*, pp. 85–95. Springer, 2020.
- [CFC14] Anežka Chovancová, Tomáš Fico, L’uboš Chovanec, and Peter Hubinsk. “Mathematical modelling and parameter identification of quadrotor (a survey).” *Procedia Engineering*, **96**:172–181, 2014.
- [CLM11] Bill Crowther, Alexander Lanzon, Martin Maya-Gonzalez, and David Langkamp. “Kinematic analysis and control design for a nonplanar multicopter vehicle.” *Journal of Guidance, Control, and Dynamics*, **34**(4):1157–1171, 2011.
- [Elk17] Omar Elkhatib. “Control allocation of a tilting rotor hexacopter.”. B.S. thesis, ETH Zurich, 2017.

- [ESG08] Juan Escareno, Anand Sanchez, O Garcia, and Rogelio Lozano. “Triple tilting rotor mini-UAV: Modeling and embedded control of the attitude.” In *2008 American Control Conference*, pp. 3476–3481. IEEE, 2008.
- [GG17] Adriano Garcia and Kanad Ghose. “Autonomous indoor navigation of a stock quadcopter with off-board control.” In *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, pp. 132–137. IEEE, 2017.
- [GT18] Matthew J Gerber and Tsu-Chin Tsao. “Twisting and tilting rotors for high-efficiency, thrust-vectoring quadrotors.” *Journal of Mechanisms and Robotics*, **10**(6):061013, 2018.
- [HCZ18] Bangcheng Han, Yulin Chen, Shiqiang Zheng, Mingxing Li, and Yangyang Shi. “Robust control for a magnetically suspended control moment gyro with strong gyroscopic effects.” In *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, pp. 2440–2446. IEEE, 2018.
- [HHM15] Minh-Duc Hua, Tarek Hamel, Pascal Morin, and Claude Samson. “Control of VTOL vehicles with thrust-tilting augmentation.” *Automatica*, **52**:1–7, 2015.
- [IL17] Davide Invernizzi and Marco Lovera. “Geometric tracking control of thrust vectoring UAVs.” *arXiv preprint arXiv:1703.06443*, 2017.
- [JV14] Guangying Jiang and Richard Voyles. “A nonparallel hexrotor UAV with faster response to disturbances for precision position keeping.” In *2014 IEEE International Symposium on Safety, Security, and Rescue Robotics (2014)*, pp. 1–5. IEEE, 2014.
- [KKR15] D Kastelan, M Konz, and J Rudolph. “Fully actuated tricopter with pilot-supporting control.” *IFAC-PapersOnLine*, **48**(9):79–84, 2015.
- [KR13] Matthias Konz and Joachim Rudolph. “Quadrotor tracking control based on a moving frame.” *IFAC Proceedings Volumes*, **46**(23):80–85, 2013.

- [KVE18] Mina Kamel, Sebastian Verling, Omar Elkhatib, Christian Sprecher, Paula Wulkop, Zachary Taylor, Roland Siegwart, and Igor Gilitschenski. “The voliro omniorientational hexacopter: An agile and maneuverable tilttable-rotor aerial vehicle.” *IEEE Robotics & Automation Magazine*, **25**(4):34–44, 2018.
- [MC17] C Molter and P Cheng. “Propeller performance calculation for multicopter aircraft at forward flight conditions and validation with wind tunnel measurements.” In *Proceedings of the International Micro Air Vehicle Conference and Flight Competition (IMAV), Stuttgart, Germany*, pp. 307–315, 2017.
- [ML12] Mohamed Kara Mohamed and Alexander Lanzon. “Design and control of novel tri-rotor UAV.” In *Proceedings of 2012 UKACC International Conference on Control*, pp. 304–309. IEEE, 2012.
- [NDT19] Tritex NDT. “Multigauge 6000 drone thickness gauge: the first dedicated drone thickness gauge available worldwide.”, 2019.
- [NGK15] Alexandros Nikou, Georgios C Gavridis, and Kostas J Kyriakopoulos. “Mechanical design, modelling and control of a novel aerial manipulator.” In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4698–4703. IEEE, 2015.
- [NK14] Alireza Nemati and Manish Kumar. “Modeling and control of a single axis tilting quadcopter.” In *2014 American Control Conference*, pp. 3077–3082. IEEE, 2014.
- [NPP18] Hai-Nguyen Nguyen, Sangyul Park, Junyoung Park, and Dongjun Lee. “A Novel Robotic Platform for Aerial Manipulation Using Quadrotors as Rotating Thrust Generators.” *IEEE Transactions on Robotics*, **34**(2):353–369, 2018.
- [OAN15] Atsushi Oosedo, Satoko Abiko, Shota Narasaki, Atsushi Kuno, Atsushi Konno, and Masaru Uchiyama. “Flight control systems of a quad tilt rotor unmanned aerial vehicle for a large attitude change.” In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2326–2331. IEEE, 2015.

- [PLA18] Sangyul Park, Jeongseob Lee, Joonmo Ahn, Myungsin Kim, Jongbeom Her, Gi-Hun Yang, and Dongjun Lee. “Odar: Aerial manipulation platform enabling omnidirectional wrench generation.” *IEEE/ASME Transactions on mechatronics*, **23**(4):1907–1918, 2018.
- [PRY 9] Chen-Huan Pi, Lecheng Ruan, Pengkang Yu, Yao Su, Stone Cheng, and Tsu-Chin Tsao. “A Simple Six Degree-of-Freedom Aerial Vehicle Built on Quadcopters.”, 2021-8-9.
- [PRY21] Chen-Huan Pi, Lecheng Ruan, Pengkang Yu, Yao Su, Stone Cheng, and Tsu-Chin Tsao. “A simple six degree-of-freedom aerial vehicle built on quadcopters.” In *2021 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 329–334. IEEE, 2021.
- [RBG14] Markus Ryll, Heinrich H Bühlhoff, and Paolo Robuffo Giordano. “A novel overactuated quadrotor unmanned aerial vehicle: Modeling, control, and experimental validation.” *IEEE Transactions on Control Systems Technology*, **23**(2):540–556, 2014.
- [RCS19] Ramy Rashad, Federico Califano, and Stefano Stramigioli. “Port-hamiltonian passivity-based control on se (3) of a fully actuated uav for aerial physical interaction near-hovering.” *IEEE Robotics and automation letters*, **4**(4):4378–4385, 2019.
- [RLO18] Fabio Ruggiero, Vincenzo Lippiello, and Anibal Ollero. “Aerial manipulation: A literature review.” *IEEE Robotics and Automation Letters*, **3**(3):1957–1964, 2018.
- [RMP17] Markus Ryll, Giuseppe Muscio, Francesco Pierri, Elisabetta Cataldi, Gianluca Antonelli, Fabrizio Caccavale, and Antonio Franchi. “6D physical interaction with a fully actuated aerial robot.” In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5190–5195. IEEE, 2017.

- [RRB15] Sujit Rajappa, Markus Ryll, Heinrich H Bühlhoff, and Antonio Franchi. “Modeling, control and design optimization for a fully-actuated hexarotor aerial vehicle with tilted propellers.” In *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 4006–4013. IEEE, 2015.
- [SYG10] Yao Su, Pengkang Yu, Matthew J Gerber, Lecheng Ruan, and Tsu-Chin Tsao. “Nullspace-Based Control Allocation of Overactuated UAV Platforms.” *IEEE Robotics and Automation Letters*, **6**(4), 2021-10.
- [TYJ10] Naoki Takatsuka, Katsuhiko Yamada, and Ichiro Jikuya. “Spacecraft attitude control using a double-gimbal control moment gyro.” *Transactions of the Japan Society for Aeronautical and Space Sciences, Aerospace Technology Japan*, **8**(ists27):Pd_45–Pd_52, 2010.
- [YSG21] Pengkang Yu, Yao Su, Matthew J Gerber, Lecheng Ruan, and Tsu-Chin Tsao. “An over-actuated multi-rotor aerial vehicle with unconstrained attitude angles and high thrust efficiencies.” *IEEE Robotics and Automation Letters*, **6**(4):6828–6835, 2021.