ROBOBAT: DYNAMICS AND CONTROL OF FLAPPING FLIGHT MICRO AERIAL VEHICLES

BY

PATRICK DANIEL KUANG

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Aerospace Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Adviser:

Assistant Professor Soon-Jo Chung

Abstract

Flapping flight micro aerial vehicles (MAVs) are of interest to the aerospace and robotics communities for their maneuverability in comparison to tradition fixed wing and rotary aircraft. However they present numerous challenges in the fields of dynamics, stability and control. This thesis examines the dynamics and kinematics of robotic flapping flight, the design and construction of a robotic bat test bed mounted on a 3-DOF pendulum, and subsequent control experiments using the test bed. The robotic bat test bed is capable of exhibiting different wing motions and is used to test the feasibility of controlling the motions of the robotic bat by using the phase differences between coupled nonlinear oscillators called central pattern generators (CPGs). A dynamic model for the robotic bat based on the complex wing kinematics is presented, and the wing kinematic motions themselves are analyzed using a high-speed motion capture system. Mechanical coupling effects which deviate from theoretical assumptions are investigated as well. Open loop experiments analyzing the steady state behavior of the bat's flight with varying phase differences showed a change of the pitch angle while elevation and forward velocity remains constant. Closed loop experiments indeed validate that control dimension reduction is achievable by controlling the phase differences of CPG oscillators. Unstable longitudinal modes are stabilized and controlled using only control of two parameters: phase difference and flapping frequency. Transition between flapping flight and gliding flight is analyzed. This shows promising results regarding the relation between phase differences of wing motions and longitudinal stability, and lays the groundwork for future research and experimentation in flapping flight MAVs.

To my parents, for their love, support, and wisdom.

Acknowledgments

The last two years of my life have been quite an adventure to say the least. My time as a graduate student would not have been successful without the help and support of many people I have met. I owe much graditude to my advisor, Professor Soon-Jo Chung, for helping me grow and develop as a researcher, engineer, and person and giving me the opportunity to learn and work in a dynamic environment. The project was supported by the Air Force Office of Scientific Research (AFOSR) under the Young Investigator Award Program (Grant No. FA95500910089) monitored by Dr. W. Larkin. I also am deeply grateful for my fellow labmates and colleagues, who have provided indispensable help and support during my time here. There are many great fellow students, current and former, whom I have had the good fortune of knowing and working with thanks to this research, but in particular I would like to thank PhD students Michael Dorothy and Aditya Paranjape for their help in countless different topics for my work and thesis. Several undergraduate students have also helped make this research possible including Harry Rocha, Ryan Cook, James Holtman, Andrew Meister, Matt Schipp, and Danny Park. Most of all, I would like to thank my family for supporting me in graduate school; none of this would have ever been possible without their unconditional love.

Table of Contents

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

Nomenclature . . . . . **ix**

Chapter 1 Introduction **1**
1.1 Motivation and Related Work 2
1.2 Thesis Organization 5

Chapter 2 A Review of Flapping Flight MAVs **6**
2.1 Introduction 6
2.2 Biological Inspiration 6
2.2.1 Flapping Flight in Nature 6
2.2.2 Central Pattern Generators 8
2.3 State of the Art of Flapping Flight MAVs 11

Chapter 3 Flapping Flight Kinematics and Dynamics **12**
3.1 Introduction 12
3.2 Testbed Kinematics and Unsteady Aerodynamics 12

Chapter 4 Description of the Robotic Bat Test Bed **23**
4.1 Introduction 23
4.2 Design of Robotic Bat 23
4.2.1 Previous Work 23
4.2.2 Current Robobat Design 23
4.3 Design of 3DOF Test Bed 30
4.4 Controller Design 32
4.5 VICON Motion Capture System 37

Chapter 5 Experimental Setup and Results with Robotic Bat **41**
5.1 Introduction 41
5.2 Open Loop Control Experiments 41
5.3 Closed Loop Control Experiments 45
5.4 Glide Transition 53
5.5 Force and Moment Calculations 58
5.6 Measurements of Mechanical Coupling 63

Chapter 6 Supplemental Implementations 73
6.1 Introduction 73
6.2 Implementation of CPGs on FPGAs 73

Chapter 7 Conclusions and Future Work 78
7.1 Conclusion 78
7.2 Future Work 79

References 83

List of Figures

3.1 RoboBat Testbed 12

4.1 Previous robotic bat design[1] 24
4.2 Front and back views of robotic bat, mounted on Quanser 3DOF Helicopter stand 25
4.3 Positive wing motion directions 26
4.4 Wing of robotic bat 27
4.5 Drive shaft to actuate flapping, with motor and encoder shown 27
4.6 Shoulder joint, with pushrods for flapping, pitch, and lead-lag 28
4.7 Pitch (top) and lead-lag (bottom) servos 28
4.8 Close-up of amplitude controlling servo and drive shaft 29
4.9 dSPACE connector board 30
4.10 Schematic of test bed setup 31
4.11 Components of test bed 33
4.12 View of test bed set up. 34
4.13 Linear actuator functioning as pitch counterweight 35
4.14 Encoders on Quanser 3DOF helicopter stand 36
4.15 Robotic bat fitted with markers 38
4.16 Vicon cameras 38
4.17 Vicon cameras setup on tripods 39
4.18 Vicon cameras around robotic bat 39
4.19 Robotic bat, as seen by Vicon Tracker 40

5.1 Open-loop pitch control via phase differences (2 Hz). 42
5.2 Open-loop pitch control via phase differences (2.5 Hz). 43
5.3 Open-loop pitch control via phase differences (3 Hz). 44
5.4 Experimental Results of Pitch Control at 2.5 Hz 47
5.5 Experimental Results of Pitch Stability by Control at 3Hz 48
5.6 Experimental Results of Pitch Control at 3Hz 48
5.7 Velocity Control Tracking 49
5.8 Pitch Stability after Large Perturbation 49
5.9 Pitch control with corresponding control input 50
5.10 Pitch control tracking with corresponding control input 51
5.11 Pitch control tracking with corresponding control input 52
5.12 Pitch control response to large perturbation 53

5.13 Velocity control tracking with corresponding control input 54
5.14 Velocity control tracking with corresponding control input 55
5.15 Pitch tracking with active velocity control 56
5.16 Change in desired pitch results in velocity tracking 57
5.17 Change in desired velocity results in pitch tracking 58
5.18 Flapping-Glide Transition 59
5.19 Flapping-Glide Transition, using flapping angle 60
5.20 Flight path angle of flapping-glide transition 61
5.21 Flapping-Glide-Flapping Transition 62
5.22 Numerically calculated lift, thrust, and moment 64
5.23 Experimental Left wing rotations with 10 deg input Δ_{75} 66
5.24 Experimental Right wing rotations with 10 deg input Δ_{31} 66
5.25 Experimental Left wing rotations with 60 deg input Δ_{75} 67
5.26 Experimental Right wing rotations with 60 deg input phase Δ_{31} 67
5.27 Measured vs Commanded phase differences between lead-lag and flapping motions (Δ_{75}) for left wing 68
5.28 Measured vs Commanded phase differences between lead-lag and flapping motions (Δ_{31}) for right wing 69
5.29 Relation between Δ_{75} and Δ_{76} for left wing 70
5.30 Relation between Δ_{31} and Δ_{32} for right wing 70
5.31 Relation between Δ_{75} and Δ_{65} for left wing 71
5.32 Relation between Δ_{31} and Δ_{21} for right wing 71

6.1 Schematic for sine wave PWM signal generation 75

Nomenclature

MAV	Micro Aerial Vehicle
CPG	Central Pattern Generator
DOF	Degree of Freedom
CFD	Computational Fluid Dynamics
FFT	Fast Fourier Transform
FPGA	Field Programmable Gate Array
GUI	Graphical User Interface
SDK	Software Development Kit

$C_L,\ C_D$	coefficients of lift and drag
$C_{m,ac}$	coefficient of pitching moment about the aerodynamic center
$\alpha,\ \beta$	angle of attack, sideslip
$\psi,\ \theta,\ \phi$	Euler angles
γ	flight path angle
ϕ_w, ψ_w, θ_w	Flapping, lead-lag, and pitch angles of each wing (left, right)
$\mathbf{x}_i = (u_i, v_i)^T$	State vector of the i-th Hopf oscillator
$\mathbf{f}(\mathbf{x}_i; \rho_i)$	Hopf nonlinear equations in the vector form with radius ρ_i
ρ_i	Radius of the limit cycle from the i-th Hopf oscillator
λ	Common rate of convergence of Hopf oscillators
ω	Common oscillation frequency of Hopf oscillators, rad/s
a_i	Amplitude bias of the i-th Hopf oscillator

σ	Bifurcation parameter. $\sigma = 1$ for a stable limit cycle or $\sigma = -1$ for convergence to a_i.
$\mathbf{R}(\Delta_{ij})$	2×2 rotational transformation matrix
Δ_{ij}	Phase lead of the i-th Hopf oscillator from the j-th
n	Total number of Hopf oscillators in the CPG network
k	Coupling gain of the coupled Hopf oscillators
	Subscripts
i	Variable number of the coupled Hopf nonlinear oscillators
R, L	Right or left wing

Chapter 1

Introduction

Animals such as birds, bats and insects are capable of agile flight motions and rely mostly on flapping their wings for stability and control. The relatively small size of these animals limits their flight regime to a Reynolds number on the order of $10^4 - 10^5$. Moreover, MAVs fly in low Reynold number regimes similar to that of small birds. MAVs with wings equipped with multiple degrees-of-freedom such as flapping, wing twist, and sweep provide greater maneuverability than conventional fixed-wing aircraft. At such a flight regime, flapping winged aircraft may have advantages over fixed wing aircraft. [2]. Thus a main goal of this project is to emulate and adapt these methods of flight, which have been time-tested over millions of years, to MAVs. These MAVs can be used for intelligence gathering, surveillance, and reconnaissance missions in tightly constrained spaces such as forests and urban areas.

The design of flapping flight micro aerial vehicles presents numerous control and dynamic challenges, as well as challenges in several other engineering fields. Structural materials must also be of consideration for both weight constraints and load capabilities. Actuators must not be too big or too heavy and are constrained by available power sources, but at the same time must be capable of generating sufficient aerodynamic forces for flight and must be of sufficient speed and precision. Avionics such as video cameras, sensors and an onboard computer would form the brain of the aircraft and control the aircraft's actuators and motion, and again would be constrained by weight and power availability. Thus, minimizing the dimensionality of the main controller of a flapping flight micro aerial vehicle would be greatly beneficial. This is achieved using central pattern generators (CPG's), a neurobiologically inspired scheme [1], which are discussed in Chapter 2.

While ornithopters capable of flapping their wings about a single axis have existed for some time, in nature animals flap their wings to yield several complex movements in three dimensions. Animals capable of flight utilize several, different complex three dimensional wing motions at the shoulder joint such as flapping, pitching and a lead-lag motion. These motions must be understood and controlled in phase with each other in order to achieve fully autonomous flapping flight. A test-bed modeled after a bat was designed and built to simulate flapping, pitching and lead-lag motions. The wing motions are controlled by CPGs. The test-bed was originally stationary and mounted on a stand and force torque sensor, meant to be used in a wind tunnel to measure aerodynamic forces. More information on this first model can be found in [3].

1.1 Motivation and Related Work

There is a growing interest in the aerospace and robotics community in the development of MAVs to learn and mimic avian flight. Advances in actuators and control systems have led to development and analysis of articulated and flapping MAVs inspired by animals [4, 5, 6, 1]. Birds achieve remarkable stability and perform agile manuevers using their wings very effectively [7]. One of the goals of reverse-engineering avian and bat flight is to learn more about the various aspects of avian flight such as stability, maneuverability and control from the dynamics of MAV.

From a controls standpoint, we should distinguish between insect scale flight and bird/bat scale flight. Biological insect flight utilizes small musculature to produce passive pitching dynamics over the course of several wingbeats [8]. From a controls standpoint, we can utilize averaging theorems to greatly aid control design around trim states such as hover condition [5, 9, 10]. On the other hand, bat scale flight is low frequency enough that averaging theorems do not directly apply in practice and intra-wingbeat effects are significant [11]. Biologically, bat flight looks more like walking locomotion in other mammals, where many

joints must be coordinated in a rhythmic fashion to produce motion. Bats have anatomical similarities with other mammals. While we are not aware of tests specifically on bats, it makes sense that they would coordinate their joints in a similar fashion to many other mammals: CPGs [12, 13]. A main objective and contribution of this paper is to establish the importance of phase difference control in bird or bat sized flapping MAVs. We also aim to demonstrate possible uses and experimentations using a robotic bat testbed and how this can be used as a stepping stone for future work.

Previous robotic flapping flyers and their control design consider one or two degrees of freedom in the wings [5, 9, 14, 6]. However, even insects like the dragonfly (*Anax parthenope*) are reported to have complex three-dimensional movements by actively controlling flapping and twisting of four independent wings [15]. Furthermore, prior studies in flapping flight [15, 16, 2, 17, 18, 19, 6] assumed a very simple sinusoidal function for each joint to generate flapping oscillations, without deliberating on how multiple limbs (or their nervous systems) are connected and actuated to follow such a time-varying reference trajectory. Other studies consider articulated wings, where steady-states are found for behavior that more resembles gliding [7, 2, 20]

In order to utilize the knowledge gained from CPGs in biological fliers, we have built a robotic bat with dimensionality far lower than the animals. These experiments are an early step toward strict modeling of biological fliers, but are more helpful for design of an artificial flapping flier. We exploit the dimensional reduction made possible by simple CPG rules to make control design and aerodynamic testing feasible. In time, we expect to gain insight into the stability and agility of animal flight, though we hope to encounter engineered solutions that are even more efficient than their biological counterparts.

Moderately large birds and bats often spend their time in either a low-frequency flapping mode or a gliding mode. The proposed CPG based controller can switch smoothly to the gliding mode by changing a bifurcation parameter. The gliding mode is not unlike that explored in the traditional flight mechanics literature. However, fully articulated wings

inherent in flapping flight create additional control possibilities and concerns. A gliding mode may be used for soaring and to shed energy in preparation for a perching maneuver. Perching can be described as a high angle-of-attack pull-up with high lift and a large drag. The large lift and drag forces cause the MAV to climb and lose speed significantly. A planted landing can be achieved in the process.

Birds successfully perch on a variety of structures such as building ledges, power lines, cliff side, and tree branches. Such perching capability in MAVs can significantly reduce the landing distance. However, perching requires the ability to maintain trajectory very accurately. Furthermore, a typical perching maneuver would not last more than a few seconds. Because of its duration and highly unsteady flight profile, perching is an important agility metric for MAVs. The unsteady flight profile makes control design for perching a challenging problem.

The aerodynamics of perching has been explored for conventional, fixed-wing aircraft by Crowther [21] who showed that perching could be performed with essentially a simple pitch-up maneuver and used genetic algorithm to optimize the maneuver. Wickenheiser and Garcia demonstrated perching manuever with controlled wing twist and variable tail incidence [22, 23]. Roberts *et al.* [24] examined the perching problem from controllability aspects. One of the most outstanding experimental demonstrations of a perching maneuver was reported by Cory and Tedrake [25], where they obtained reliable estimates of the open loop dynamics and used them to perform an maneuver optimized to minimize the error in the final position. In contrast with the aforementioned work, [7] considers a completely different mechanism (wing dihedral) to control the flight path angle as well as lateral-directional dynamics during perching. The lateral-directional control, in particular, is often neglected in the literature on robotic perching. More perching work can be found in [26].

1.2 Thesis Organization

We first introduce some basic dynamics of our flapping flight model in Chapter 3. The next goal was to design and engineer a flapping flight test-bed capable of moving in three-dimensions, as opposed to remaining static on a stand. This test-bed can be used to verify previous research regarding the synchronization and control of phase differences between wing motions to achieve stability [1]. A Quanser 3DOF Helicopter is a commercially available platform for control experiments and includes a set of rotors, mounted on a base embedded with encoders in three directions. Using the base of a Quanser 3DOF Helicopter, the rotors of the helicopter were replaced with the robotic bat itself. The experimental hardware is discussed in further detail in Chapter 4. We conducted numerous experiments using the testbed to test control schemes for longitudinal stability and velocity during flapping flight, and also experiments which transition from flapping flight to gliding flight. The experiments and results are detailed in Chapter 5. Finally, we review some further possible implementations of flapping flight control in Chapter 6.

Chapter 2

A Review of Flapping Flight MAVs

2.1 Introduction

This chapter presents a literature review of research regarding biological inspiration from animal flight, CPGs, and other research regarding flapping flight MAVs.

2.2 Biological Inspiration

2.2.1 Flapping Flight in Nature

Flapping flight has historically been inspired by flying creatures found in nature; birds flight formed the basis of inspiration for most early attempts at flight, such as the gliding experiments conducted by Otto Lilienthal. Leonardo da Vinci is thought to have designed a ornithopter (an aircraft that flaps its wings to propel itself into flight) as a method of flight for humans. Therefore, there are several important examples of flapping flight in nature that we derive our design from. Mass, beating frequency, and Reynolds number of flying creatures varies greatly over a spectrum ranging from large birds to the smallest of insects. Power required for steady flight also varies widely. For forward flight, air must be accelerated rearward and downward to generate sufficient thrust and lift to counter the weight and drag sustained by the flying creature.

Insects have been a popular choice because of the relatively simple configuration of their flight system. Insects are generally smaller than birds or bats, and thus operate at a lower

Reynolds number and fly in more viscous air. The flapping frequency is quite high (often exceeding several hundred Hz, see [15] for data relating body mass to beating frequency) and the wing is usually not a streamlined airfoil, but is instead simpler in structure by consisting of a membrane reinforced with a fiber structure [15]. They require only two control inputs (stroke angle and pitch angle) which can be modeled sinusoidally. Control design is much easier, as averaging methods are valid within the high frequencies in insect flight. Unfortunately, the aerodynamics of insect flight vary significantly from the mechanics of bat flight and bird flight. Unsteady effects dominate their flight regime because of the extremely low Reynolds number [15]. For example, a species of dragonfly, *Anax parthenope*, is observed to have high maneuverability and adept at catching prey. Fast and skillful flight is enabled by its large wing load and high beating frequency, which can be represented as a combination of flapping and pitching motions [15].

In bird flight, the wings and method flight become quite complex. A bird's wings must generate lift and thrust to support the animal's weight and provide forward propulsion. Thus, the wing structure is adapted to bear the aerodynamic force and moment without adversely affecting the bird's flight performance during gliding or flapping. The wings are usually flexible and collapsible, with some exceptions [15]. Light weight bone structures in the wing, the complex airfoils formed by the feathers, and the addition of the elbow and wrist joints make birds more difficult to simulate [15]. Their flight mechanics also differ significantly from insects as well. Bird flight spans a large range of sizes, shapes, and methods. Specific power required to maintain a hovering flight decreases with disc loading (defined as weight divided by wing area, W/S) of wings. Thus, for birds, hovering is believed to only be possible with birds of smaller mass such as humming birds. Humming birds are small and heavily rely on unsteady effects to maintain their amazing hovering performance. Larger birds which are specialized for traveling long distances rely much more on soaring and the use of air currents, with flapping propulsion being used sparingly. In between these extremes are many other types of birds, some specialized for agility and others specialized for diving to catch prey

at high speed. Birds have muscles located mostly inside their body acting on the shoulder joint. Most of the motions are controlled by the shoulder, while the elbow helps fold the wing to shorten the span. There are smaller muscles inside the arm which allow the actuation of the elbow and wrist joints to control the shape of the wings during flapping flight [15]. The range of motion of the joints on the wing is limited by the physical constraints of the bird's body structure. An adequate phase lag is maintained between the flapping, pitching, and lead-lag wing motions to maintain an approximately elliptical wing orbit with respect to the body of the bird. The stroke plane is defined as the major axis of this elliptical wing trajectory. Extremum of lifts are obtained midway through the downstroke and upstroke respectively [15].

Bat flight differs from both bird and insect flight. They operate in a Reynolds number range where unsteady effects are important at low speeds but decreasingly less important at the higher range of flight speeds[27]. The tension on the wing membrane is controlled by a combination of several joints, the legs and the numerous finger joints. Larger species of bats behave similarly to large birds, relying more on soaring than flapping. Smaller bats, specialized as insect hunters, have developed extremely high agility and flap continuously. Because bats are equally or even more complicated than birds, and because of their flight performance, we have chosen to model our robotic test bed after a bat. Bat flight is also well suited to central pattern generator control because it relies heavily on the synchronization of phase differences between several different oscillatory motions.

2.2.2 Central Pattern Generators

A principle factor in choosing how many degrees of freedom were necessary for the test bed was based on biological principles. Another biological principle followed in the design of the bat was choosing the control scheme to be used. Many creatures produce their motion by synchronizing periodic motions of limbs, such as running, swimming or flapping. They do this by coupling biological oscillators and synchronizing their outputs. Biological

oscillators rely on short timescale (ms) neuron dynamics including spike-bursting, spike frequency adaptation, and post-inhibitory rebound. Herrero-Carrón, et. al. [28] designed a control law for modular robots by approximating short timescale neuron dynamics. Because there is such a short timescale required for integration, the neuron dynamics were integrated offline. We are unlikely to be able to perform such strict mimicry in an online controller as we add additional neurons for feedback, active control of phase differences, or gait transitions.

In order to make online control more feasible, we can emulate these biological oscillators by using limit cycle oscillators coupled together. A limit cycle oscillator is a nonlinear model that converges to a stable trajectory which is called the limit cycle. Because of this convergence the oscillator will quickly forget disturbances and converge back to the stable limit cycle. If the oscillator itself is a smooth vector field, we can smoothly transition between desired trajectories without abrupt changes being required in the motor output.

Following Chung and Dorothy [1], we use the following limit-cycle model called the Hopf oscillator, named after the supercritical Hopf bifurcation model with $\sigma = 1$:

$$\frac{d}{dt}\begin{pmatrix} u-a \\ v \end{pmatrix} = \begin{bmatrix} -\lambda\left(\frac{(u-a)^2+v^2}{\rho^2} - \sigma\right) & -\omega(t) \\ \omega(t) & -\lambda\left(\frac{(u-a)^2+v^2}{\rho^2} - \sigma\right) \end{bmatrix} \begin{pmatrix} u-a \\ v \end{pmatrix} + \mathbf{u}(t) \tag{2.1}$$

$$\text{Or, } \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}; \rho; \sigma) + \mathbf{u}(t), \text{ where } \mathbf{x} = (u-a, v)^T$$

where the $\lambda > 0$ denotes the convergence rate to the symmetric limit circle of the radius $\rho > 0$ and $\mathbf{u}(t)$ is an external or coupling input. A rigorous proof that coupled networks of Hopf oscillators on balanced graphs exhibit smooth exponentially stable behavior in both oscillatory mode and fixed point mode can be found in [1].

The possibly time-varying parameter $\omega(t) > 0$ determines the oscillation frequency of the limit cycle, which in our case refers to the frequency of any of the flapping, pitching, or lead-lag wing motions. A time-varying $a(t)$ sets the bias to the limit cycle such that it converges to $u(t) = \rho\cos(\omega t + \delta) + a$ and $v(t) = \rho\sin(\omega t + \delta)$ on a circle. The output

variable to generate the desired oscillatory motion of each joint is the first state u from the Hopf oscillator model in Eq. (2.1).

Synchronization means an exact match of the scaled amplitude or the frequency in this paper. Hence, phase synchronization permits different actuators to oscillate at the same frequency but with a prescribed phase lag. In essence, each CPG dynamic model in Eq. (2.1) is responsible for generating the limiting oscillatory behavior of a corresponding joint, and the diffusive coupling among CPGs reinforces phase synchronization. For example, the flapping angle has roughly a 90-degree phase difference with the pitching joint to maintain a positive angle of attack. A relation between mean efficiency, defined as the ratio of work performed by mean thrust with speed to required power, reduced frequency and the phase shifts of wing motions is derived in detail in [15]. Through theoretical computations of the unsteady aerodynamics characteristics of a two-dimensional thin airfoil based on studying harmonics, the phase difference between flapping and pitching motions which exhibits optimal mean efficiency is shown to be close to 90 degrees. Observations of the flight of dragonflies show that the phase difference between flapping and pitching motions is shown to be also close to 90 degrees [15]. The oscillators are connected through diffusive couplings, and the i-th Hopf oscillator can be rewritten with a diffusive coupling with the phase-rotated neighbor.

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i; \rho_i) - k \sum_{j \in \mathcal{N}_i}^{m_i} \left(\mathbf{x}_i - \frac{\rho_i}{\rho_j} \mathbf{R}(\Delta_{ij}) \mathbf{x}_j \right) \tag{2.2}$$

where the Hopf oscillator dynamics $\mathbf{f}(\mathbf{x}_i; \rho_i)$ with $\sigma = 1$ is defined in Eq. (2.1), $\mathcal{N}_i$ denotes the set that contains only the local neighbors of the i-th Hopf oscillator, and m_i is the number of the neighbors. The 2×2 matrix $\mathbf{R}(\Delta_{ij})$ is a 2-D rotational transformation of the phase difference Δ_{ij} between the i-th and j-th oscillators. The positive (or negative) Δ_{ij} indicates how much phase the i-th member leads (or lags) from the j-th member and $\Delta_{ij} = -\Delta_{ji}$. The positive scalar k denotes the coupling gain.

2.3 State of the Art of Flapping Flight MAVs

Some previous examples of flapping flight models can be found in [5], [29], [30], [31], [32], [33], and [6], as well commercially available products. More examples and research can be found in [2]. Most of these systems use a crankshaft mechanism to produce the flapping motions by converting a rotary motion from a motor to a linear oscillating motion, and are therefore limited to producing sinusoidal motions of a fixed amplitude with possibly variable frequency. However, experiments with high speed cameras have shown that the flapping motions in bats are not sinusoidal [34]. Furthermore, several parameters of the flapping motion change depending on flight conditions. The amplitude of wing motions vary, along with the phase difference between different wing motions, the wing beat frequency and the angle of attack. Studies of insect flight such as in [5]] and [29] accurately model insect flight by allowing changes in pitch and stroke plane angle. However, these systems do not allow changes in stroke amplitude, and thus there is no ability to generate arbitrary stroke motions.

While there are freely flying ornithopters, capable of flying only by flapping their wings, there are some issues and/or simplification issues which still exist. Some flapping flight vehicles have a large wing span on the order of 2 meters; as tall if not taller than the average human being. Thus, their large size allows the aircraft to take advantage of gliding effects. Other flapping flight MAVs are small, closer to a hummingbird or insect in size. The beating frequency of these vehicles is fast enough such that they can use averaging to control their flight instead of finely tuned, precise wing movements. Moreover, many of these ornithopters or otherwise flapping flight aerial vehicles use a tail; that of which bats do not usually have. Their wing motions are often simplified and constrained to move only in certain directions in certain fashions. Thus, flapping flight inspired by bats remains, as of this writing, fairly novel with many things still unknown or not well understood.

Chapter 3

Flapping Flight Kinematics and Dynamics

3.1 Introduction

We derive some basic kinematics, dynamics, and aerodynamics for our test bed and show they can and will be used for further experimentation and research regarding flapping flight MAVs.

3.2 Testbed Kinematics and Unsteady Aerodynamics

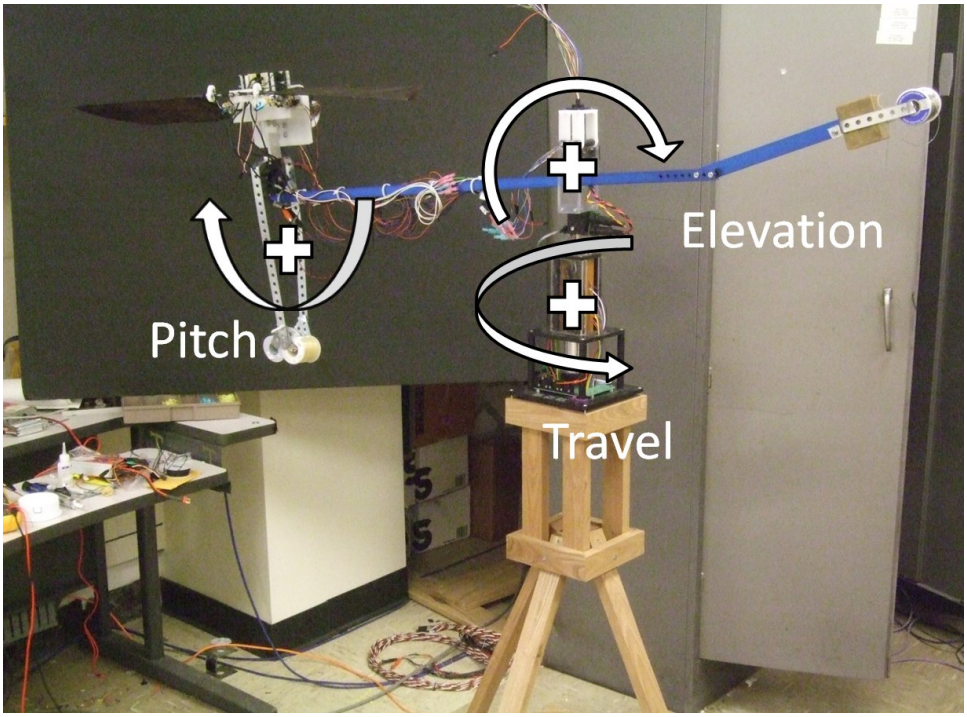


Figure 3.1: RoboBat Testbed

The current RoboBat is not intended to be a free flying platform. It is intended as a testbed for CPG control designs, experimental confirmation of unsteady aerodynamics, and experimental determination of optimal wing motions. The weight and power requirements have not been optimized for free flight. In order to test longitudinal control strategies, it has been attached to a Quanser pendulum platform, which provides encoder feedback signals that we use for control. Figure 4.12 shows the three degrees of freedom: travel, elevation, and pitch (λ,ϵ,p).

Of note is the fact that the pitch rotation point is not near the center of gravity of the bat. To make experimentation feasible, we have affixed a counterweight on the pitch arm. By moving this counterweight or changing its mass, we can alter the natural stability of the pitch motion. One consequence of this scheme is that the pitch motion has an artificially high moment of inertia. Therefore, we expect that our moment-producing control schemes for a tailless vehicle will have even more effectiveness in a free flier. To move toward computations of actual forces and moments generated, we desire dynamic modeling of the pendulum set-up and the unsteady aerodynamics. If we define our generalized coordinates to be $[q_1, q_2, q_3] = [\epsilon, p, \lambda]$, then using Lagrange's equations, $\frac{d}{dt}\frac{\partial L(q,\dot{q})}{\partial \dot{q}} - \frac{\partial L(q,\dot{q})}{\partial q} = F$ and algebraic manipulations, we can transform the EOM to standard robot form [35].

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) = \tau. \tag{3.1}$$

$M(q)$ is known to be a symmetric positive definite matrix [35]. In this case, we have elements

$$\begin{aligned}
M_{11} = \quad & \ell_a{}^2\, m_b + \ell_a{}^2\, m_c + \ell_w{}^2\, m_w \\
& + \ell_b{}^2\, m_b \cos(q_2)^2 + \ell_c{}^2\, m_c \cos(q_2)^2 \\
M_{12} = M_{21} = \quad & -\ell_a\, \sin(q_2)\, (\ell_b\, m_b - \ell_c\, m_c) \\
M_{13} = M_{31} = \quad & \sin(q_2)\, \Big(m_b\, \cos(q_1)\, \cos(q_2)\, \ell_b{}^2 \\
& + \ell_a\, m_b\, \sin(q_1)\, \ell_b + m_c\, \cos(q_1)\, \cos(q_2)\, \ell_c{}^2 \\
& - \ell_a\, m_c\, \sin(q_1)\, \ell_c \Big) \\
M_{22} = \quad & m_b\, \ell_b{}^2 + m_c\, \ell_c{}^2 \\
M_{23} = M_{32} = \quad & -m_b\, \sin(q_1)\, \ell_b{}^2 + \ell_a\, m_b\, \cos(q_1)\, \cos(q_2)\, \ell_b \\
& - m_c\, \sin(q_1)\, \ell_c{}^2 - \ell_a\, m_c\, \cos(q_1)\, \cos(q_2)\, \ell_c \\
M_{33} = \quad & \ell_b{}^2\, m_b + \ell_c{}^2\, m_c + \ell_a{}^2\, m_b \cos(q_1)^2 \\
& + \ell_a{}^2\, m_c \cos(q_1)^2 + \ell_w{}^2\, m_w \cos(q_1)^2 \\
& - \ell_b{}^2\, m_b \cos(q_1)^2 \cos(q_2)^2 - \ell_c{}^2\, m_c \cos(q_1)^2 \cos(q_2)^2 \\
& - 2\, \ell_a\, \ell_b\, m_b\, \cos(q_1)\, \cos(q_2)\, \sin(q_1) \\
& + 2\, \ell_a\, \ell_c\, m_c\, \cos(q_1)\, \cos(q_2)\, \sin(q_1)
\end{aligned} \tag{3.2}$$

where ℓ_x and m_x are length and mass, respectively. Subscripts a,w,b, and c denote the point of pitch rotation, elevation counterweight, RoboBat, and pitch counterweight, respectively.

The skew-symmetric $C(q, \dot{q})$ matrix is defined as [35]

$$c_{ij} = \frac{1}{2} \sum_{k=1}^{n} \frac{\partial M_{ij}}{\partial q_k} \dot{q}_k + \frac{1}{2} \sum_{k=1}^{n} \left(\frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i} \right) \dot{q}_k \tag{3.3}$$

and is computed to be

$$
\begin{aligned}
C_{11} = & \quad -\frac{\dot{q}_2 \, \sin(2\,q_2) \left(m_b \, {\ell_b}^2 + m_c \, {\ell_c}^2 \right)}{2} \\
C_{12} = & \quad -\cos(q_2) \, \Big({\ell_b}^2 \, m_b \, \dot{q}_1 \, \sin(q_2) + {\ell_c}^2 \, m_c \, \dot{q}_1 \, \sin(q_2) \\
& \quad + \ell_a \, \ell_b \, m_b \, \dot{q}_2 - \ell_a \, \ell_c \, m_c \, \dot{q}_2 \\
& \quad - {\ell_b}^2 \, m_b \, \dot{q}_3 \, \cos(q_1) \, \cos(q_2) - {\ell_c}^2 \, m_c \, \dot{q}_3 \, \cos(q_1) \, \cos(q_2) \\
& \quad - \ell_a \, \ell_b \, m_b \, \dot{q}_3 \, \sin(q_1) \\
& \quad + \ell_a \, \ell_c \, m_c \, \dot{q}_3 \, \sin(q_1) \Big) \\
C_{13} = & \quad {\ell_a}^2 \, m_b \, \dot{q}_3 \, \cos(q_1) \, \sin(q_1) + {\ell_a}^2 \, m_c \, \dot{q}_3 \, \cos(q_1) \, \sin(q_1) \\
& \quad + {\ell_w}^2 \, m_w \, \dot{q}_3 \, \cos(q_1) \, \sin(q_1) - \ell_a \, \ell_b \, m_b \, \dot{q}_3 \, \cos(q_2) \\
& \quad + \ell_a \, \ell_c \, m_c \, \dot{q}_3 \, \cos(q_2) + {\ell_b}^2 \, m_b \, \dot{q}_2 \, \cos(q_1) \, \cos(q_2)^2 \\
& \quad + {\ell_c}^2 \, m_c \, \dot{q}_2 \, \cos(q_1) \, \cos(q_2)^2 + \ell_a \, \ell_b \, m_b \, \dot{q}_2 \, \cos(q_2) \, \sin(q_1) \\
& \quad - \ell_a \, \ell_c \, m_c \, \dot{q}_2 \, \cos(q_2) \, \sin(q_1) - {\ell_b}^2 \, m_b \, \dot{q}_3 \, \cos(q_1) \, \cos(q_2)^2 \, \sin(q_1) \\
& \quad - {\ell_c}^2 \, m_c \, \dot{q}_3 \, \cos(q_1) \, \cos(q_2)^2 \, \sin(q_1) + 2 \, \ell_a \, \ell_b \, m_b \, \dot{q}_3 \cos(q_1)^2 \, \cos(q_2) \\
& \quad - 2 \, \ell_a \, \ell_c \, m_c \, \dot{q}_3 \cos(q_1)^2 \, \cos(q_2)
\end{aligned}
$$

$$
\begin{aligned}
C_{21} = \quad & \cos(q_2)\ \Big({\ell_b}^2\, m_b\, \dot{q}_1\ \sin(q_2) + {\ell_c}^2\, m_c\, \dot{q}_1\ \sin(q_2) \\
& -{\ell_b}^2\, m_b\, \dot{q}_3\ \cos(q_1)\ \cos(q_2) \\
& -{\ell_c}^2\, m_c\, \dot{q}_3\ \cos(q_1)\ \cos(q_2) \\
& -\ell_a\, \ell_b\, m_b\, \dot{q}_3\ \sin(q_1) \\
& +\ell_a\, \ell_c\, m_c\, \dot{q}_3\ \sin(q_1) \Big) \\
C_{22} = \quad & 0 \\
C_{23} = \quad & -\Big(\dot{q}_1\ \cos(q_2) + \dot{q}_3\ \cos(q_1)\ \sin(q_2) \Big)\ * \\
& \Big(m_b\ \cos(q_1)\ \cos(q_2)\ {\ell_b}^2 + \ell_a\, m_b\ \sin(q_1)\ \ell_b \\
& +m_c\ \cos(q_1)\ \cos(q_2)\ {\ell_c}^2 - \ell_a\, m_c\ \sin(q_1)\ \ell_c \Big) \\
C_{31} = \quad & -\Big(\cos(q_3)^2 + \sin(q_3)^2 \Big)\ \Big({\ell_a}^2\, m_b\, \dot{q}_3\ \cos(q_1)\ \sin(q_1) + {\ell_a}^2\, m_c\, \dot{q}_3\ \cos(q_1)\ \sin(q_1) \\
& +{\ell_w}^2\, m_w\, \dot{q}_3\ \cos(q_1)\ \sin(q_1) + {\ell_b}^2\, m_b\, \dot{q}_2\ \cos(q_1)\ \sin(q_2)^2 \\
& +{\ell_c}^2\, m_c\, \dot{q}_2\ \cos(q_1)\ \sin(q_2)^2 + {\ell_b}^2\, m_b\, \dot{q}_1\ \cos(q_2)\ \sin(q_1)\ \sin(q_2) \\
& +{\ell_c}^2\, m_c\, \dot{q}_1\ \cos(q_2)\ \sin(q_1)\ \sin(q_2) - \ell_a\, \ell_b\, m_b\, \dot{q}_1\ \cos(q_1)\ \sin(q_2) \\
& +\ell_a\, \ell_c\, m_c\, \dot{q}_1\ \cos(q_1)\ \sin(q_2) - {\ell_b}^2\, m_b\, \dot{q}_3\ \cos(q_1)\ \cos(q_2)^2\ \sin(q_1) \\
& -{\ell_c}^2\, m_c\, \dot{q}_3\ \cos(q_1)\ \cos(q_2)^2\ \sin(q_1) + \ell_a\, \ell_b\, m_b\, \dot{q}_3 \cos(q_1)^2\ \cos(q_2) \\
& -\ell_a\, \ell_c\, m_c\, \dot{q}_3 \cos(q_1)^2\ \cos(q_2) - \ell_a\, \ell_b\, m_b\, \dot{q}_3\ \cos(q_2)\ \sin(q_1)^2 \\
& +\ell_a\, \ell_c\, m_c\, \dot{q}_3\ \cos(q_2)\ \sin(q_1)^2 \Big)
\end{aligned}
\tag{3.4}
$$

$$
\begin{aligned}
C_{32} = \quad & \cos(q_1)\,\Big(\ell_b{}^2\, m_b\, \dot{q}_1 \cos(q_2)^2 - \ell_c{}^2\, m_c\, \dot{q}_1 - \ell_b{}^2\, m_b\, \dot{q}_1 \\
& + \ell_c{}^2\, m_c\, \dot{q}_1 \cos(q_2)^2 - \ell_a\, \ell_b\, m_b\, \dot{q}_2\, \sin(q_2) \\
& + \ell_a\, \ell_c\, m_c\, \dot{q}_2\, \sin(q_2) + \ell_b{}^2\, m_b\, \dot{q}_3\, \cos(q_1)\, \cos(q_2)\, \sin(q_2) \\
& + \ell_c{}^2\, m_c\, \dot{q}_3\, \cos(q_1)\, \cos(q_2)\, \sin(q_2) + \ell_a\, \ell_b\, m_b\, \dot{q}_3\, \sin(q_1)\, \sin(q_2) \\
& - \ell_a\, \ell_c\, m_c\, \dot{q}_3\, \sin(q_1)\, \sin(q_2)\Big) \\
C_{33} = \quad & \ell_a\, \ell_b\, m_b\, \dot{q}_1\, \cos(q_2) - \ell_a{}^2\, m_c\, \dot{q}_1\, \cos(q_1)\, \sin(q_1) \\
& - \ell_w{}^2\, m_w\, \dot{q}_1\, \cos(q_1)\, \sin(q_1) - \ell_a{}^2\, m_b\, \dot{q}_1\, \cos(q_1)\, \sin(q_1) \\
& - \ell_a\, \ell_c\, m_c\, \dot{q}_1\, \cos(q_2) + \ell_b{}^2\, m_b\, \dot{q}_1\, \cos(q_1)\, \cos(q_2)^2\, \sin(q_1) \\
& + \ell_b{}^2\, m_b\, \dot{q}_2 \cos(q_1)^2\, \cos(q_2)\, \sin(q_2) + \ell_c{}^2\, m_c\, \dot{q}_1\, \cos(q_1)\, \cos(q_2)^2\, \sin(q_1) \\
& + \ell_c{}^2\, m_c\, \dot{q}_2 \cos(q_1)^2\, \cos(q_2)\, \sin(q_2) - 2\, \ell_a\, \ell_b\, m_b\, \dot{q}_1 \cos(q_1)^2\, \cos(q_2) \\
& + 2\, \ell_a\, \ell_c\, m_c\, \dot{q}_1 \cos(q_1)^2\, \cos(q_2) + \ell_a\, \ell_b\, m_b\, \dot{q}_2\, \cos(q_1)\, \sin(q_1)\, \sin(q_2) \\
& - \ell_a\, \ell_c\, m_c\, \dot{q}_2\, \cos(q_1)\, \sin(q_1)\, \sin(q_2)
\end{aligned}
$$

The gravity term $\mathbf{g}(\mathbf{q}) = \frac{\partial \mathbf{V}(\mathbf{q})}{\partial \mathbf{q}}$ is computed to be

$$
\begin{aligned}
g_1 = \quad & g\,\Big(\ell_a\, m_b\, \cos(q_1) + \ell_a\, m_c\, \cos(q_1) \\
& - \ell_w\, m_w\, \cos(q_1) - \ell_b\, m_b\, \cos(q_2)\, \sin(q_1) + \ell_c\, m_c\, \cos(q_2)\, \sin(q_1)\Big) \\
g_2 = \quad & -g\, \cos(q_1)\, \sin(q_2)\, (\ell_b\, m_b - \ell_c\, m_c) \\
g_3 = \quad & 0
\end{aligned}
\tag{3.5}
$$

where g is acceleration due to gravity.

The forces and moments on the right hand side can be found as a function of wing kinematics and an aerodynamic model, The generalized forces remain intact through the

transformation to robot form, i.e., $F = \tau$, and are computed to be

$$\tau = \begin{pmatrix} \ell_a \left(\mathsf{L}\cos(q_2) + \mathsf{T}\sin(q_2)\right) \\ \mathsf{M} - \ell_b \mathsf{T} \\ \ell_a \left(\mathsf{T}\cos(q_2) - \mathsf{L}\sin(q_2)\right) \end{pmatrix}, \tag{3.6}$$

where L,T, and M are found later in the aerodynamic model. This formulation can then be applied to a free-flying MAV. Here, we present a refinement on the aerodynamic model of [1].

The pendulum rig consists of

1. A bar hinged at its center of gravity such that it can spin about the vertical axis (angle given by λ, positive counter-clockwise) and rotate upwards and downwards in the vertical plane (angle denoted by ϵ, positive downwards).

2. A compound pendulum mounted on one end of the bar consisting of two point masses: the robotic bat itself modeled as a point mass m_b and a mass m_c, which functions as a pitch counterweight and is also modeled as a point mass. The compound pendulum is free to swing in the plane normal to the bar, with the swing angle given by p.

3. An elevation counter-weight, m_w, located at the opposite end of the bar as the bat.

Three frames of reference can be defined for this system, given an inertial frame of reference I fixed to the Earth:

1. A frame B fixed to the compound pendulum with its origin at the suspension point. The frame B parallel to the aircraft body axis frame centered at the aircraft CG.

2. A frame P with its origin at the bar's hinge point such that under nominal conditions, the axes of P and B are parallel to each other.

3. A frame S constructed locally at every wing station for calculation the local wind velocity and the aerodynamic forces and moments.

The frame I is first rotated about the z-axis by an angle λ, followed by a rotation about the x-axis by ϵ to coincide with the P frame. Therefore, the following rotation matrix is obtained to transform the components of a vector from I to P:

$$\begin{aligned} R_{PI} &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\epsilon & -\sin\epsilon \\ 0 & \sin\epsilon & \cos\epsilon \end{bmatrix} \begin{bmatrix} \cos\lambda & -\sin\lambda & 0 \\ \sin\lambda & \cos\lambda & 0 \\ 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \cos\lambda & -\sin\lambda & 0 \\ \cos\epsilon\sin\lambda & \cos\epsilon\cos\lambda & -\sin\epsilon \\ \sin\epsilon\sin\lambda & \sin\epsilon\cos\lambda & \cos\epsilon \end{bmatrix} \end{aligned} \tag{3.7}$$

The frame P is rotated about the y-axis to obtain frame B:

$$R_{BP} = \begin{bmatrix} \cos p & 0 & -\sin p \\ 0 & 1 & 0 \\ \sin p & 0 & \cos p \end{bmatrix} \tag{3.8}$$

The rotation matrix from B to S is obtained by performing the standard $3-2-1$ rotations through angles ψ, θ and $-\phi$, which correspond to lead-lag, pitching, and flapping motions respectively:

$$R_{SB} = \begin{bmatrix} \cos\phi\cos\psi & \sin\psi\cos\phi & \sin\phi \\ -\sin\phi\sin\theta\cos\psi - \sin\psi\cos\theta & -\sin\phi\sin\psi\sin\theta + \cos\theta\cos\psi & \sin\theta\cos\phi \\ -\sin\phi\cos\theta\cos\psi - \sin\psi\sin\theta & -\sin\phi\sin\psi\cos\theta + \sin\theta\cos\psi & \cos\theta\cos\phi \end{bmatrix} \tag{3.9}$$

The matrix $R_{SP} = R_{SB}R_{BP}$ transforms the coordinates of a vector from the P frame to S frame, and will be useful while calculating velocities and forces.

Let ω_P denote the angular velocity of the horizontal rod in the P frame, i.e., $\omega_P = \begin{bmatrix} \dot{\epsilon} & 0 & -\dot{\lambda} \end{bmatrix}^{\mathrm{T}}$. Let $\omega_B = [0 \ \dot{p} \ 0]^{\mathrm{T}}$ denote the angular velocity of the compound pendulum about O_B. Finally, let ω denote the angular velocity of a wing with components in the B

frame (a similar expression can be derived for the other wing). It follows that

$$\omega = \begin{bmatrix} -\dot{\phi} - \dot{\psi}\sin\theta \\ \dot{\theta}\cos\phi - \dot{\psi}\cos\theta\sin\phi \\ \dot{\theta}\sin\phi + \dot{\psi}\cos\theta\cos\phi \end{bmatrix} \tag{3.10}$$

Let $\bar{Y} = [0\ l_a\ 0]^{\mathrm{T}}$ denote the position vector from O_P to O_B. Let $\bar{z} = [0\ 0\ -l_b]^{\mathrm{T}}$ denote the position vector from O_P to the bat CG, and $\bar{y} = [0\ y\ 0]^{\mathrm{T}}$ denote the vector from bat CG to a wing station with components in the S frame. Then, the local wind velocity at a wing station is given by

$$\mathbf{V} = \begin{bmatrix} U \\ V \\ W \end{bmatrix} = \underbrace{R_{SP}S(\omega_P)\bar{Y} + R_{SB}S(\omega_B + R_{BP}\omega_P)\bar{z}}_{\text{equivalent to flight speed } V_\infty} + S(R_{SB}(\omega + \omega_B) + R_{SP}\omega_P)\bar{y} \tag{3.11}$$

The local angle of attack is given by

$$\alpha = \tan^{-1}\left(\frac{W}{U}\right) \tag{3.12}$$

The local acceleration is assumed to arise entirely from flapping. Since the model developed in this paper is intended to be as generic as possible, the model proposed by Goman and Khrabrov [36] is presented as a candidate model for computing the lift and the quarter chord moment while drag is estimated assuming the classic drag polar. It is estimated that Goman and Khrabrov's model offers at least two advantages over the existing models (e.g., Theodorsen or Peters [37]). First, the model is cast in the form of a single ordinary differential equation (ODE) and two algebraic equations, one each for life and the quarter chord pitching moment. The state variable for the ODE corresponds, physically, to the chordwise location of flow separation on the airfoil. Therefore, the model is quite easy to implement

as part of a numerical routine. Second, the model is inherently nonlinear and applicable to post-stall conditions.

The following equation describes the movement of the separation point for unsteady flow conditions

$$\tau_1 \dot{\nu} + \nu = \nu_0 \left(\alpha - \tau_2 \dot{\alpha}\right) \tag{3.13}$$

where τ_1 is the relaxation time constant, $\tau 2$ captures the time delay effects due to the flow, while ν_0 is an expression for the nominal position of the separation point. These three parameters need to be identified experimentally or using CFD for the particular airfoil under consideration. However, at present time the wings are assumed to be rigid flat plates. The coefficients of lift and quarter-chord moment are then given by

$$C_L = \frac{\pi}{2} \sin\left(\alpha\left(1 + \nu + 2\sqrt{\nu}\right)\right)$$

$$C_{m_{ac}} = \frac{\pi}{2} \sin\left(\alpha\left(1 + \nu + 2\sqrt{\nu}\right)\right)\left[\frac{5 + 5\nu - 6\sqrt{\nu}}{16}\right] \tag{3.14}$$

The lift and the quarter chord moment per unit span are then given by

$$L(y) = 0.5\rho V(y)^2 c C_l^* + \frac{\pi}{4}\rho c^2 \left(\ddot{\xi} + V_\infty \dot{\alpha} - (x_a - 0.25)c\ddot{\alpha}\right)$$

$$M(y) = 0.5\rho V(y)^2 c^2 C_{m_{ac}}^* + \frac{\pi}{4}\rho c^2 \left(V_\infty \dot{\xi} + \frac{(x_a - 0.25)c\ddot{\xi}}{2} + V_\infty^2 \alpha - c^2\left(\frac{1}{32} + (x_a - 0.25)^2\right)\ddot{\alpha}\right) \tag{3.15}$$

where $\theta(y)$ is the twist angle, ρ denotes the density of air and ξ is the transverse displacement of the wing due to flapping. Furthermore, $V = \|\mathbf{V}\|$ is the local wind speed with $\mathbf{V}$ defined in Eq. (3.11), and V_∞ is the freestream speed of the aircraft. The last term of each expression was added to Goman's original model [36] and corresponds to the apparent mass effect [38].

For the sectional drag coefficient, there is unfortunately no simple expression for such. Assuming laminar flow on the wing, the sectional drag coefficient can be written as $C_D =$

$\frac{0.89}{\sqrt{RE}} + \frac{1}{\pi e A_R} C_L^2$ where A_R is the aspect ratio of the wing, $Re = \frac{\rho c V_{\text{inf}}}{\mu}$ is the chordwise Reynolds number, and e is Oswald's efficiency factor. A refined model for calculating drag, incorporating dynamic stall, may be found in [38].

Chapter 4

Description of the Robotic Bat Test Bed

4.1 Introduction

In this chapter we describe the details of the technical components that the robotic bat test bed consists of, as well as essential hardware and software that supplements the overall test bed environment.

4.2 Design of Robotic Bat

4.2.1 Previous Work

The previous iteration of the robotic bed test bed was designed to be mounted on a stationary stand and mounted on a force-torque sensor. This set-up could be placed in a wind tunnel where forces and moments generated by the wing motions in an air stream could be measured. Further details regarding this model can be found in [1]. One major goal for redesigning the test bed was to allow it to move in a limited 3-degree of freedom fashion. Another goal was to used the robotic bat's orientation and position as feedback data for a closed loop controller.

4.2.2 Current Robobat Design

The robotic bat is a highly controllable platform, modeled after the kinematics of a bat. Eight degrees of freedom are provided; three in each shoulder joint and two for the amplitude of

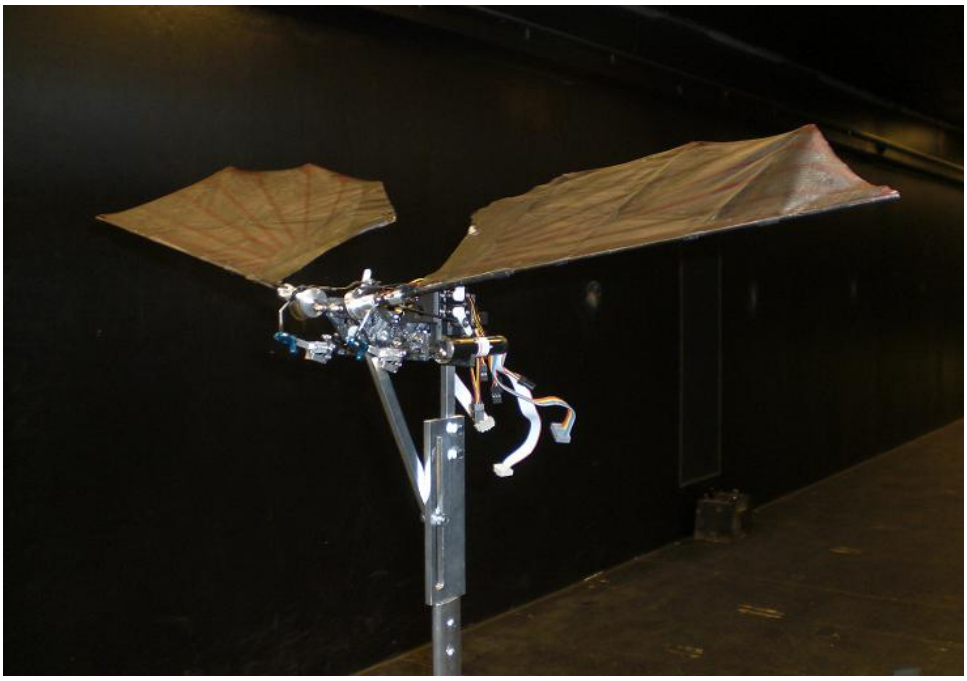
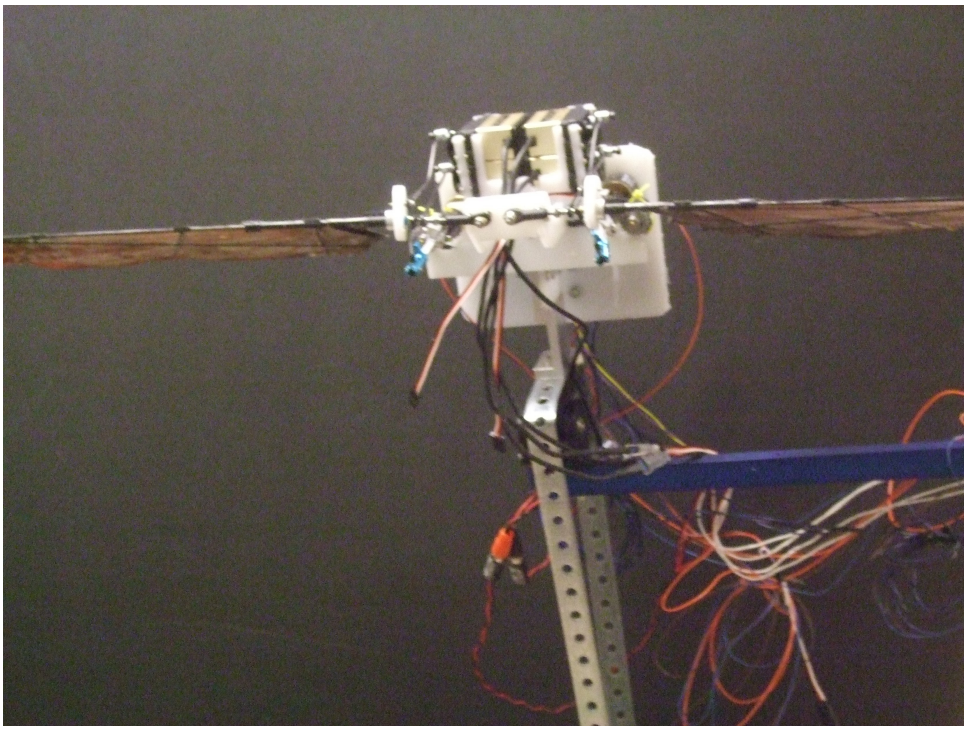
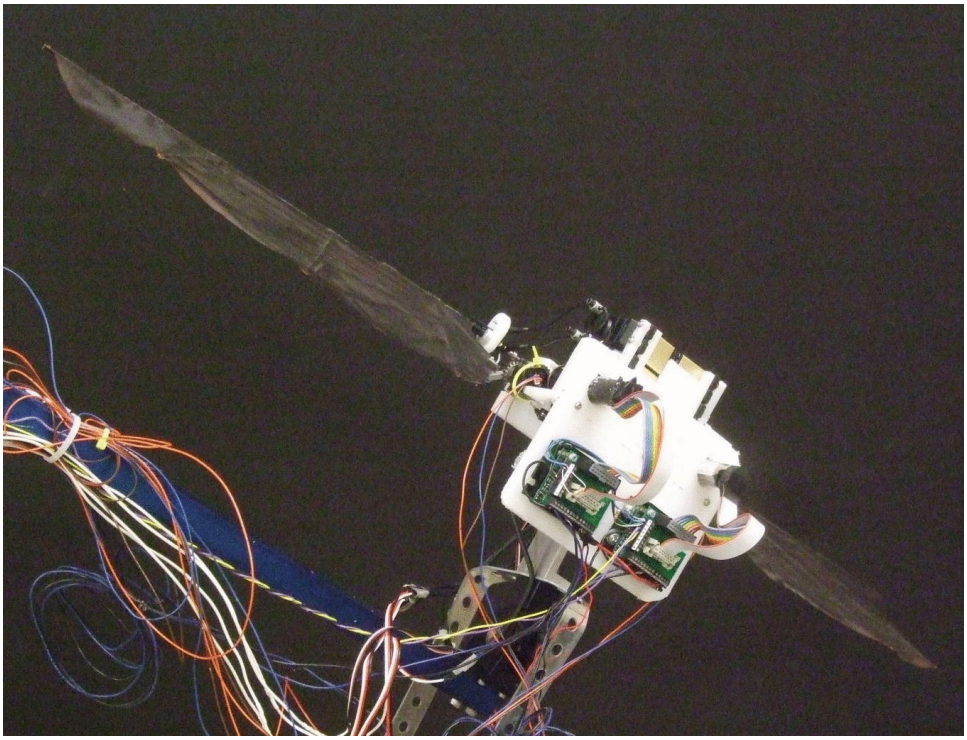


Figure 4.1: Previous robotic bat design[1]

flapping. The shoulder joints are also analogous to human shoulder joints, able to move forward, backwards, up, down, and can twist in both directions. Those motions correspond to lead-lag, flapping, and pitching respectively. These 8 degrees of freedom are combined with variable speed motors to allow for maximum flapping in control schemes. The flapping motion of the wings are independently powered by two 10 watt Maxon motors. Electronic controllers for the two Maxon motors allow for precise control of motor velocity and thus flapping frequency. All other degrees of freedom are controlled with Futaba servos. In the previous bat design from [1], the servos actuating the lead-lag and pitch motions were feather servos. The new structure uses bigger and more powerful servos for increased torque and speed. Two US Digital absolute encoders are attached to the sides of the two motors and connected with gears in order to measure the absolute position of the wings. This position data is used to create a closed loop controller for the wings and allows them to synchronize to a desired signal from the CPG's. The membrane of the wings is taken from a commercially available ornithopter, while the supporting wing structure is custom made

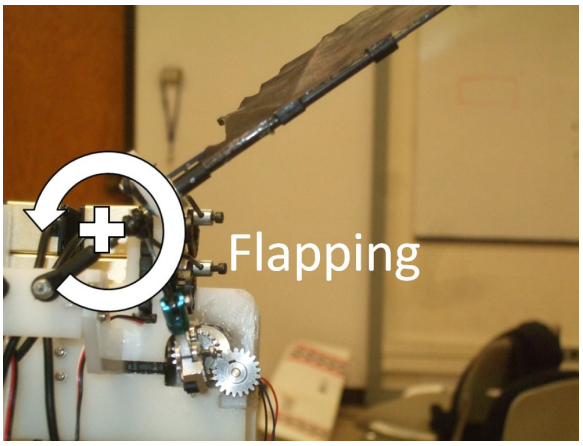


(a) Front View

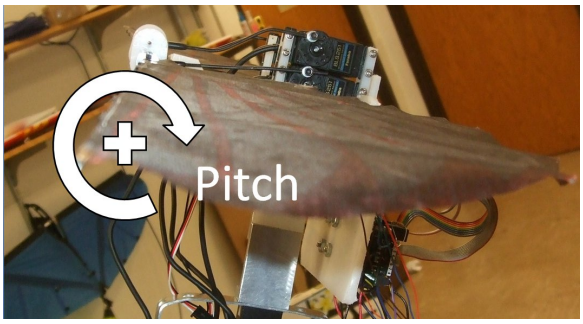


(b) Back View

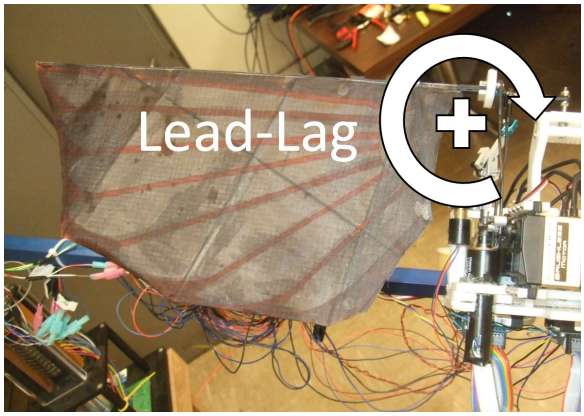
Figure 4.2: Front and back views of robotic bat, mounted on Quanser 3DOF Helicopter stand



(a) Flapping Motion



(b) Pitching Motion



(c) Lead-Lag Motion

Figure 4.3: Positive wing motion directions

using a combination of carbon fiber, plastic, and metal parts.

The flapping amplitude is varied by a mechanism consisting of a moving crank arm and a rotating slider. A tail pitch plate used on RC helicopter tail rotors is used to control the flapping amplitude. As servo controls the slider on the tail pitch plate and moves it to vary the distance from the motor shaft to the crank arm. This changes the flapping amplitude. Additionally, the servo has to move only a small angular distance to change the flapping amplitude.

The main frame of the test bed was fabricated with a CNC machine. This method allows for quick changes in the design to be made. More complex parts can be machined. High density polyethylene was chosen as the main frame material for its low cost and ease of use

Figure 4.4: Wing of robotic bat

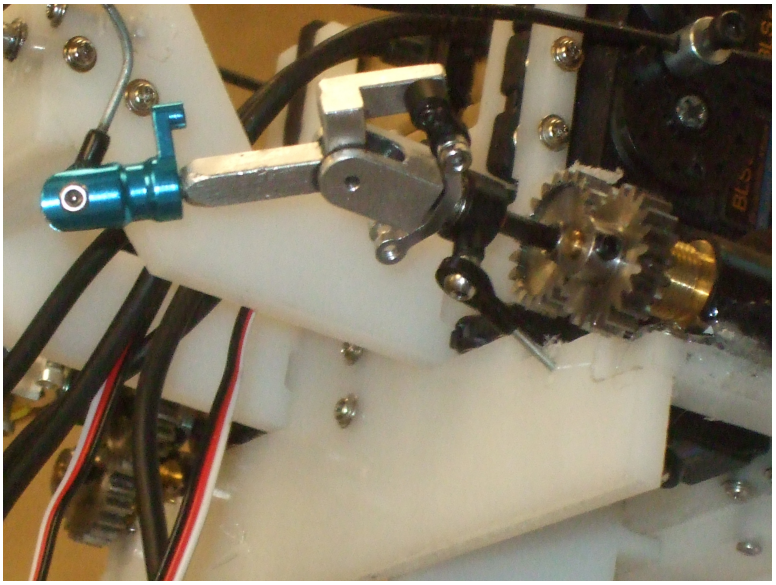


Figure 4.5: Drive shaft to actuate flapping, with motor and encoder shown

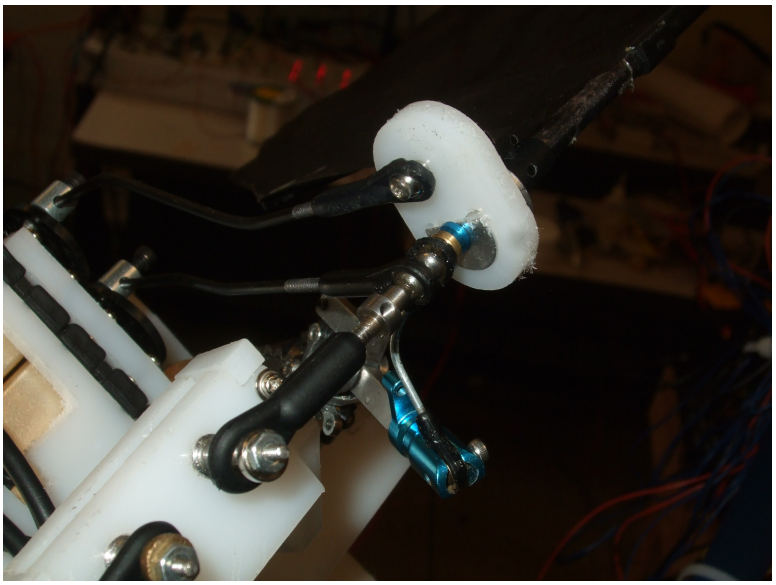


Figure 4.6: Shoulder joint, with pushrods for flapping, pitch, and lead-lag

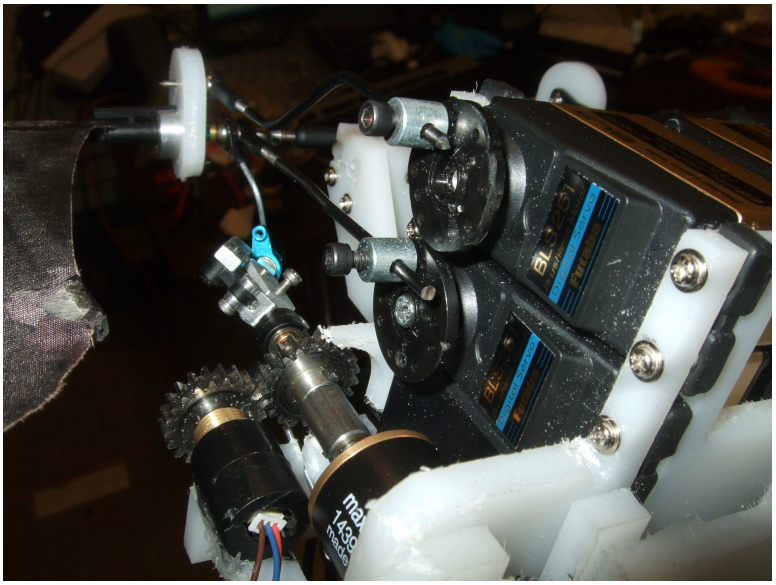


Figure 4.7: Pitch (top) and lead-lag (bottom) servos

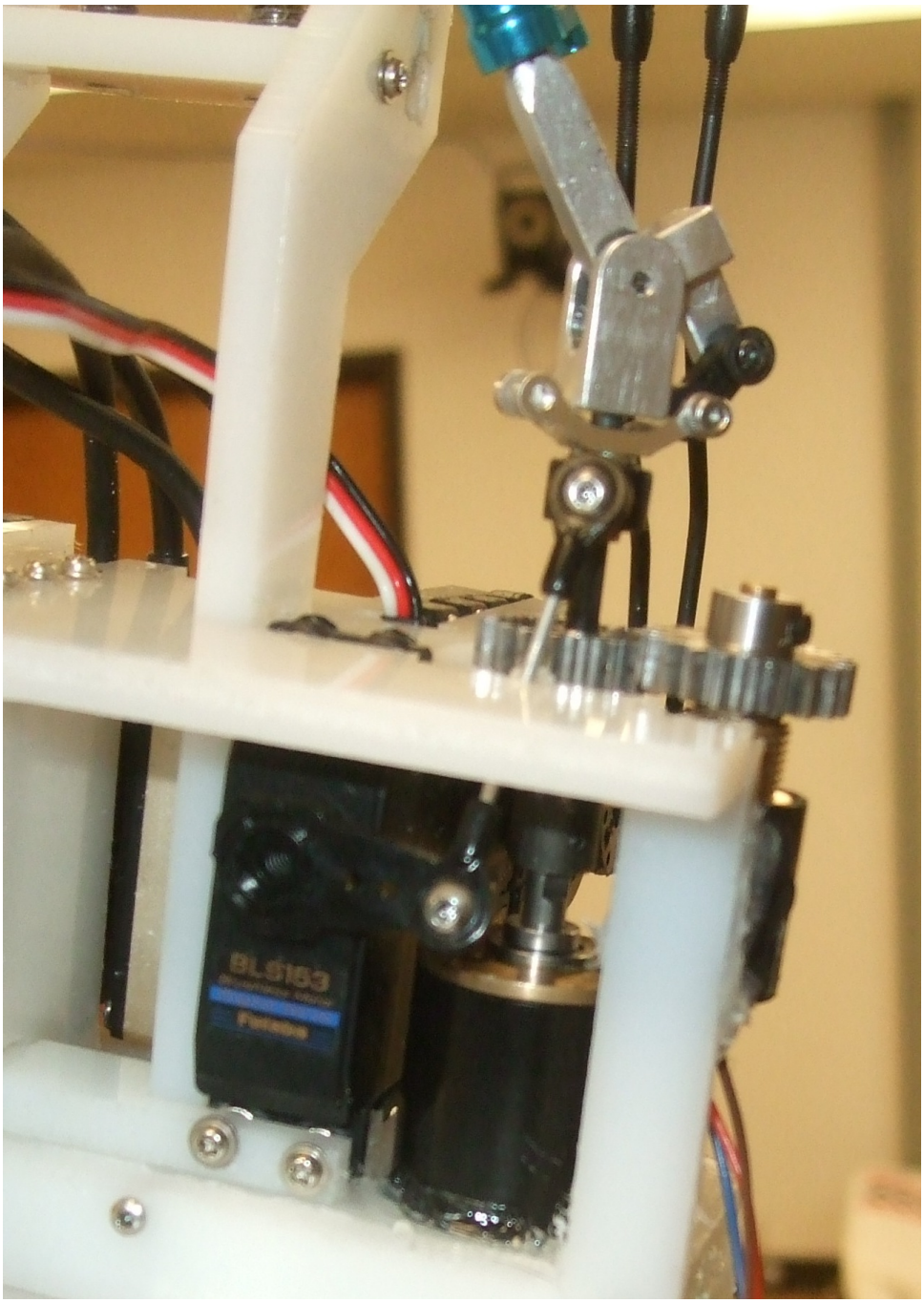


Figure 4.8: Close-up of amplitude controlling servo and drive shaft

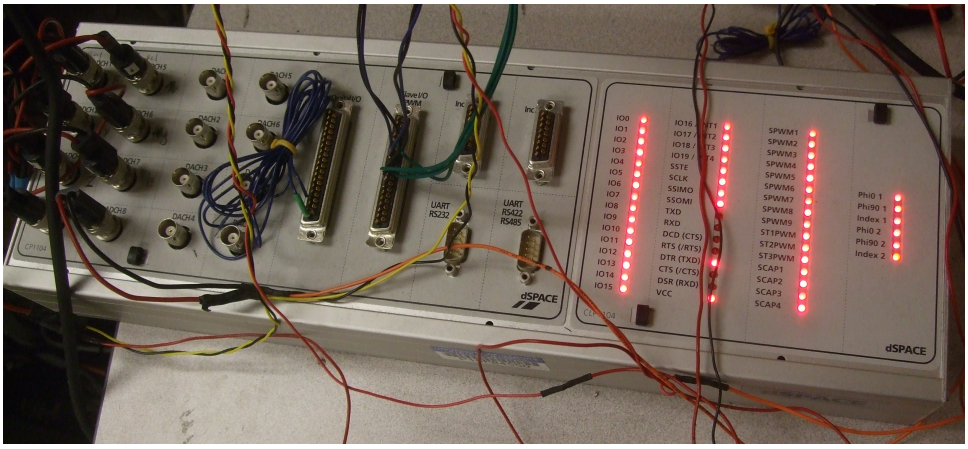


Figure 4.9: dSPACE connector board

with the CNC machine. All drive train materials are constructed of aluminum and steel, with non standard drive train parts being machined.

Controlling the robotic bat is done with a dSPACE DS1104 Controller board. The setup consists of a PPC board, which is mounted inside the computer via PCI slot, and a connector board which connects to the PPC board and provides an I/O interface for wiring to the robotic bat. The connector board outputs PWM signals and square waves to control the servos and motors, respectively. An analog/digital converter on the connector board allows the absolute motor encoders to output a voltage to the dSPACE board and have it converted to a digital signal, which can be read by the computer.

4.3 Design of 3DOF Test Bed

To hold the robotic bat in place while providing movement in three directions, the Quanser 3DOF Helicopter was used for its stand, encoders and Q4 board. The stand of the Quanser 3DOF Helicopter uses built-in encoders to measure pitch, travel, and elevation, which can be feed information regarding the robotic bat's position and orientation back into the controller of the robotic bat.

A wooden base was constructed to provide higher elevation of the stand and to provide

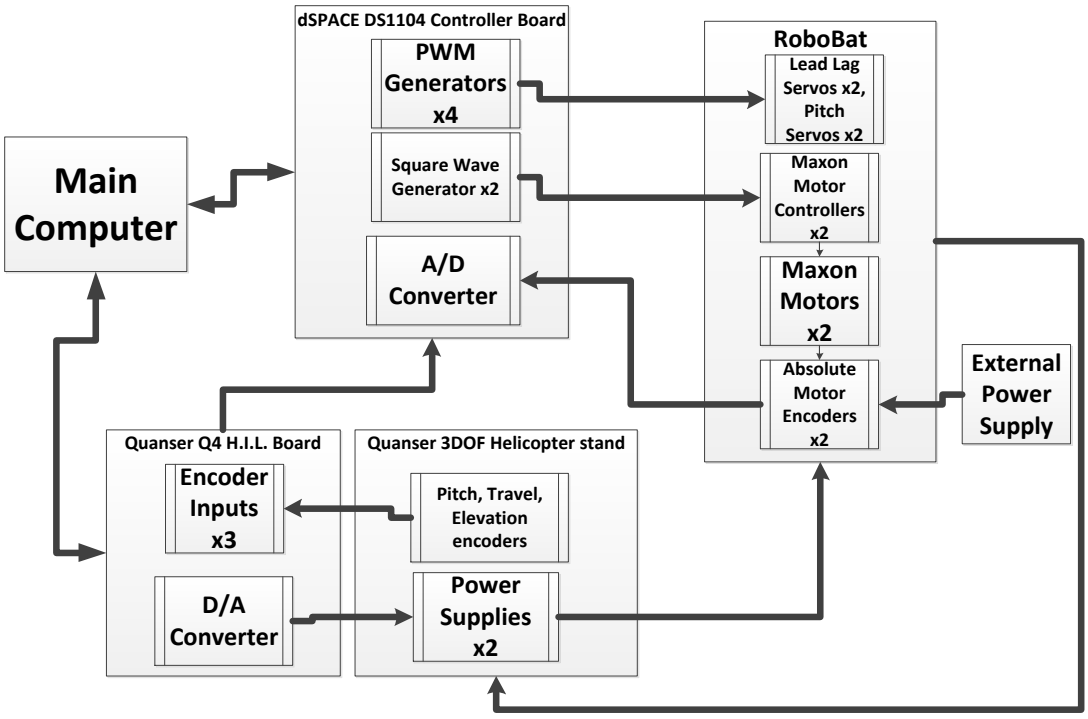


Figure 4.10: Schematic of test bed setup

a larger, more stable base for the entire test bed to rest upon. To power the components of the robotic bat, the RCA connectors which were originally used to power the helicopter rotors included with the stand are used to provide the voltage necessary for the servos and motors of the bat.

Electrical connections to the Quanser 3DOF Helicopter are controlled by the Quanser Q4 Hardware in the Loop (HIL) board. The Q4 board provides us with D/A voltage outputs and encoder inputs [39]. Two D/A voltage outputs on the Q4 board allow the necessary voltages needed for the servos and motors of the robotic bat. Three encoder inputs are used to read the encoder counts from the travel, pitch, and elevator encoders.

A constant electrical connection must be maintained between the controlling computer and the robotic bat so that controlling signals can be sent. dSPACE is used to create and send the controlling signals to the robotic bat, and since a wireless transfer method between dSPACE and the robotic bat's servos and motors was deemed too difficult if not infeasible,

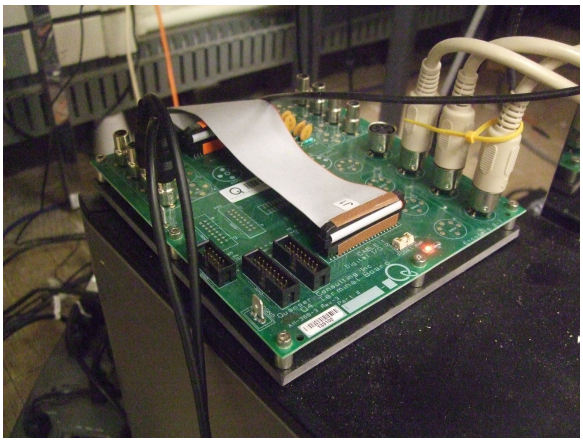
an Orbex Group slip ring was used to main a contact electrical connection while still allowing the stand to rotate freely. The wires from dSPACE were threaded through the ceiling of the laboratory to prevent physical interference with the rotation of the helicopter stand's arm. Care is taken so that the wires do not interfere with the motion of the bat.

The helicopter stand includes an elevation counterweight located at the opposite end of the helicopter stand's arm. The position of this weight can be changed to simulate different effective weights of the robotic bat. This weight brings the elevation axis closer to neutral stability and would simulate the low weight (on the order of 30 grams) of the average bat [16]. To change the stability point of the robotic bat's pitch, a pitch counterweight is attached below the robotic bat. This pitch counterweight is added to the test bed by using perforated metal straps to attach it to a certain distance below the point of connection between the robotic bat and the helicopter stand's main rotating arm. The position of this pitch counterweight can be varied by moving the weight to different positions on the perforated metal straps, which would change the inherent longitudinal stability of the robotic bat. This allows us to test our control schemes on differing levels of natural stability. Counterweights were calibrated in order to bring the points of stability closer to neutral stability, with a slight amount of positive stability.

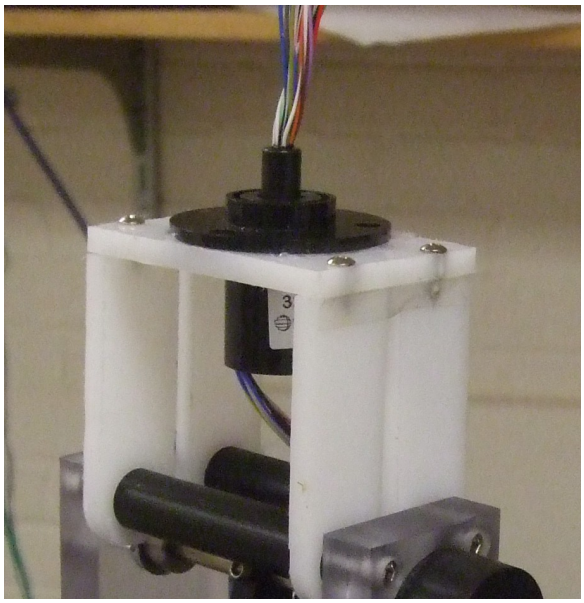
4.4 Controller Design

dSPACE, MATLAB, and Simulink are used to write the controller for the entire test bed. A Simulink model using dSPACE's own real time interface and blocksets was used in [1]. This model has since been improved to allow for better synchronization of motors to CPG signals, and more control over the phase differences between wing motions as well as other control parameters. To create a closed loop controller for the robotic bat with respect to its position and orientation on the Quanser 3DOF Helicopter stand, data must be exchanged between the dSPACE DS1104 and Quanser Q4 boards. In particular, the controller which is compiled

(a) Power Sources included with Quanser testbed

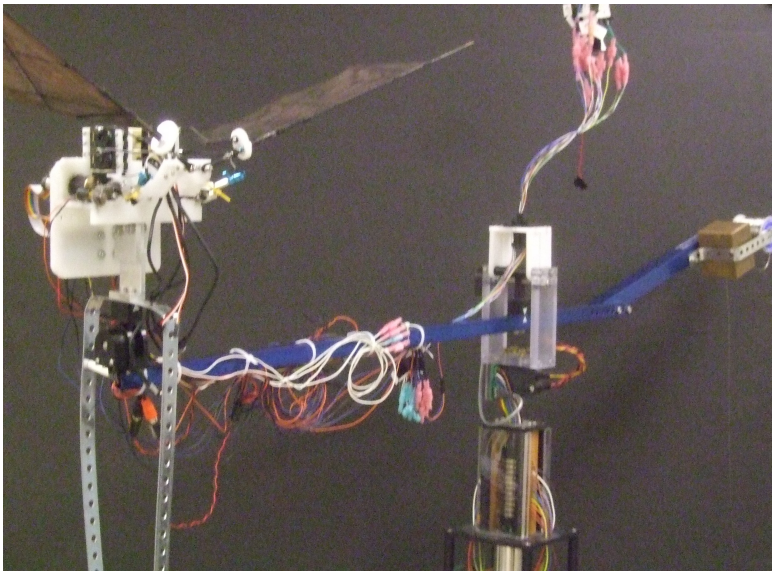


(b) Quanser Q4 board

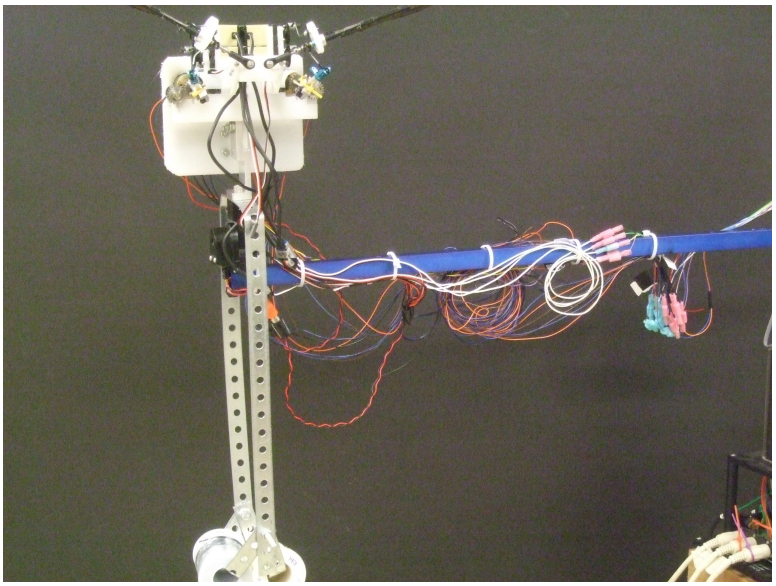


(c) Slip ring, mounted on top of stand and connected to dSPACE/computer and robotic bat

Figure 4.11: Components of test bed



(a) Robotic bat mounted on horizontal arm of Quanser 3DOF Helicopter stand



(b) Closer view of robotic bat, with pitch counterweight

Figure 4.12: View of test bed set up.

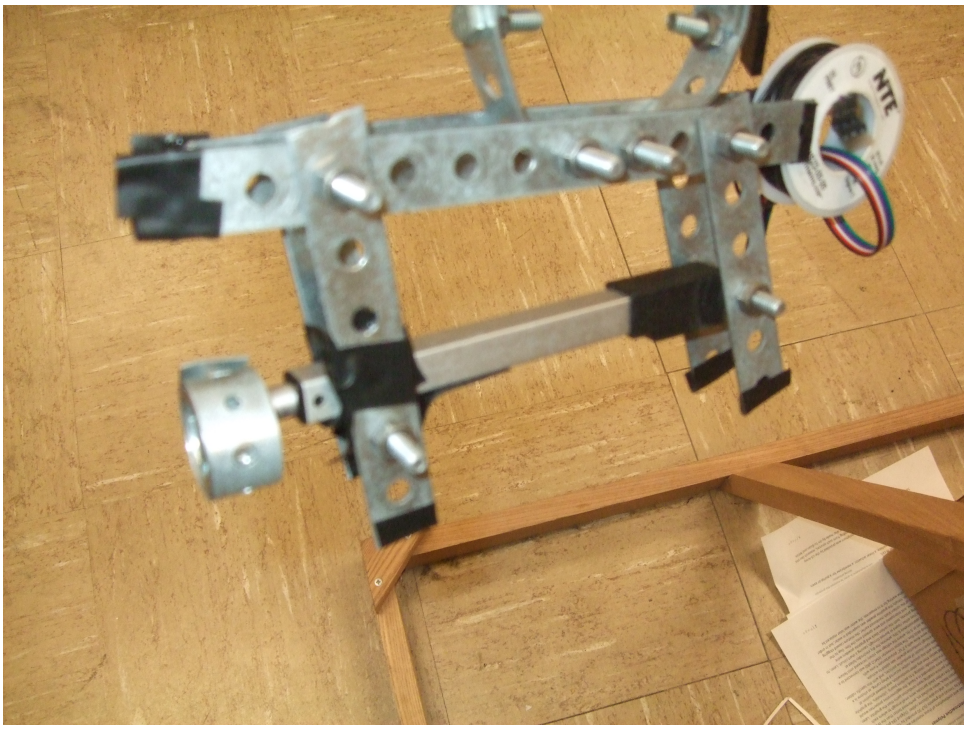
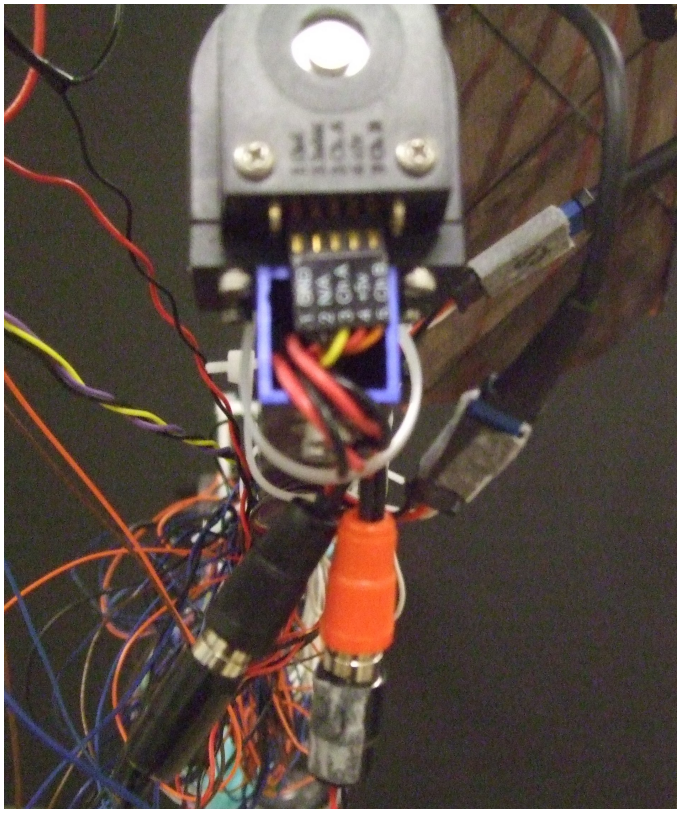


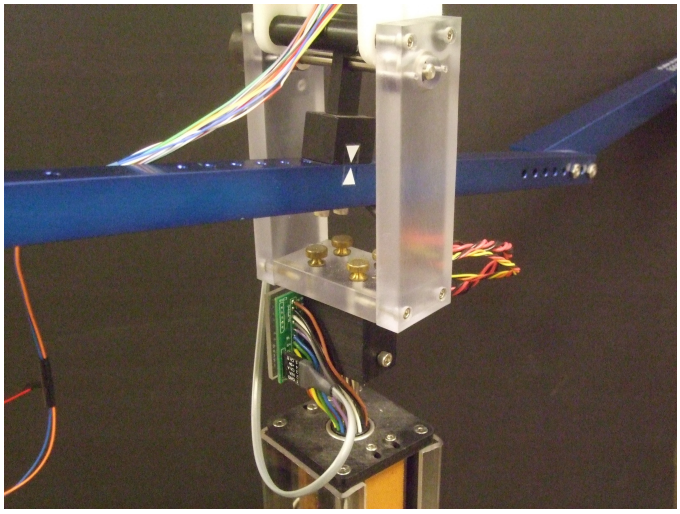
Figure 4.13: Linear actuator functioning as pitch counterweight

and run on the DS1104 board requires the encoder data from the Quanser Q4 board. Quanser also includes its own real time interface and blockset for Simulink, however it is currently not possible to compile a Simulink model containing both dSPACE and Quanser blocks due to how they are compiled to their respective real time processor boards. Therefore, a different approach was taken to interface both Quanser and dSPACE so that data could be exchanged between the two boards.

In order for the robotic bat to receive encoder data from the Quanser 3DOF Helicopter's encoders, we designed a software interface using dSPACE's MLIB and Quanser's Stream API. MLIB is a library of MATLAB functions which allowed for communication with the DS1104 board, and Quanser's Stream API contains MATLAB functions for data transfer between the computer and the Q4 board. Raw encoder count data was read into MATLAB, which then converted this data into radians and degrees, and wrote this data into the robotic bat's Simulink experiment file. The DS1104 board ran the Simulink experiment in real time and thus we could get feedback on the robotic bat's attitude and orientation in real time via



(a) Pitch encoder



(b) Elevation and travel encoders

Figure 4.14: Encoders on Quanser 3DOF helicopter stand

the encoders. This encoder data is used to develop a closed loop controller for the elevation, pitch, and travel angles and velocities for the bat. Ideally, the closed loop controller would use the encoder data as an input, and would change the phase differences between the pitch, flapping, and lead-lag motions as an output to control longitudinal modes.

The relatively high weight of the robotic bat, combined with the relatively high weight of the pitch counterweight on the opposite end of the pitching arm, create a considerable moment of inertia. Because of this moment of inertia, even a small amount of pitch variation due to the modulation of phase differences shows a significant effect with regards to maintaining longitudinal stability.

4.5 VICON Motion Capture System

To analyze the exact motions of the robotic bat's wings, the Vicon Motion Capture system was used. This allows us to characterize the profile of the wings and use this data to improve our controller a priori using kinematic data. It is also possible to characterize the robotic bat's flight trajectory using a motion capture system, however for simplicity's sake we prefer to use the encoders already built in to the helicopter stand.

Special retro-reflective markers are placed on the bat's wings and body. Several infrared cameras are arranged around the robotic bat test bed, and use triangulation to record the position and orientation of these markers down to sub-millimeter accuracy. The position of these markers is used to calculate the rotations of the wings and body. From this wing angle data, the phase differences between the flapping, lead-lag, and pitch motions can be calculated. Rotations of the body are measured so that the rotation of the wings with respect to the body can be calculated while accounting for the body's own slight rotations due to its placement on the pendulum. The system is capable of recording data from anywhere between 100 and 200 Hz.

The markers on the robotic bat are seen by the cameras, and the data is read by Vicon

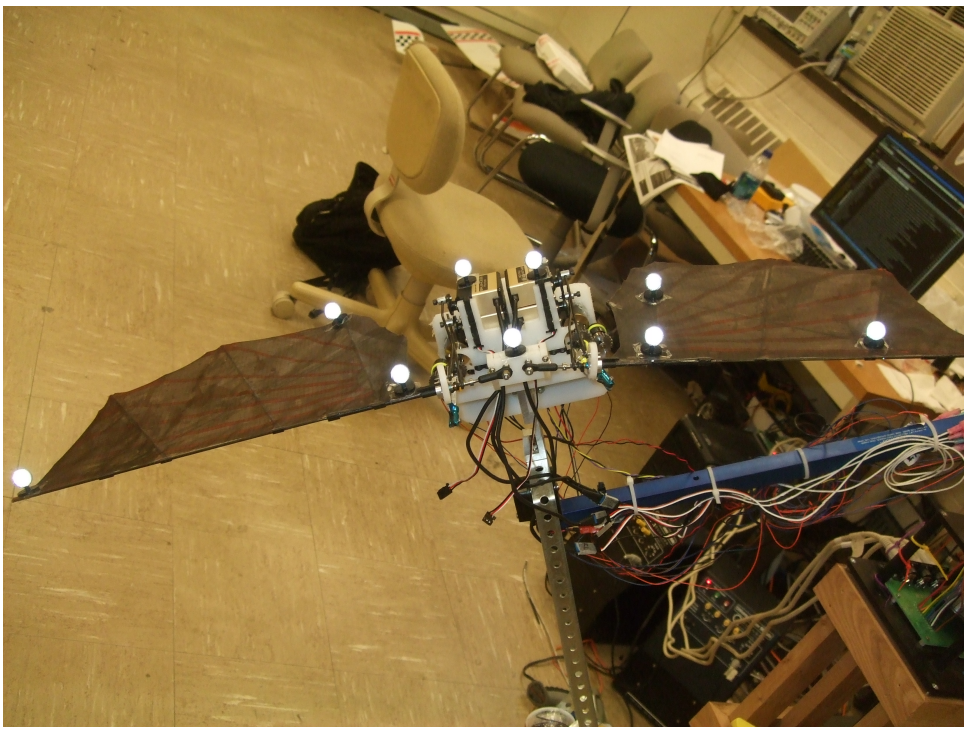


Figure 4.15: Robotic bat fitted with markers

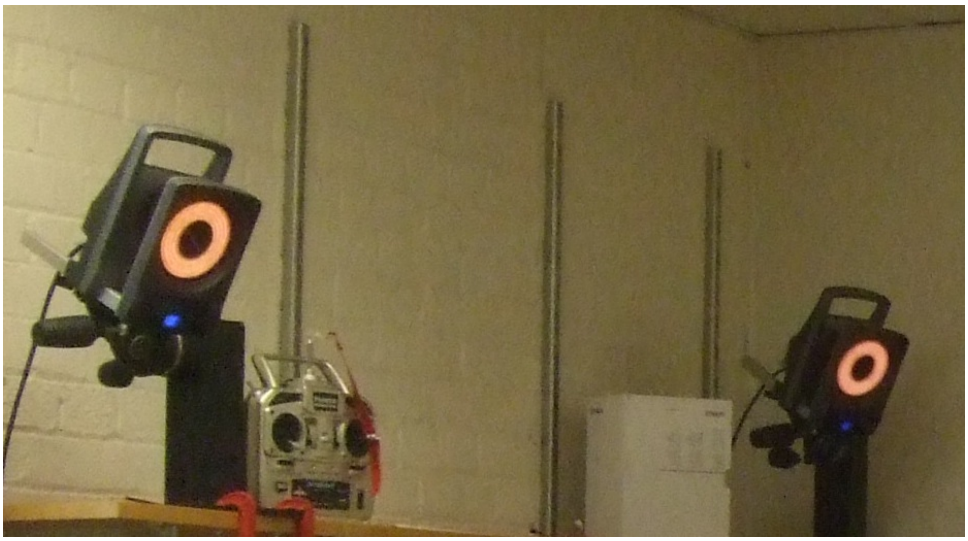


Figure 4.16: Vicon cameras

Figure 4.17: Vicon cameras setup on tripods

Figure 4.18: Vicon cameras around robotic bat

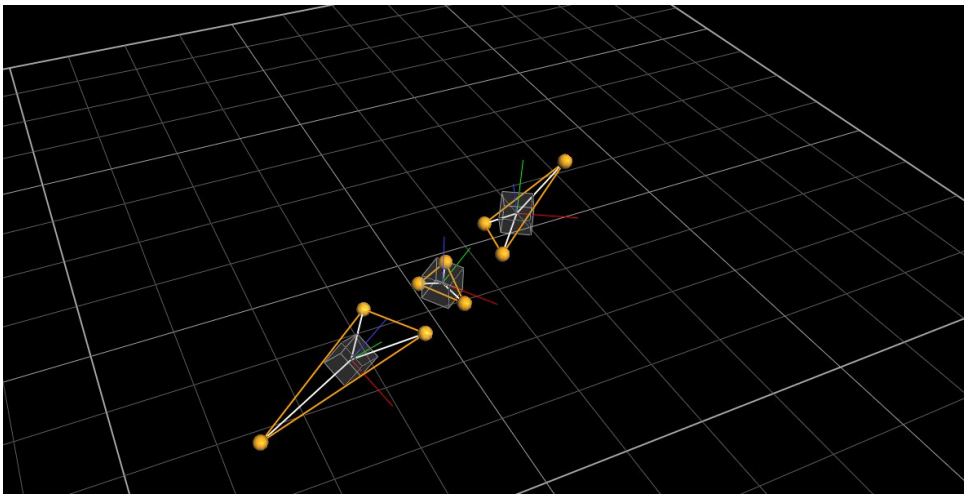


Figure 4.19: Robotic bat, as seen by Vicon Tracker

Tracker: software which allows us to create objects from this marker data. The wings and body of the robotic bat are seen as three individual objects by Vicon Tracker; 4.19 shows what Vicon Tracker sees as a result of the marker placement and object creation. The Vicon Motion Capture System includes a software development kit (SDK) which allows us to create programs which use data from the Vicon system. We use the included MATLAB SDK, which allows us to extract position and orientation data of the robotic bat's wings in real time. This data is used in post processing for calculating and analyzing the kinematic data of the robotic bat's wings.

Chapter 5

Experimental Setup and Results with Robotic Bat

5.1 Introduction

Major experiments conducted using the robotic bat test bed are shown in this chapter, along with results and interpretations.

5.2 Open Loop Control Experiments

To characterize the robotic bat's responses to certain inputs and design a controller based on those responses, open loop experiments were conducted. They have both focused on steady-state behavior of pitch, elevation, and travel velocity with respect to phase differences. Experiments are conducted by commanding flapping frequency and phase differences while observing the bat's orientation and velocities from the encoder data. Using dSPACE's ControlDesk software, a GUI is created for direct interaction with the real time controller of the bat, which is compiled and run on the dSPACE real time computer. Control variables can be changed and analyzed in real time, and data is captured and saved to a MATLAB data file.

As mentioned in chapter 4, there is an offset between the center of the bat and the pitch rotational point. This creates a coupling between the dynamics of the second pendulum with the longitudinal dynamics of the bat. While we use this to our advantage to obtain stability states desired for testing phase difference control, it necessarily creates a large rotational moment of inertia that is many times that of an actual bat. Therefore, we expect the

pitching moments from phase difference control to have less effect in this experimentation than in free flight of a low moment of inertia bat. Regardless, we can see pitch control via only flapping/lead-lag phase difference even in this set-up.

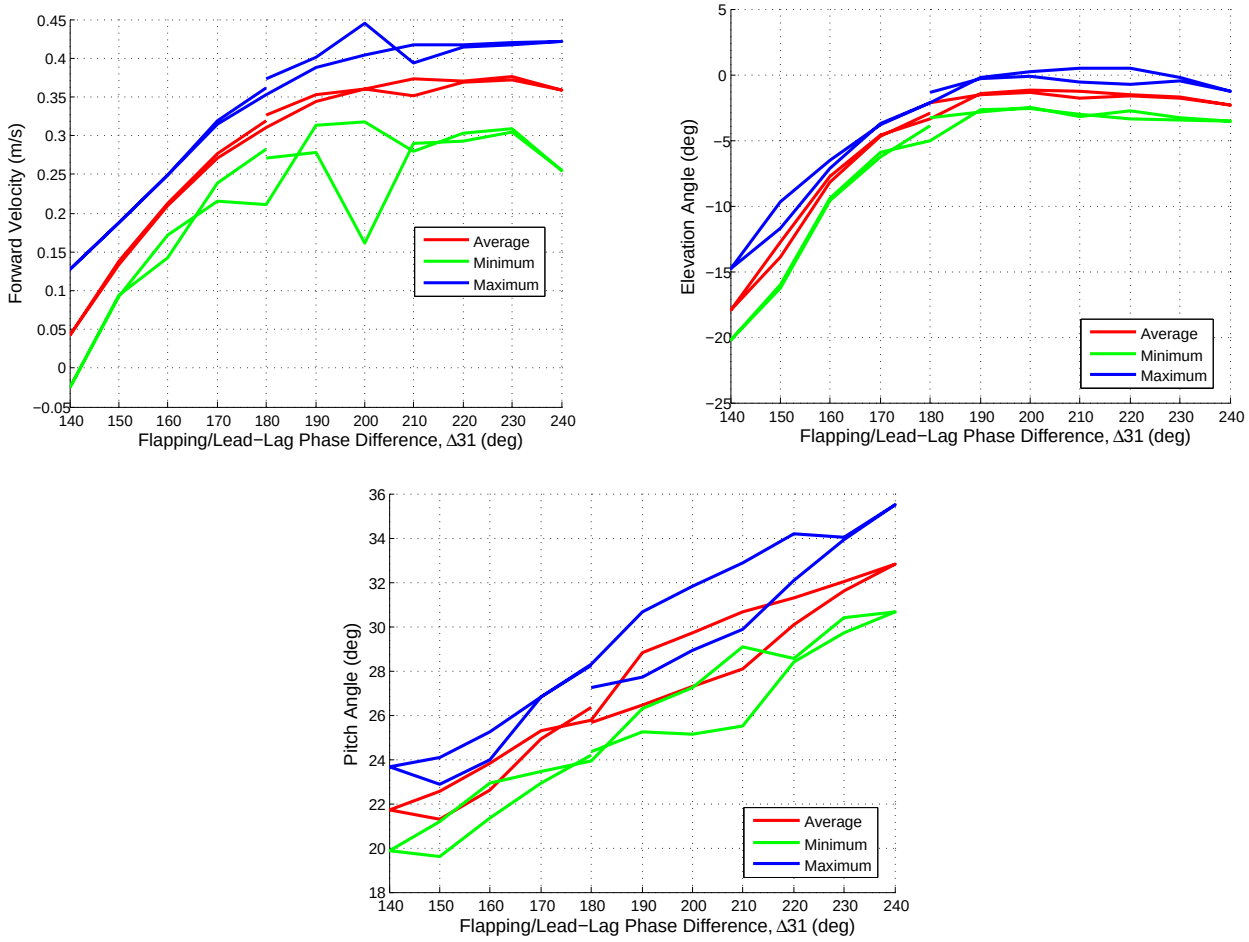


Figure 5.1: Open-loop pitch control via phase differences (2 Hz).

In the first open loop experiment, the phase difference between flapping and lead-lag, Δ_{31}, was varied between 140 deg and 240 deg twice. The system was allowed to converge to a non-equilibrium steady state. Encoder data was captured for 20 seconds. Figure 5.1 shows the minimum, maximum, and average value over the 20 second period. As expected, between 180 and 240 deg, the forward velocity and elevation curves look very flat, but the pitch angle increased between 6 and 8 degrees. This corresponds with the idea that lift and thrust generation remained similar while only a control moment was created. It is

postulated that a free flight system with a low pitching moment of inertia would see much stronger pitching effects from such control. This is supported by physical intuition and the numerical simulations performed previously [1]. The lower range of phase differences, 140-180 deg, saw a large dropoff of thrust and lift generation. In fact, the bat came to a complete stop at one point with the phase difference at 140 deg. Therefore, we should not plan to use this range in control of flapping flight.

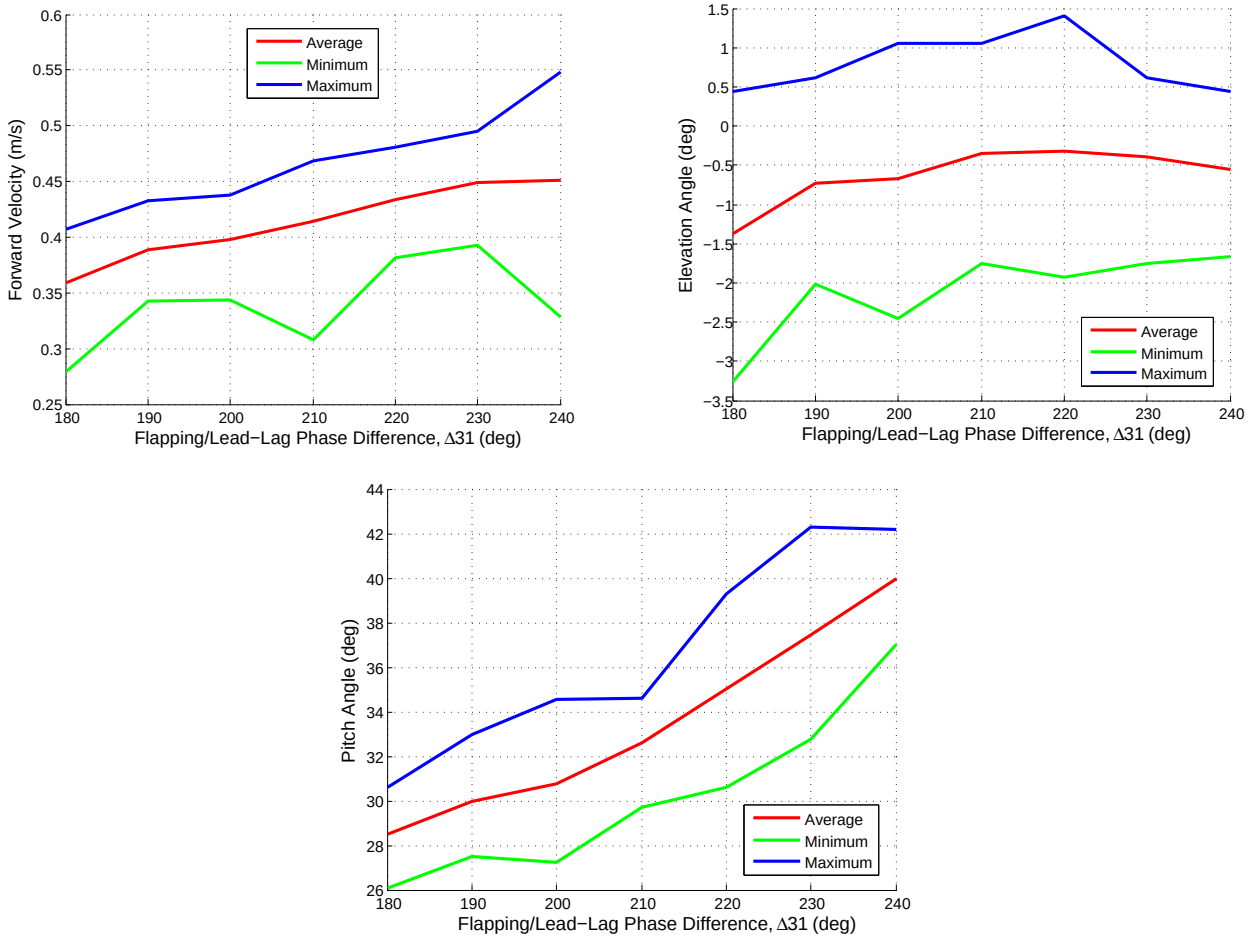


Figure 5.2: Open-loop pitch control via phase differences (2.5 Hz).

The second experiment repeated the same process at 2.5 Hz instead of 2 Hz. Only one sweep through the values was performed. The results are plotted in figure 5.2 and support the same conclusion as the first experiment. Additionally, they preliminarily confirm the postulate that flapping frequency can be used as strong control of forward velocity and

elevation. Finally, note that the author is not concerned about the fact that all the relevant body pitch angles are all around 20-40 degrees. Adjustment of the center of gravity location with the pitch counter weight can set the trim state as desired while control moments are created from phase differences.

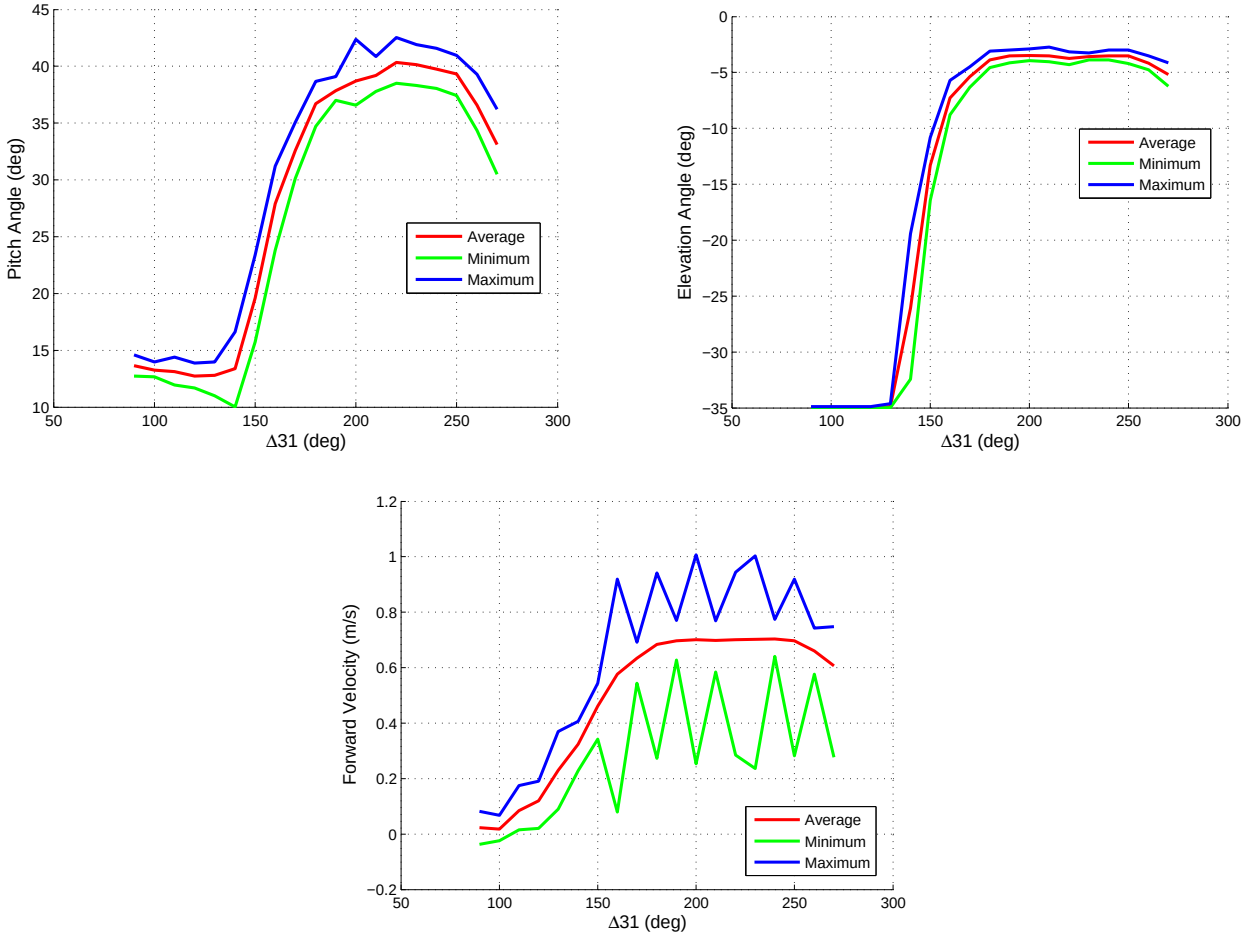


Figure 5.3: Open-loop pitch control via phase differences (3 Hz).

Some time after the open loop experiments at 2 Hz and 2.5 Hz were conducted for [40], some more open loop experiments were conducted after some mechanical modifications were made to the testbed. These modifications included higher beating frequency capabilities (as high as 5 Hz or more), and the addition of a linear actuator which allows us to automatically vary the fixed point of stability with the assistance of an RC controller. The set of open loop experiments were run at 3 Hz, and only one set of data was taken since these experiments

were meant only as preparation for more in-depth closed loop experiments. The results of this open loop experiment are plotted in figure 5.3. Again, it is shown that for a certain limited regime of phase differences there is control of elevation and velocity, while pitch control authority is maintained over a wider regime of phase differences. It is important to note that the position of the pitch counterweight is relevant insofar as the center of gravity position in the pitch direction; a change of the center of gravity position changes the behavior of the system due to the change in moment about the pitch direction. Also, the use of a linear actuator increases the rotational moment of inertia of the system due to both the weight of the linear actuator itself, the structure which holds the linear actuator to the setup, and the weight that the linear actuator moves. The moment of inertia was already artificially high to begin with due to the use of counterweights in the pitch and elevation directions. Regardless, the high moment of inertia should serve to only strengthen the results obtained from experimentation since we are able to show longitudinal control, and further the idea that control authority would be far greater in a low inertia system similar to those found in bats in flight.

5.3 Closed Loop Control Experiments

Previous numerical results have shown that, for longitudinal modes, dimensional reduction via CPGs can be effective [1]. It was shown that control could be reduced to just two parameters: frequency (corresponding with velocity) and the phase difference between flapping and lead-lag (Δ_{31} corresponding with pitch state). The open loop non-equilibrium steady state experiments have supported this idea further [40]. These closed loop experiments conducted intend to show how the CPG structure allows very simple top-level controllers to provide stability and control in closed loop. These PID controllers

Very simple symmetric PID controllers were used for all experiments,

$$\Delta_{31} = \Delta_{75} = -K_p(p - p_d) - K_d\dot{p} - K_i \int_0^t (p - p_d)\, dt, \tag{5.1}$$

where $\dot{p}$ is computed using the second order derivative filter, $\frac{\omega_{cf}^2 s}{s^2+2\zeta_f\omega_{cf}s+\omega_{cf}^2}$, with $\omega_{cf} = 40\pi$ and $\zeta_f = 0.9$. The values of this filter can be changed to reduce the amount of noise and phase lag that is inherent to taking the derivative of a signal. The PID gains were tuned manually in order to obtain a satisfiable system response, and to prevent integral windup we used state saturators on the integrators. Saturation values were set so $\Delta 31 \in [180°, 270°]$. Even though we are able to use simple PID controllers in the top level, the overall controller is very nonlinear due to the CPGs described in Equation (2.2).

We begin experimentation at an open loop frequency of 2.5 Hz. At this frequency, the open loop appeared stable. Figure 5.4 shows the response to a change in desired body pitch from $-10°$ to $-20°$. Two notes from [40] should be kept in mind. First, the actual value of body pitch is affected by the precise position of the pitch counterweight. Therefore, it is not worrisome that the values are not exactly around zero or some other intuitively desired value; in fact simple open loop experiments show that in order for the robotic bat to maintain sufficient elevation and velocity a nonzero pitch value tends to be desirable. Second, at 2.5 Hz, the apparent maximum change of body pitch due to open loop control of Δ_{31} is around $10 - 12°$. This experiment demands a change of $10°$ and experiences saturation problems as it nears the final desired state.

Moving the frequency to 3 Hz caused instability in the open loop. Figure 5.5 shows that by activating the PID control of Δ_{31}, we can stabilize the unstable system. At this frequency, we also have appreciably more control authority. Figure 5.6 shows a commanded pitch change of $15°$, which is easily obtained. We expect that at speeds typical of bat flight (2-3 m/s with frequencies of 7-10 Hz [16]) and pitch moment of inertias not inflated by the pendulum setup we will see even more control effectiveness. Numerical results have

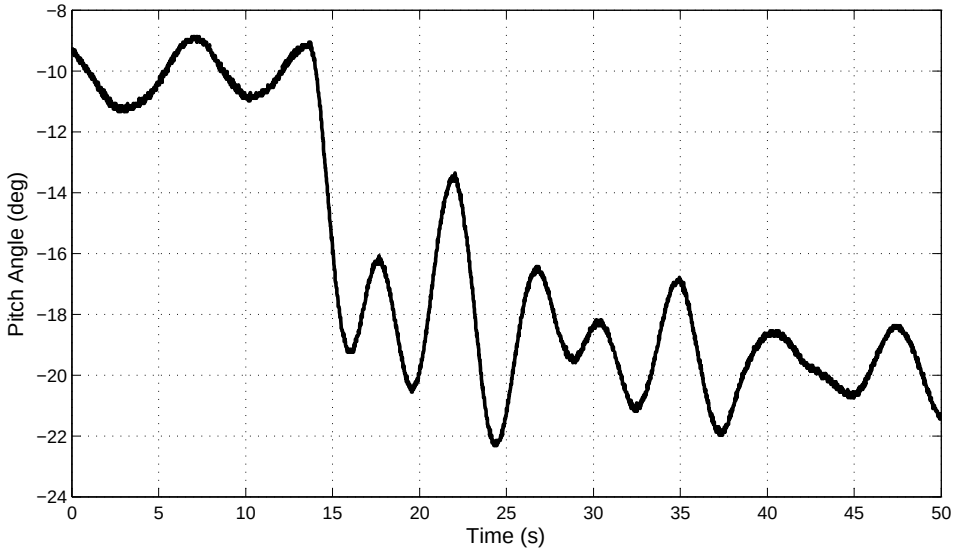


Figure 5.4: Experimental Results of Pitch Control at 2.5 Hz

supported the idea that this control effectiveness will be much higher [1].

In addition to controlling the pitch angle of the robotic bat, we would like to feedback velocity into flapping frequency. Preliminary results are shown at a nominal frequency of 3 Hz in Figures 5.7 and 5.8. The frequency PID controller,

$$\omega(t) = \ell_a \left(-K_p(\dot{\lambda} - \dot{\lambda_d}) - K_d\ddot{\lambda} - K_i \left(\lambda - \lambda_d\right) \right), \tag{5.2}$$

where $\dot{\lambda}$ is the travel velocity and with the same derivative filter and saturation values $\omega \in [1, 5]$, was activated at approximately 7 seconds. It initially causes a great change in frequency, which initiates a large oscillation in pitch mode. The Δ_{31} controller saturates, but is able to slowly damp out the oscillation. The frequency controller's integrator also saturates, preventing the velocity from reaching the desired 0.5 m/s. Also note that the filter originally used had a cutoff frequency of $\omega_{cf} = 40\pi$, or 20 Hz which was later determined to allow too much noise into the derivative calculation. The discrete nature of the encoders warrants the use of a low pass filter when numerically calculating the derivative of a signal, and thus setting this cutoff frequency to a much lower value on the order of 1 Hz or less would allow a more cleaner calculation of the velocity and acceleration of a direction in real time by

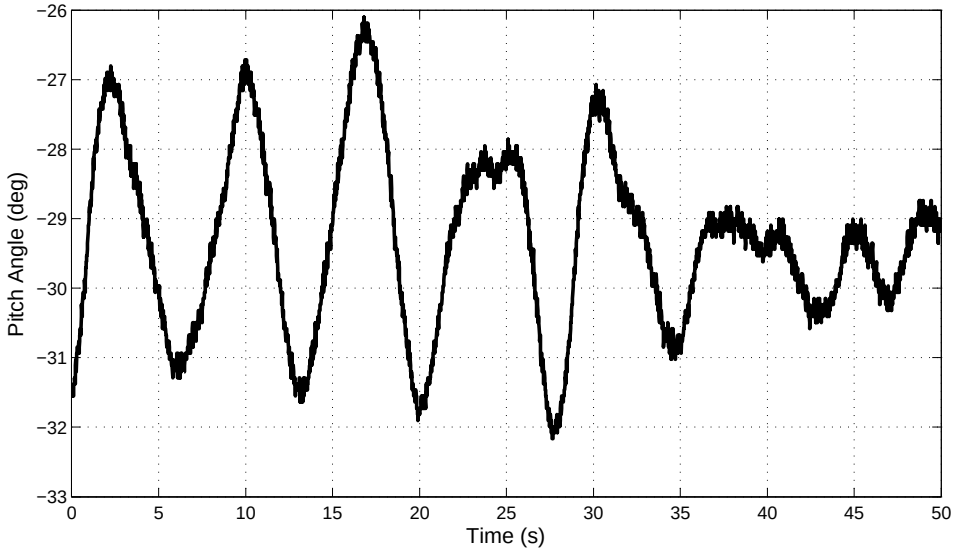


Figure 5.5: Experimental Results of Pitch Stability by Control at 3Hz

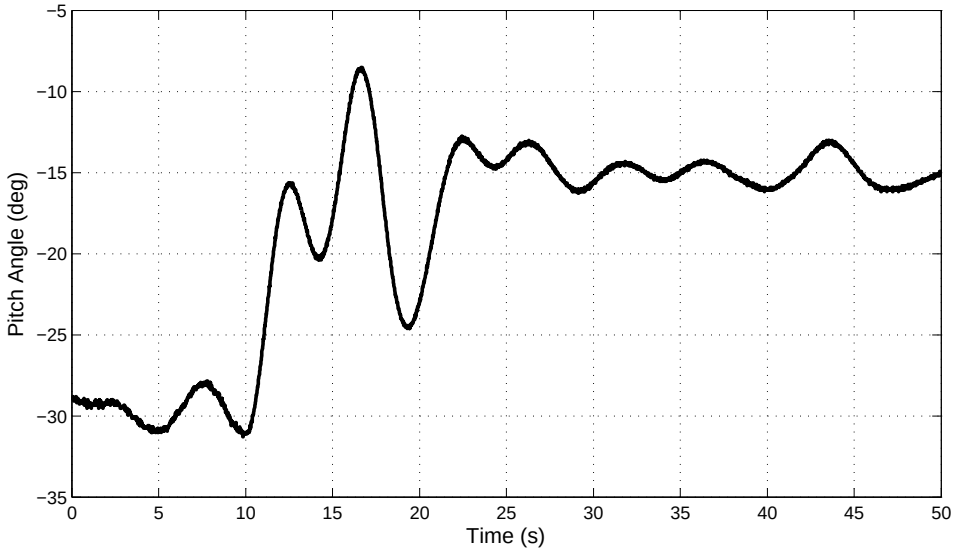


Figure 5.6: Experimental Results of Pitch Control at 3Hz

reducing the magnitude of noisy data. However, the cutoff frequency and dampening factor of a filter must be chosen with care because a filter will inevitably induce some phase lag into the calculations; however subsequent tests revealed this phase lag did not have any major adverse effects on data capture. When derivative data is not needed online, it is preferable to use a post processing low pass filter of data instead to calculate derivatives of signals.

More pitch control is shown in figure 5.9, with the desired pitch value superimposed

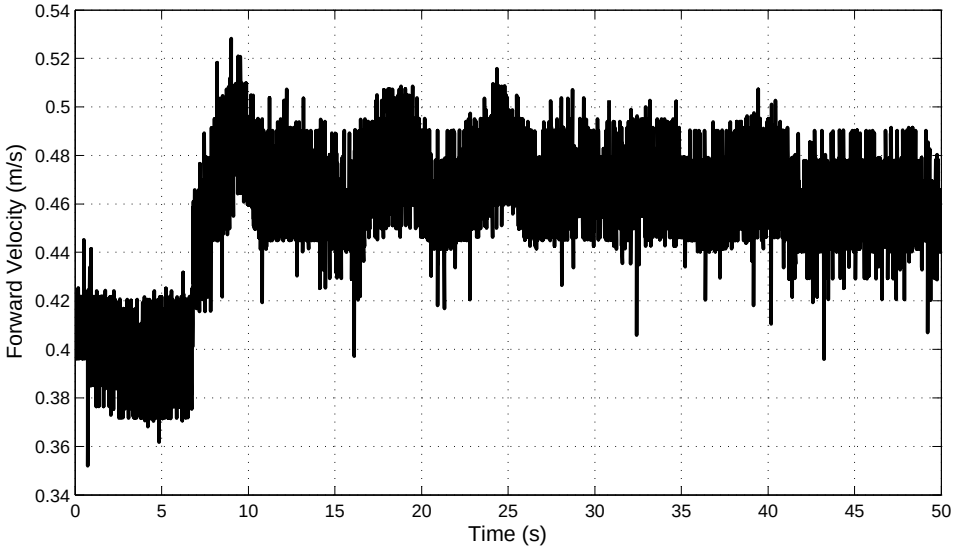


Figure 5.7: Velocity Control Tracking

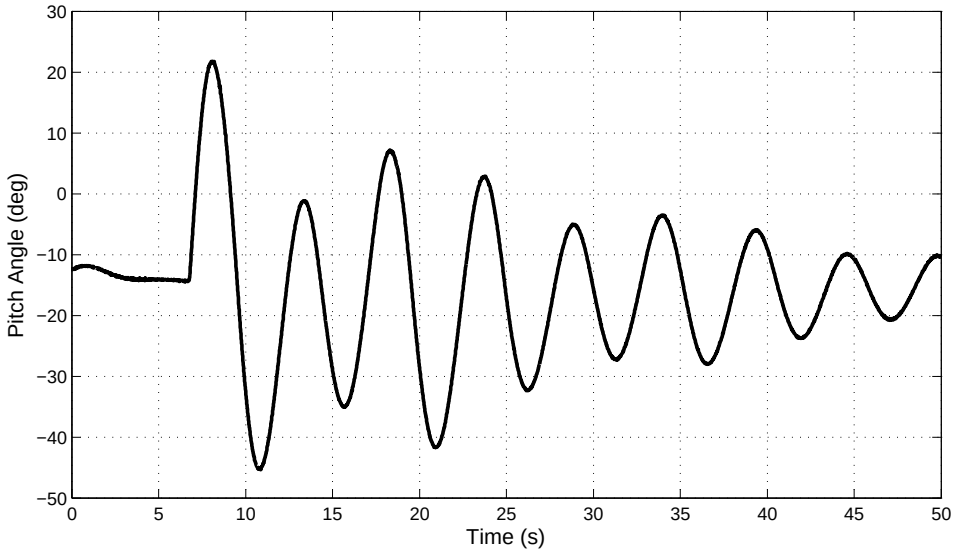


Figure 5.8: Pitch Stability after Large Perturbation

over the measured pitch value, and the corresponding control input (Δ_{31}). The experiment begins with pitch control switched off, then at approximately t=10 seconds the pitch control is switched on and the pitch soon increases by 10 degrees to its desired pitch value of 35 degrees.

Figures 5.10 and 5.11 show a change in desired pitch (both positive and negative) and the robotic bat's response to such changes. In figure 5.10 the robotic bat is settled near a

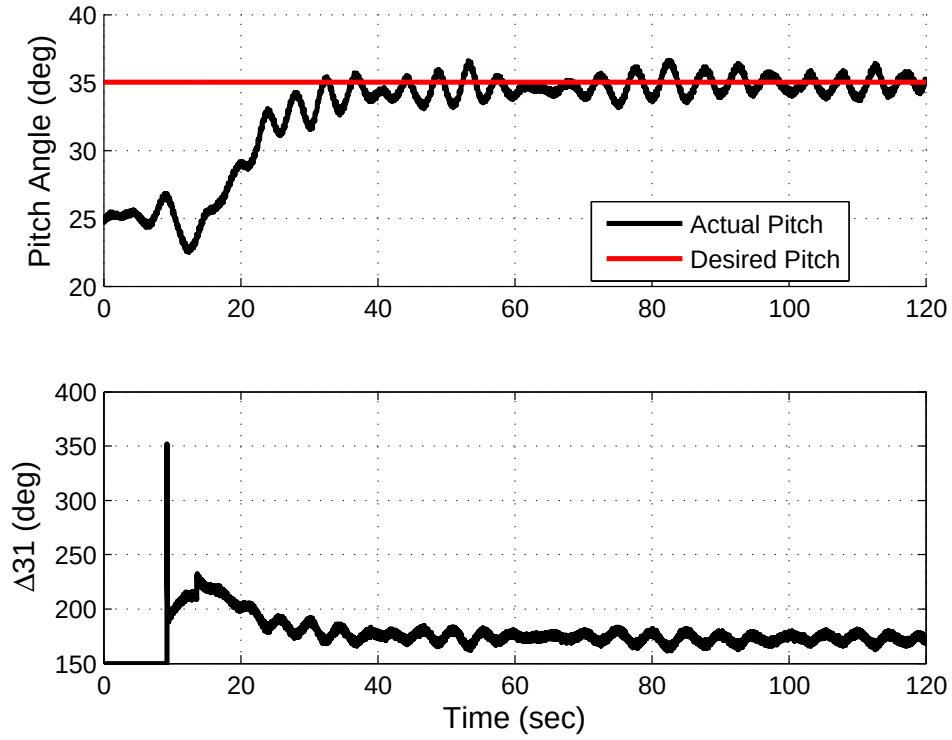


Figure 5.9: Pitch control with corresponding control input

desired pitch value of 25 degrees with the pitch controller running, and then at t=60 seconds the desired value is increased all the way up to 40 degrees. In Fig. 5.11 the desired pitch is changed from 45 degrees to 30 degrees at approximately t=35 seconds. Note that after the desired pitch is decreased, the Δ_{31} controller reaches its lower bound saturation value of 90 degrees momentarily before settling close to its desired value.

In figure 5.12, the robotic bat is flying at steady state with a desired pitch of approximately 15 degrees. To test the response of our controller to large perturbations, the robotic bat was manually perturbed in the pitch direction by over 60 degrees. This caused the Δ_{31} value of our controller to reach saturation momentarily, however the large oscillations are eventually dampened out and the robotic bat returns to its original desired pitch position.

More experiments were conducted to control the travel velocity by modulating the flapping frequency. Figure 5.13 shows the robotic bat initially at 0.4 m/s with the velocity controller switched on, then at approximately $t = 7$ seconds the desired velocity is increased

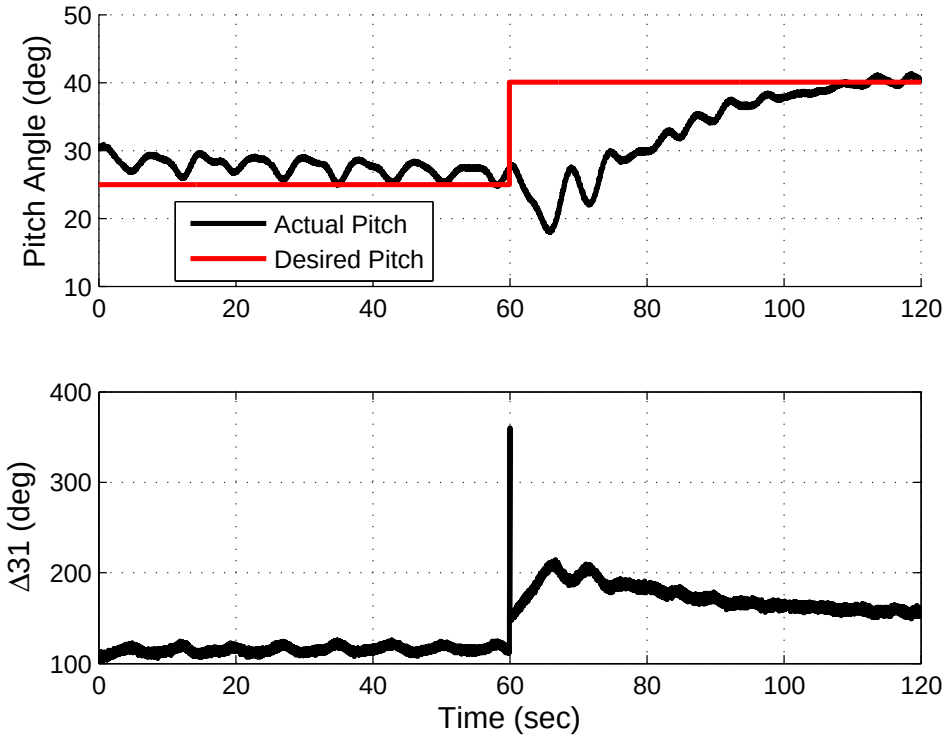


Figure 5.10: Pitch control tracking with corresponding control input

to 0.5 m/s. The robotic bat's response and the accompanying control input are shown. Figure 5.14 shows a velocity control tracking with decreasing desired velocity, with the robotic bat initially at 0.5 m/s with the velocity controller switched on, then at approximately $t = 12$ seconds the desired velocity is decreased to 0.4 m/s.

Finally, we could like to implement a combination of pitch and velocity control. This is accomplished by modulating both the flapping frequency and the lead-lag phase difference with the PID controllers described in equations (5.1) and (5.2). In figure 5.15 we show we can change the desired pitch value from 30 degrees to 35 degrees, while simultaneously reaching a desired velocity of near 0.5 m/s.

However, we can not simply control the velocity and pitch independently by setting either to an arbitrarily desired value. Intuitively, a change in forward velocity would create a pitching moment; moreover a change in pitching moment would redirect the aerodynamic forces generated by the robotic bat's wings and would potentially affect the forward velocity.

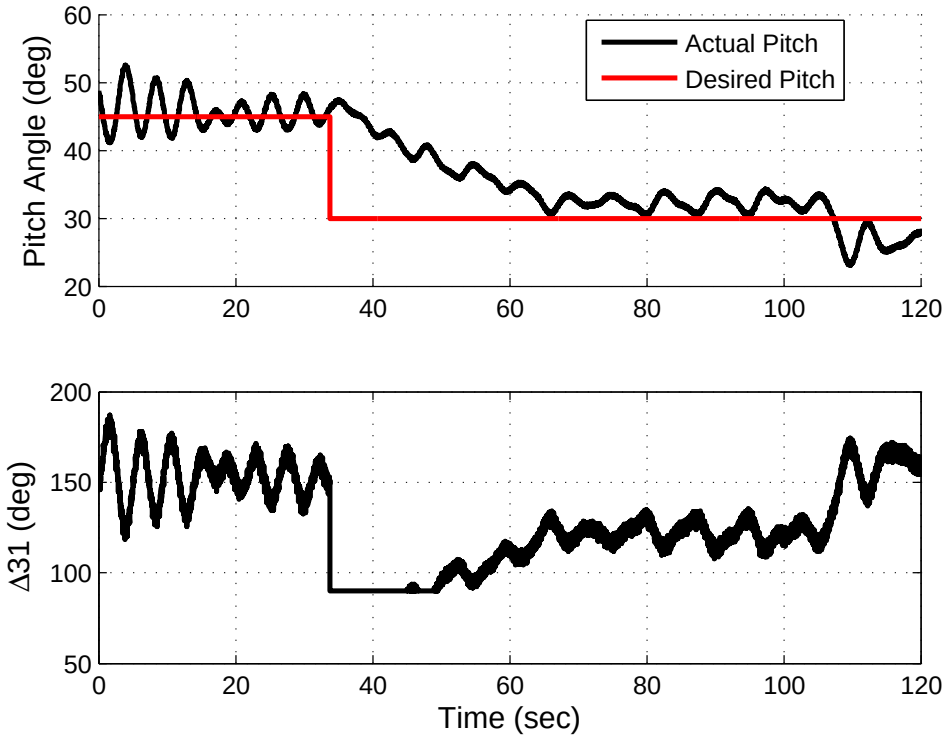


Figure 5.11: Pitch control tracking with corresponding control input

We demonstrate this relation in figure 5.16. In the beginning of the experiment, note that both the measured velocity and pitch values are significantly below their desired values. The control inputs, frequency and phase difference, are high because the actual values had not yet converged to their desired values. As mentioned in the open loop control experiments in 5.2, there are phase differences which reduce the robotic bat's performance and may reduce forward velocity to zero; since the pitching motion is controlled by changing phase differences by extension there are pitch values which reduce the robotic bat's performance as well. This shows that the dynamics of the robotic bat do not allow for the robotic bat to fly at the desired velocity and pitch value; at $t = 4$ seconds the desired pitch value was decreased by 10 degrees, and almost immediately we see the robotic bat's velocity and pitch values begin to increase to its desired value. In figure 5.17 we show the reverse of this relation as well. Initially the pitch value remains seemingly steady with steady state error, and the velocity is in fact decreasing, which causes the flapping frequency controller to increase the

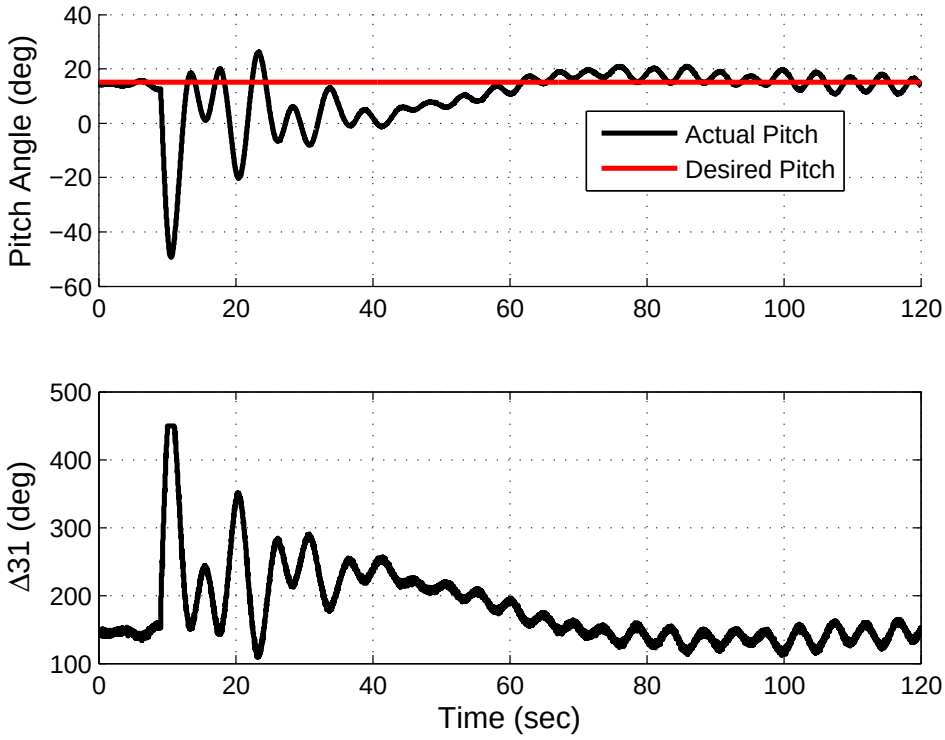


Figure 5.12: Pitch control response to large perturbation

flapping frequency. When the desired velocity value is increased from 0.4 m/s to 0.5 m/s, the pitching motion responds and converges to its desired value of near 40 degrees while the velocity increases to its desired value. More work will need to be done to characterize the performance limits of the robotic bat regarding longitudinal motion and forward velocity.

5.4 Glide Transition

While most work done for this thesis involved flapping flight experimentation, it is also possible to incorporate gliding flight into our model as well. Gliding is usually more common with larger flying creatures such as the albatross, however even smaller flying creatures like bats seem to glide if only for a brief period of time for flying maneuvers such as perching. Moreover, the Hopf oscillators used to represent CPGs are capable of converging to a fixed point when the bifurcation parameter σ is switched from $+1$ to -1, as shown in [1].

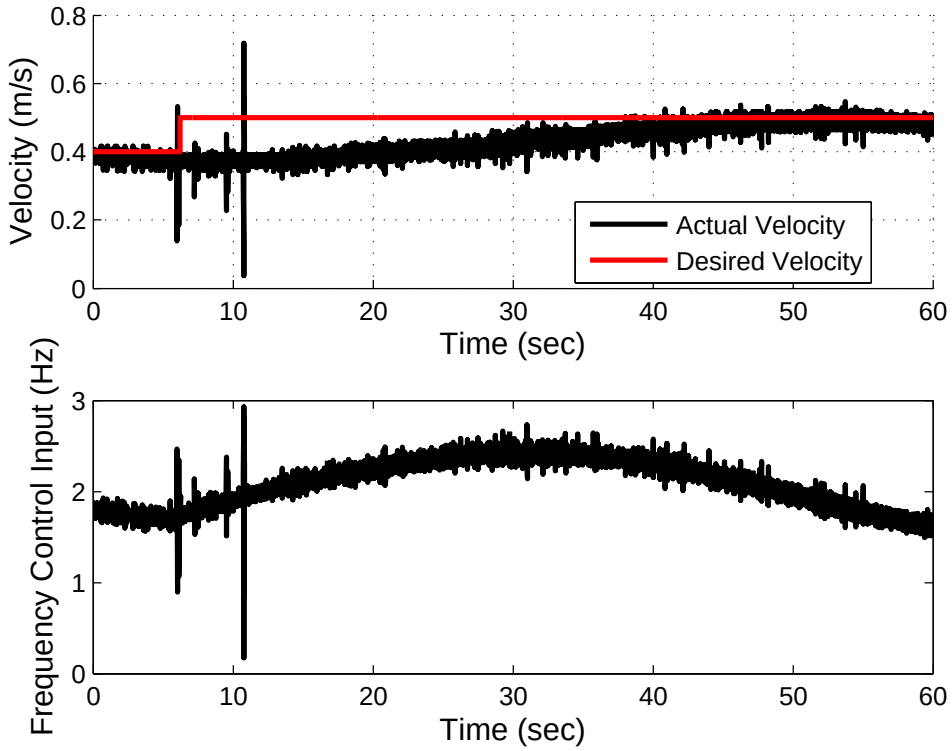


Figure 5.13: Velocity control tracking with corresponding control input

Gliding flight is activated by manually switching the bifurcation parameter σ from $+1$ to -1. In figure 5.18 the open-loop response to a switch from flapping flight to glide flight is shown, with gliding flight activated at around $t = 8$ seconds. When gliding mode is activated, the travel velocity begins steadily decreasing, the pitch is momentarily excited by the sudden change and then settles to a constant value, and elevation decreases with some very slight, very dampened oscillations.

Since the CPGs converge to a globally stable fixed point when σ is negative, we can adjust the biases of the CPGs to change the location of the fixed point of convergence and by extension, change the biases of the wing motions. In figure 5.19, after the robotic bat is flapping, at about $t = 22$ seconds gliding mode is activated with the flapping CPGs at a bias of 1; this corresponds to the wings at the peak of upstroke during flapping. To imitate the a perching maneuver as described in [7], at $t = 25$ seconds the flapping bias is reduced to zero and the lead-lag and pitching biases (not pictured) were set to allow maximum angle of attack

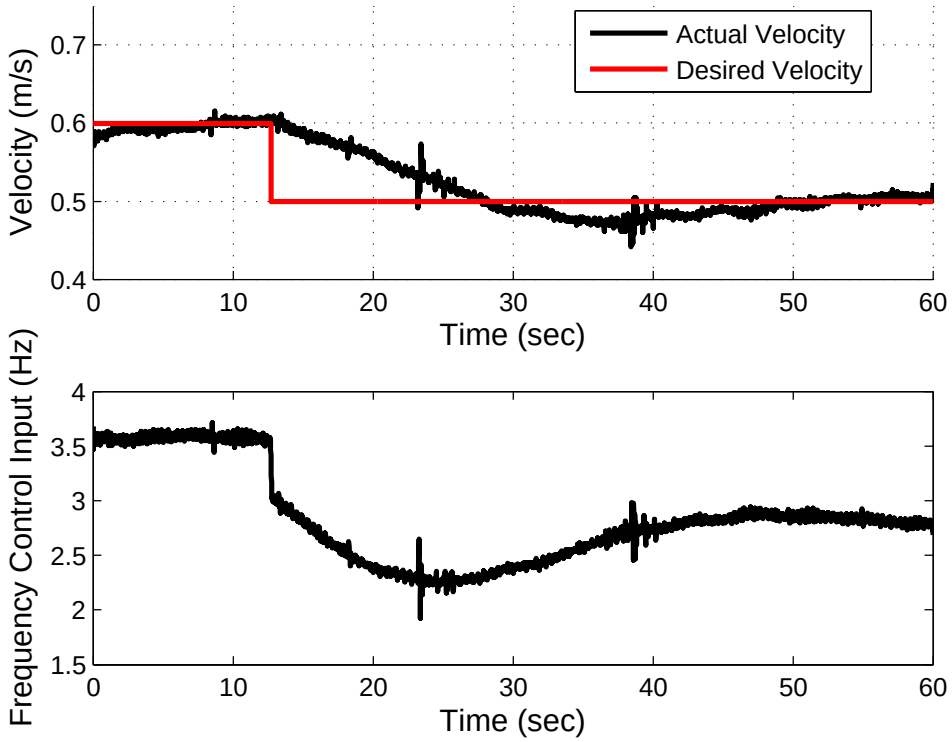


Figure 5.14: Velocity control tracking with corresponding control input

with the bat's wings. Unfortunately, the moment of inertia of the system is far too high to show any major observable effects, however the attempted perching maneuver appeared to increase the pitching angle momentarily and the elevation angle appeared to increase momentarily as well. Figure 5.20 shows the calculated gliding flight path angle using the travel and elevation velocities. Again, a high moment of inertia made any major observable changes in flight path angle quite difficult to discern; however the slow oscillations in the flight path angle appear to increase slightly at the time of the perching maneuver before dampening and dissipating.

In figure 5.21, we show a flapping to glide transition, and back to flapping. When the robotic bat is initially in flapping flight, the velocity and pitch controllers are active with desired values of 0.5 m/s travel velocity and 40 degrees pitch respectively. At $t = 15$ seconds, gliding motion is activated however the velocity and pitch controllers are kept active as indicated by their respective plotted control inputs; these control inputs are not

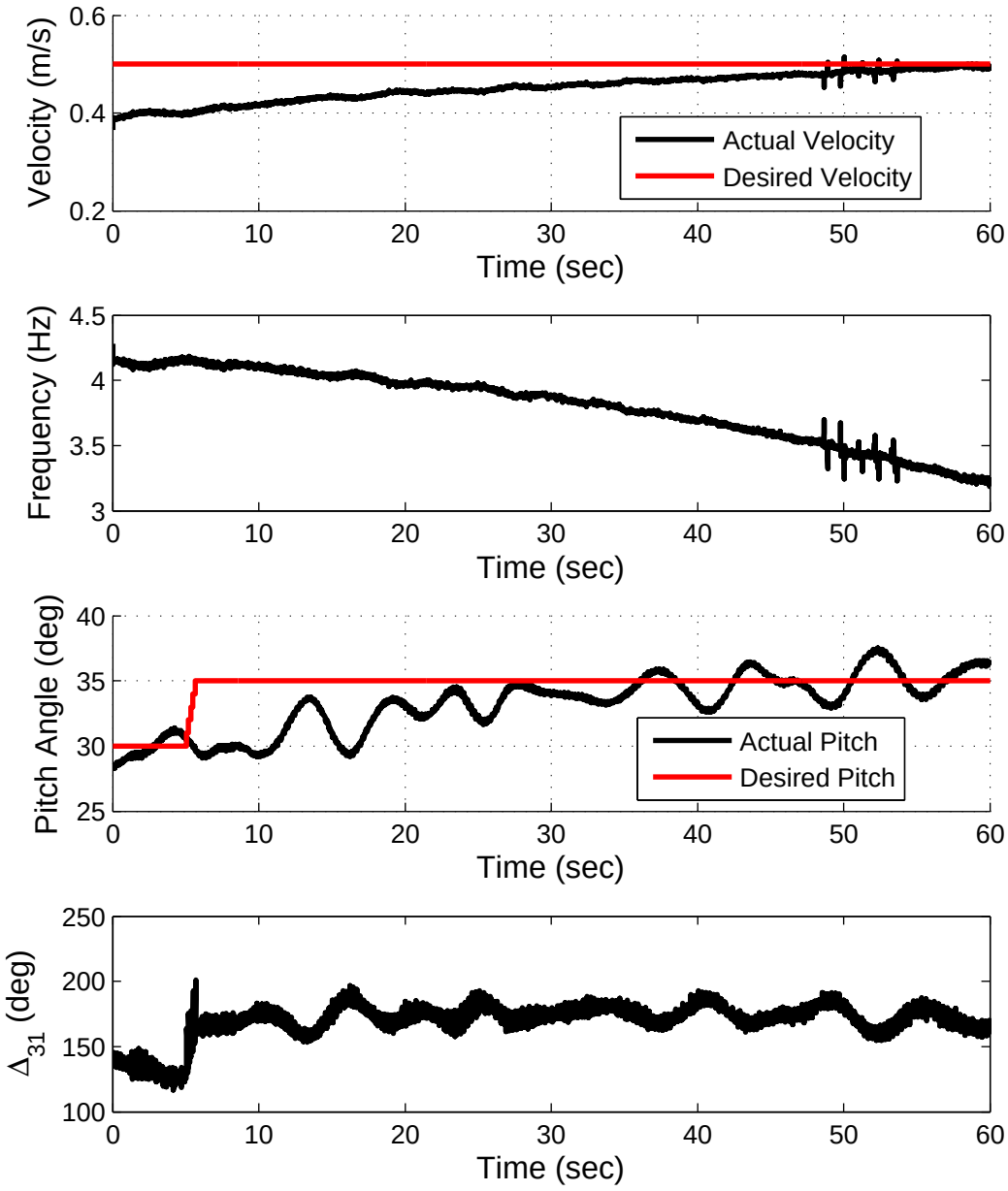


Figure 5.15: Pitch tracking with active velocity control

relevant during the gliding portion of flight due to the rapid inhibition of oscillations due to the Hopf oscillator based CPGs. As the energy from gliding dissipates and velocity and elevation approach a minimum, around $t = 35$ seconds the mode of flight is switched back to flapping. The CPGs soon converge back to their oscillatory limit cycle and flapping flight soon resumes. After a rise time due to the high moments of inertia in both the pitch and

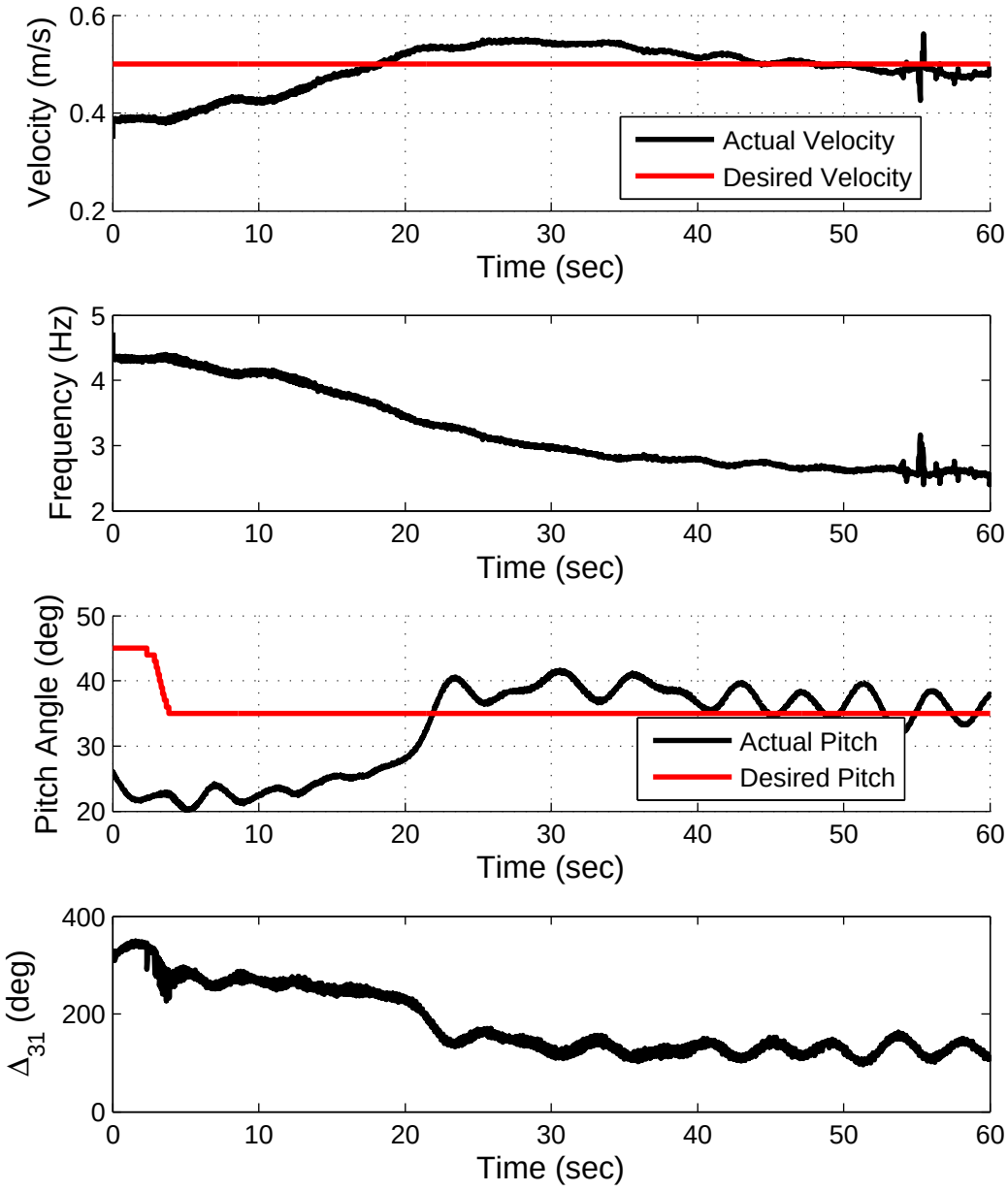


Figure 5.16: Change in desired pitch results in velocity tracking

elevation directions, the robotic bat returns to its previous desired velocity and pitch values.

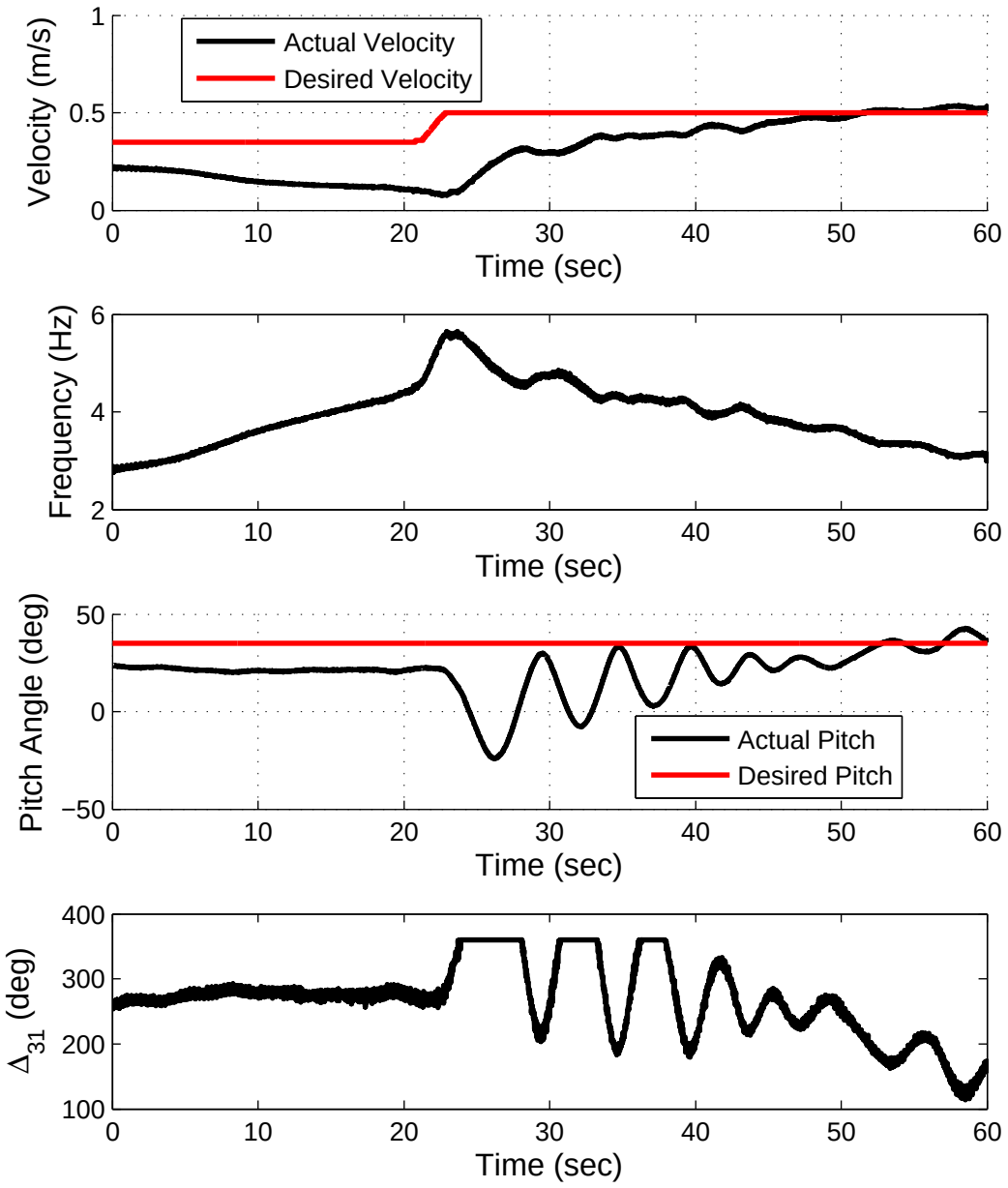


Figure 5.17: Change in desired velocity results in pitch tracking

5.5 Force and Moment Calculations

Referring back to the equations of motion described in 3.2, we can numerically calculate the right hand side of the equation (τ) and solve for L,T, and M using the equations of the generalized forces given in (3.6) since we have three equations and three unknowns. The values for $\dot{q}$ and $\ddot{q}$ were found numerically using a first order finite difference scheme and

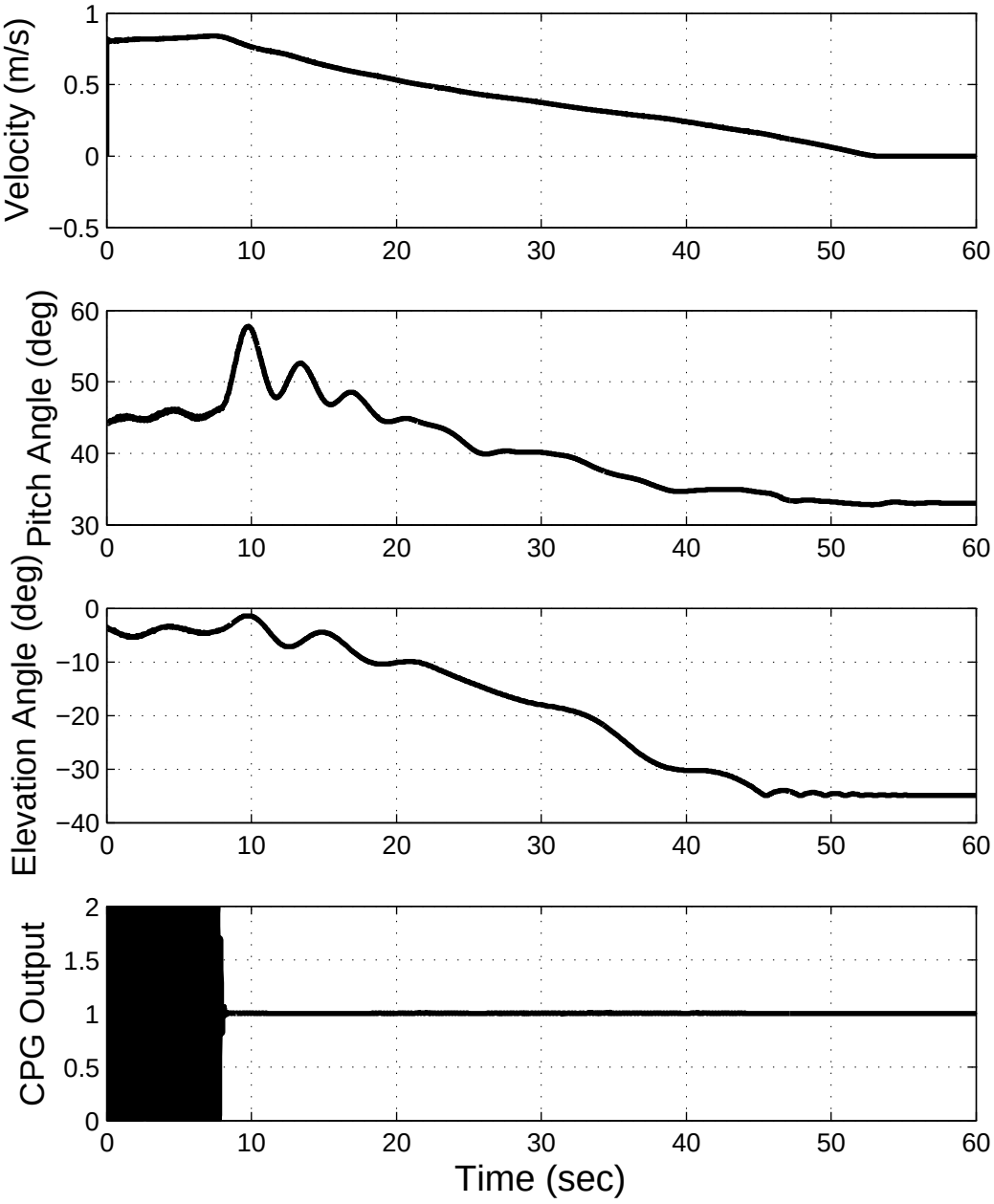


Figure 5.18: Flapping-Glide Transition

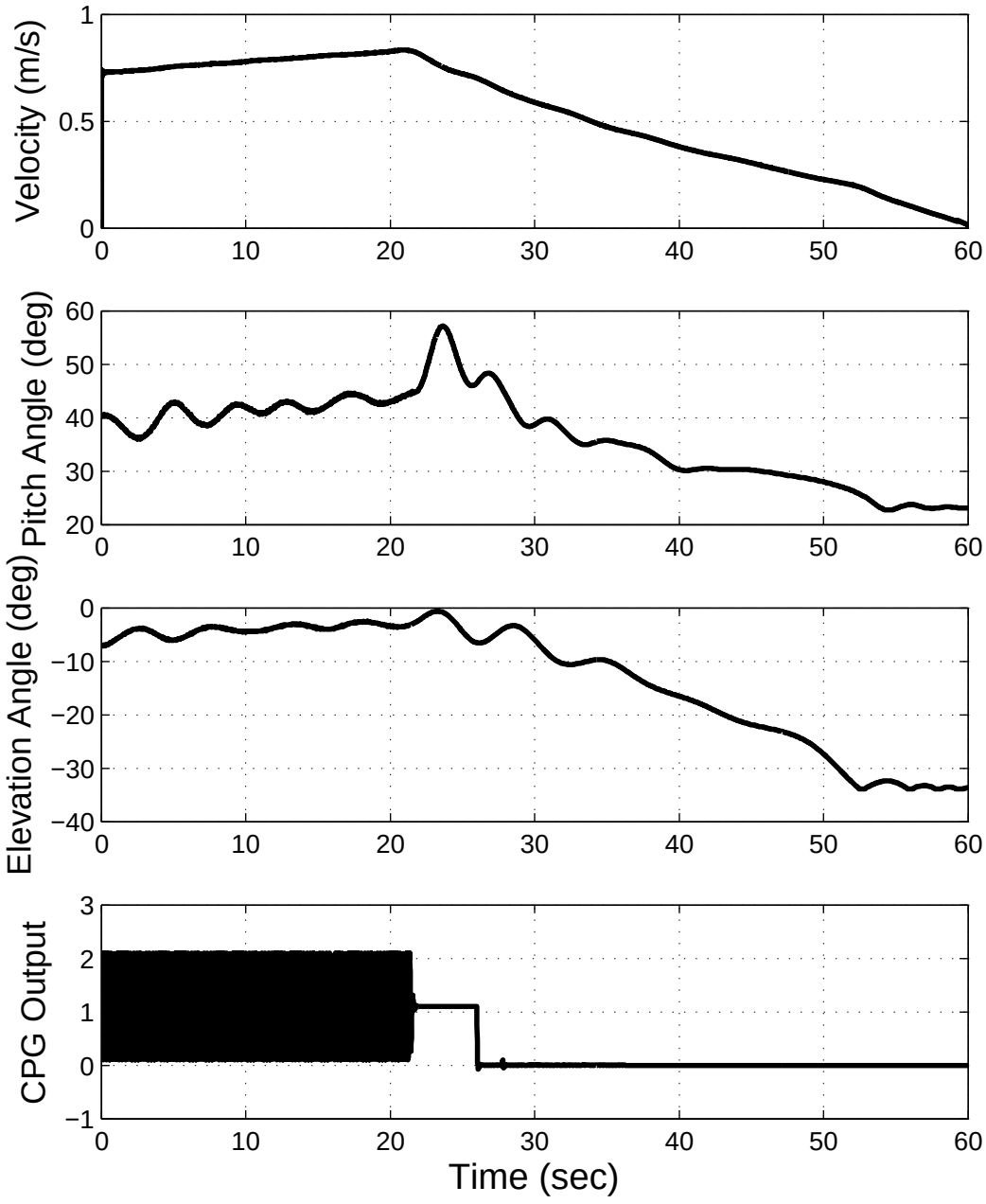


Figure 5.19: Flapping-Glide Transition, using flapping angle

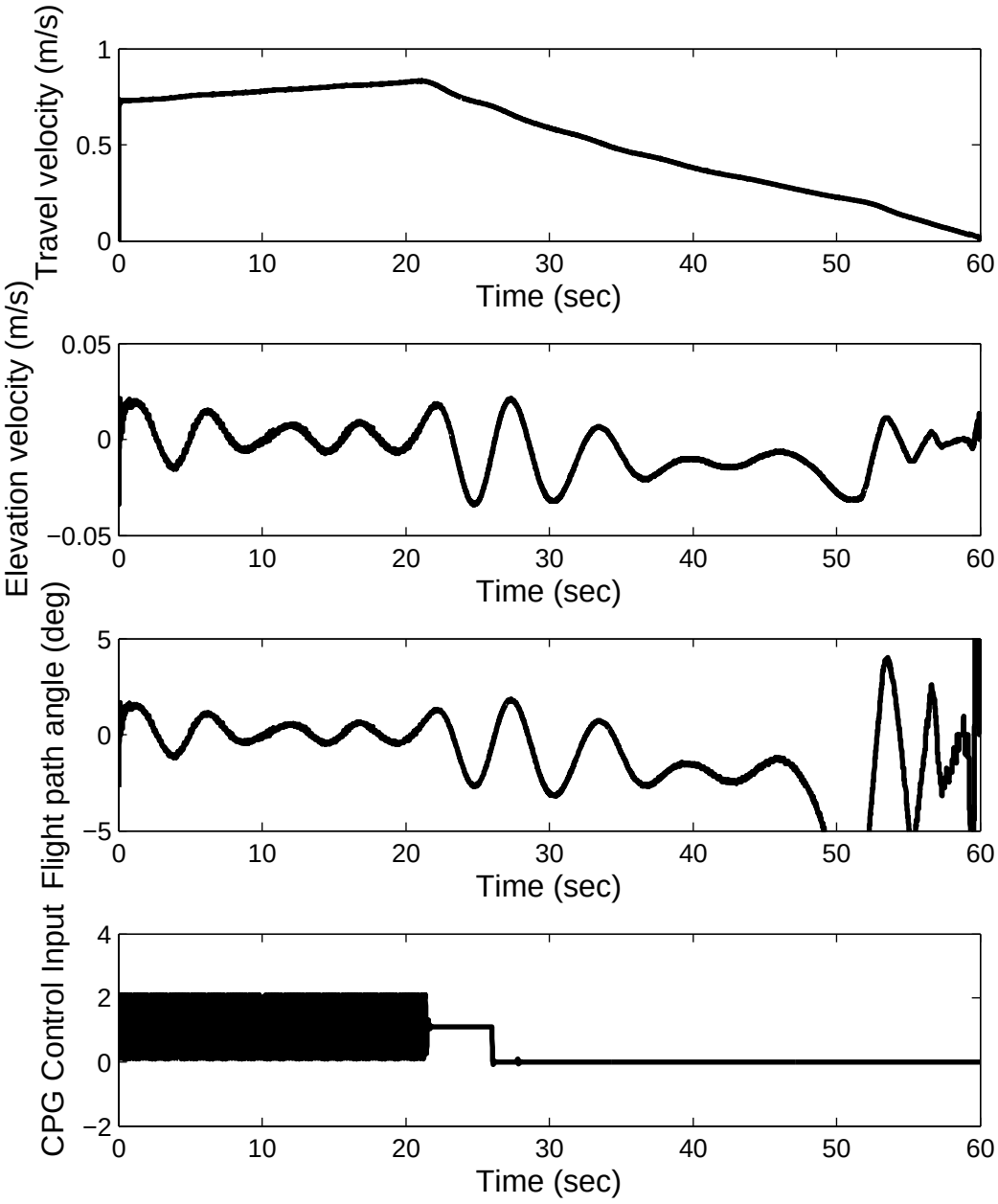


Figure 5.20: Flight path angle of flapping-glide transition

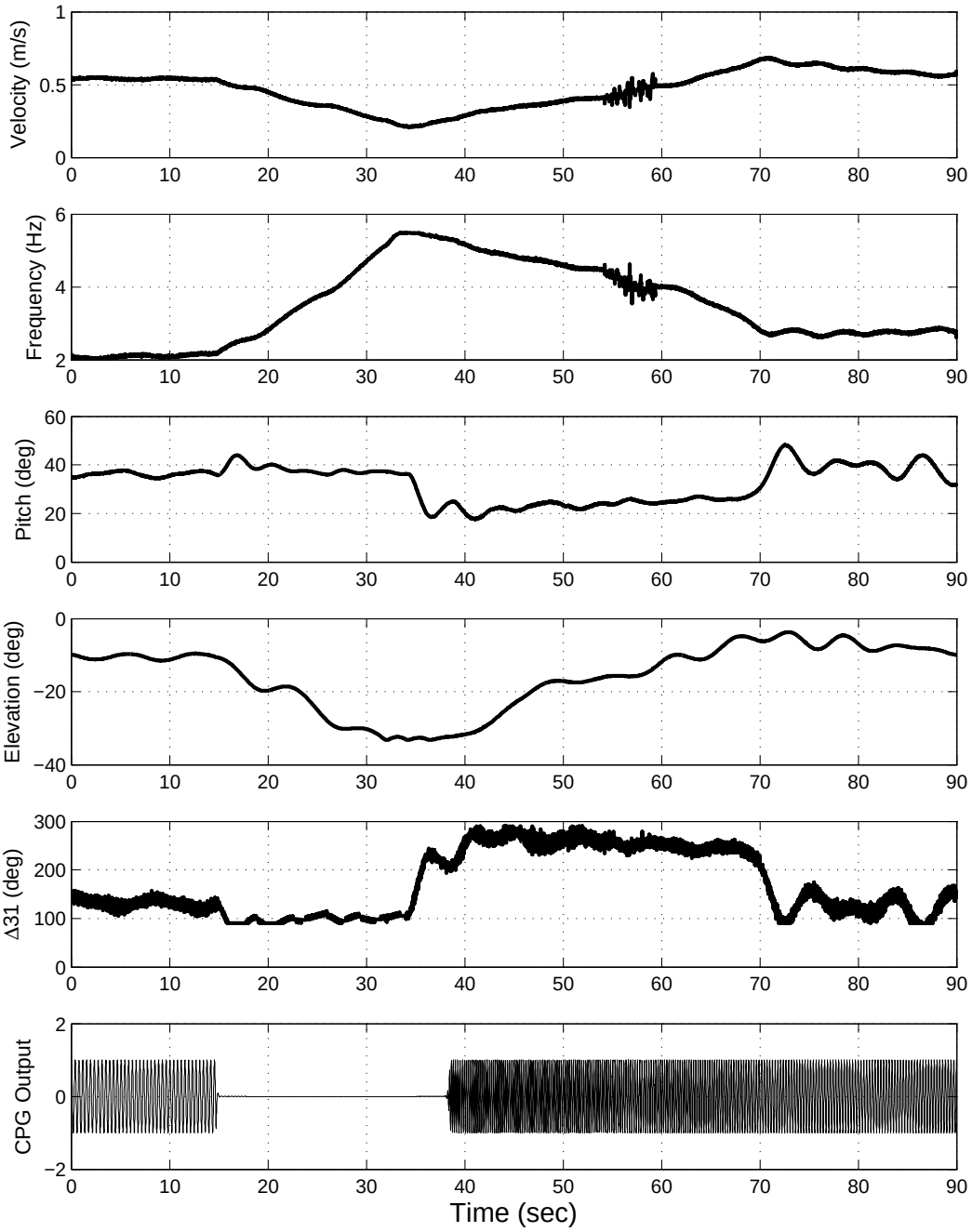


Figure 5.21: Flapping-Glide-Flapping Transition

a low pass filter. The M, C, and g matrices are calculated numerically at every time step since they are dependent on q and $\dot{q}$. The lengths and masses of the testbed are simply measured manually. Figure 5.22 shows the calculation of these forces and moments using the data from the experiment shown in figure 5.21.

Initially when the robotic bat is in flapping mode, the lift, thrust, and moment are relatively constant since the bat is in steady state. When gliding is activated at $t = 15$, as expected thrust drops since the wings are no longer propelling the robotic bat. Also the moment changes since the wings are not providing motion to change the pitch of the robotic bat. The lift does not automatically drop when the robotic bat is in glide mode due to the kinetic energy built up during flapping. When flapping mode is resumed near $t = 35$, there are noticeable spikes in lift, thrust, and moment as the robotic bat begins rapidly flapping its wings to return to its desired states.

There are some things to note with these calculations. First, the Euler-Lagrangian equations of motion derived in 3.2 assume the robotic bat and counterweights are point masses, and subsequently the pendulums are assumed massless. Moreover, it also assumes these point masses, corresponding to the centers of gravity of the robotic bat and counterweights, are not offset from each other and are placed in straight lines from one another. Of course, this is not the case since the positions of the elevation counterweight and pitch counterweight are slightly offset from their respective axes. In future work, this data can be compared with inertial measurement unit or force torque sensor data so that the analytical modeling can be refined to match a free flying vehicle. Also, the calculations can be done online by the controller for use in force control.

5.6 Measurements of Mechanical Coupling

When reproducing a fully rotatable shoulder joint with pitching motion, some amount of mechanical coupling is unavoidable. Because all of our numerical work assumes perfect

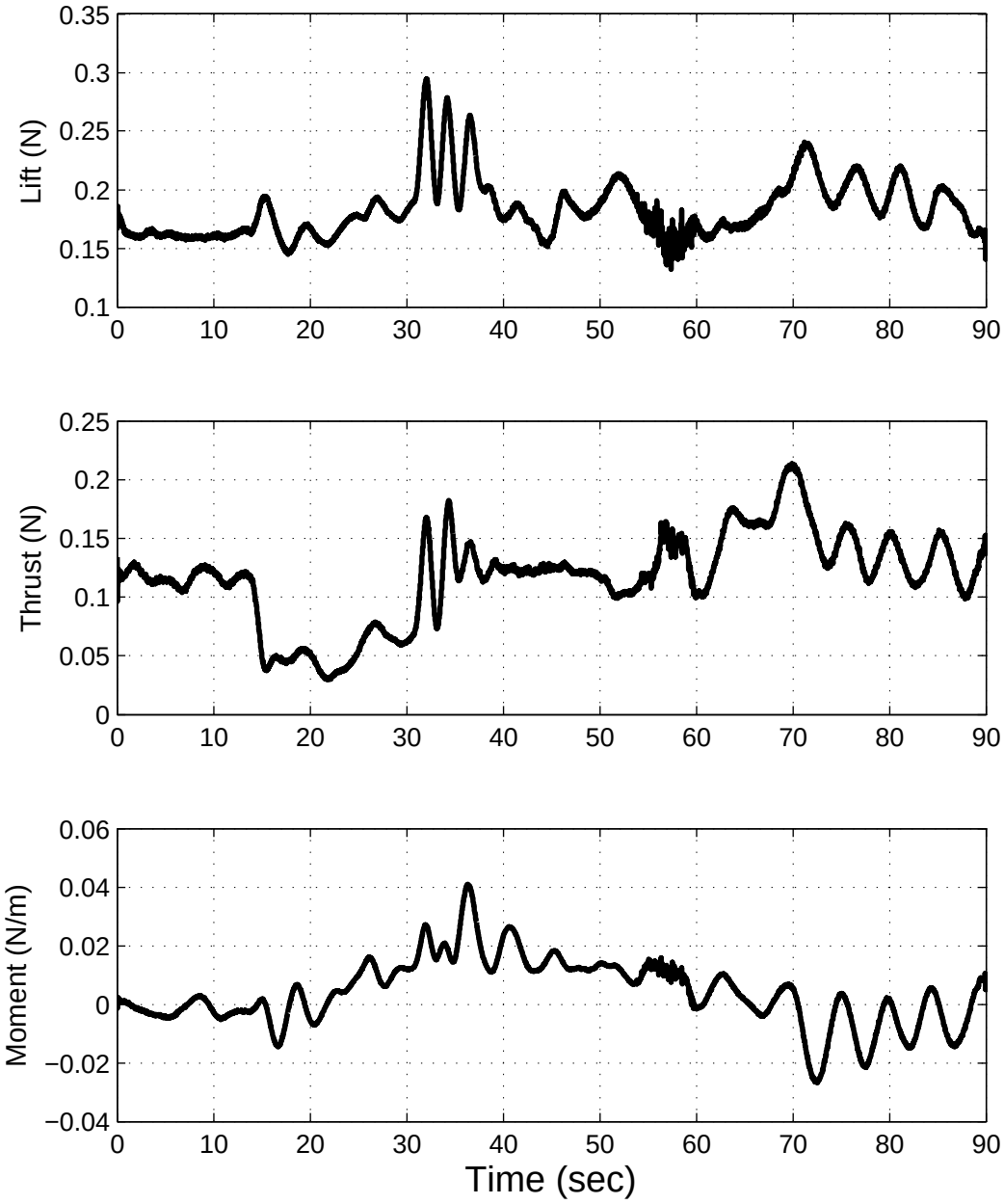


Figure 5.22: Numerically calculated lift, thrust, and moment

sinusoidal output with no coupling, we would like to measure the actual coupled output and potentially alter our control output waveforms to account for coupling. Moreover, with an idea of the wings motions we can further understand the dynamics of our system.

Special retro-reflective markers are placed on the bat's wings and body. Eight cameras are arranged around the robotic bat test bed record the position of these markers down to sub-millimeter accuracy. The position of these markers is used to calculate the rotations of the wings and body. From this wing angle data, the phase differences between the flapping, lead-lag, and pitch motions can be calculated. Rotations of the body are measured so that the rotation of the wings with respect to the body can be calculated while accounting for the body's own slight rotations.

Data was collected by having the robotic bat flap at 1 Hz while Vicon Tracker recorded rotation matrix for the robotic bat. Using the rotation matrices of the robotic bat's wings and the robotic bat's body, we could obtain the local rotation matrix for each of the wings with respect to the body. The pitch phase difference (Δ_{21}) was held at a constant 90°. Between each run, the phase difference between Lead-lag and flapping (Δ_{31}) was varied from -90 degree to 90 degree in 10 degree increments. Some example plots of the wing's rotations and the differences once phase differences are applied are shown in figures 5.23, 5.24, 5.25, and 5.26.

The wing motions are clearly not perfect sinusoids. In order to extract meaningful data from these motions, we treat them as noisy signals, and employ spectral analysis techniques to figure out the frequencies and phases of the major sinusoidal components. This is done using the Fast Fourier Transform(FFT) function in MATLAB. Using the FFT, we transform the wing's waveform in the time domain to the frequency domain, and we can figure out which sinusoidal components have the maximum amplitude. Once we know which frequency component is most significant and has the highest magnitude, we can figure out the phase angle of that particular frequency component. We do this for all of the flapping, pitching, and lead-lag motions and use the phase angle found using the FFT to computer the phase

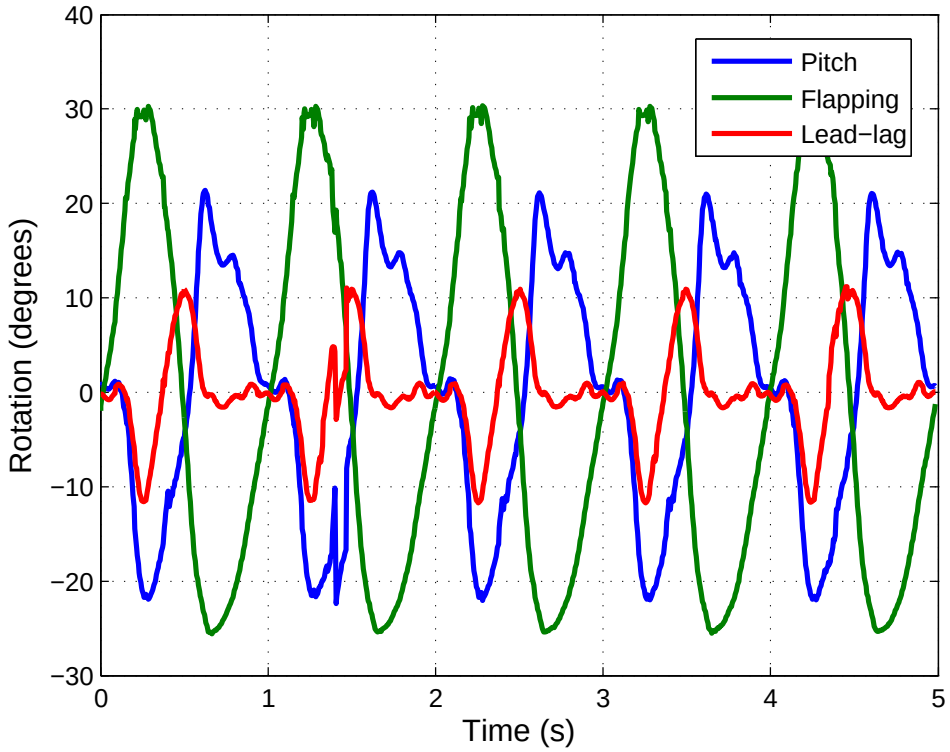


Figure 5.23: Experimental Left wing rotations with 10 deg input Δ_{75}

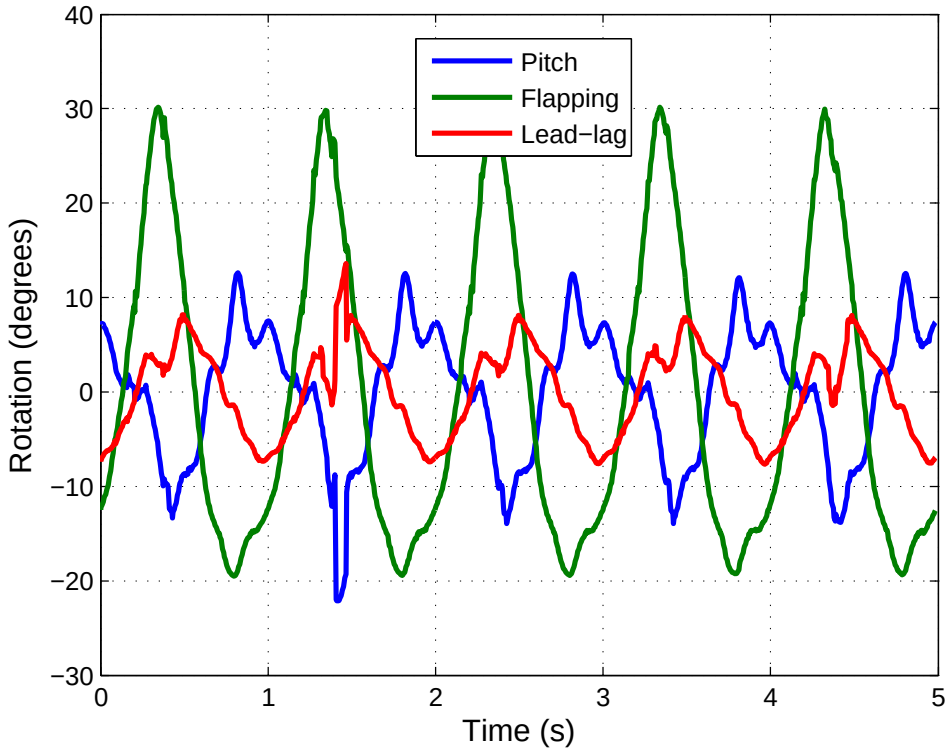


Figure 5.24: Experimental Right wing rotations with 10 deg input Δ_{31}

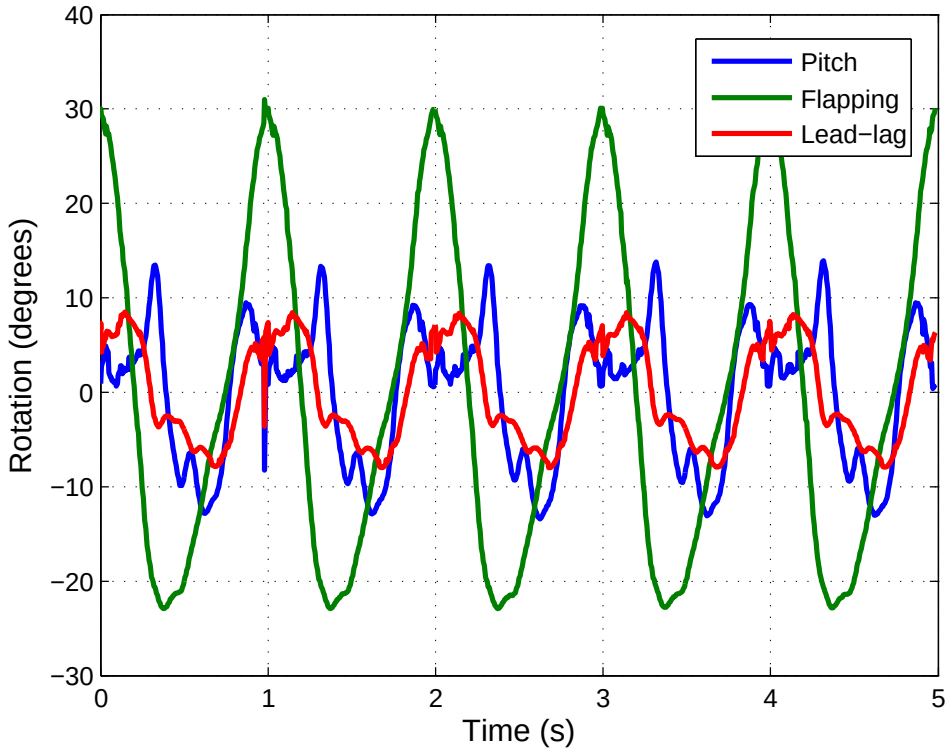


Figure 5.25: Experimental Left wing rotations with 60 deg input Δ_{75}

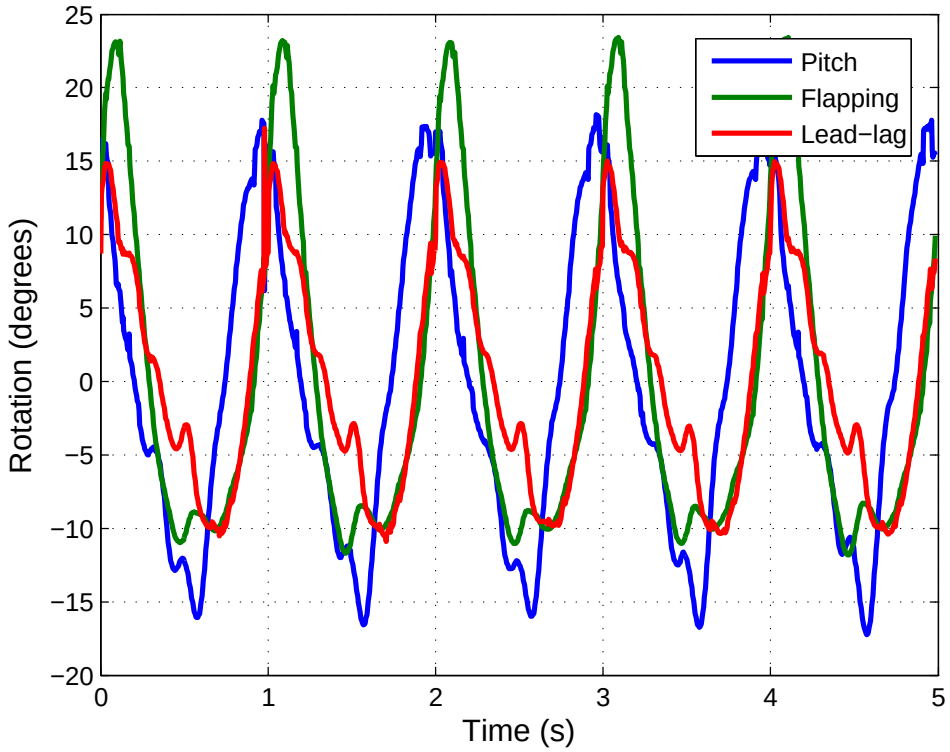


Figure 5.26: Experimental Right wing rotations with 60 deg input phase Δ_{31}

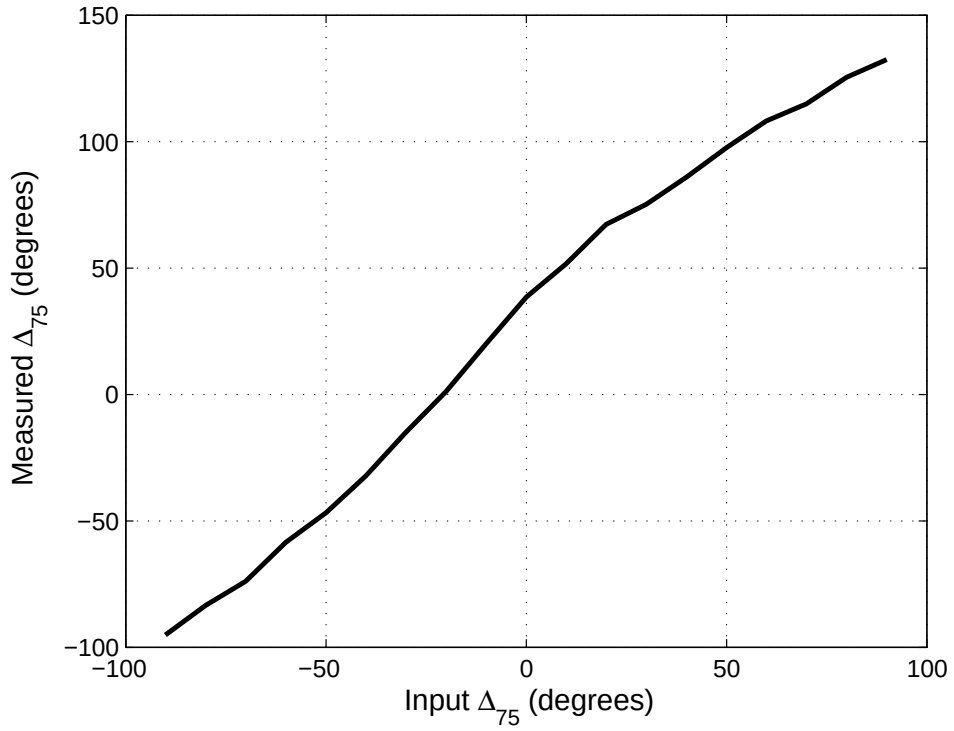


Figure 5.27: Measured vs Commanded phase differences between lead-lag and flapping motions (Δ_{75}) for left wing

difference in between different wing motions.

Each calculated phase difference was stored for every trial run, and after all data was collected, a relation between input phase difference in the controller vs. actual phase difference measured and calculated could be formed. The results are shown in figures 5.27, 5.28, 5.29, 5.30, 5.31, and 5.32.

Note that in figures 5.27 and 5.28, the relation between measured and commanded phase differences between lead-lag and flapping motions is close to the identity function. This shows that the flapping and lead-lag motions are effectively decoupled. From the mechanical design of the robotic bat this is expected to be the case since the actuators for lead-lag and flapping should have virtually no effect on each other. The small vertical shift in figures 5.27 and 5.28 can be attributed to a possibly not perfectly perpendicular stroke plane with respect to the robotic bat's body and noise due to data acquisition. A simple linear interpolation can be used to correct for this shift if desired, however subsequent tests showed this is not

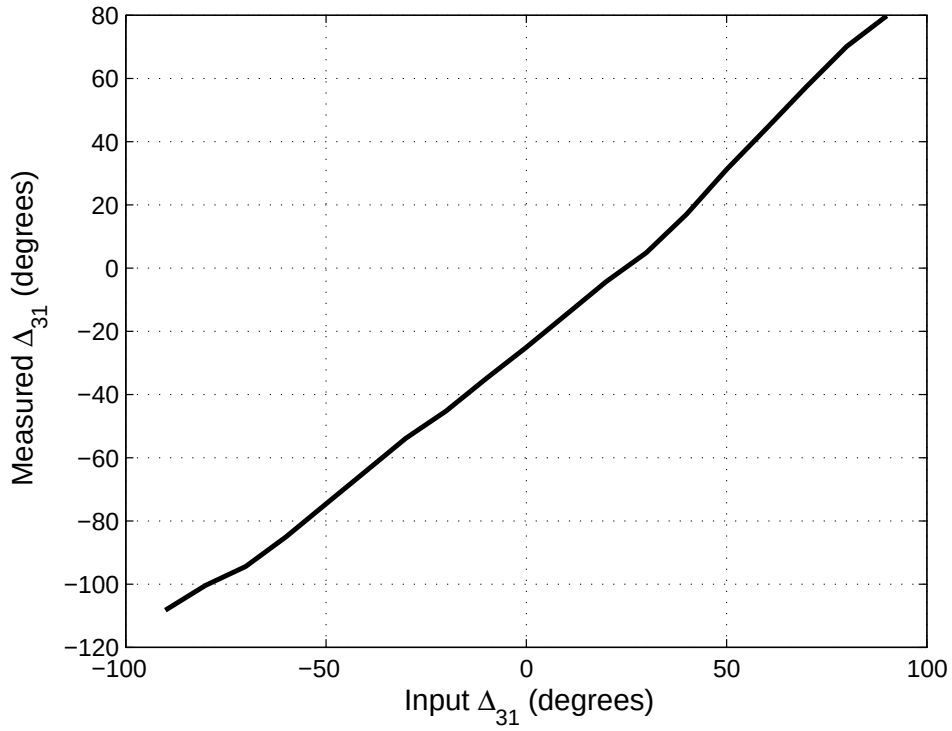


Figure 5.28: Measured vs Commanded phase differences between lead-lag and flapping motions (Δ_{31}) for right wing

entirely necessary to maintain efficiency in experiments.

Since the phase difference between flapping and lead lag was kept constant throughout all data acquisitions at 90 degrees, the phase difference between pitch and lead-lag is simply the phase difference between flapping and lead lag minus 90 degrees. Note data in plots 5.29 and 5.30 was unwrapped at the border between 180 degrees and -180 degrees, hence the y-axis ranges from 0 degrees to 360 degrees while the x-axis ranges from -180 degrees to 180 degrees. Keeping this relation in mind, both plots are fairly accurate for most range of phase differences of interest with a small amount of vertical shift, likely attributed to the reasons mentioned with the previous two plots. When the x-axis nears -90 degrees the relation is not as linear as it is in the rest of the graph, but regardless is not beyond simple numerical correction.

The relation between the pitch/flapping phase difference and the flapping/lead-lag phase difference is shown in figures 5.31 and 5.32. In an ideal situation, the measured pitch/flapping

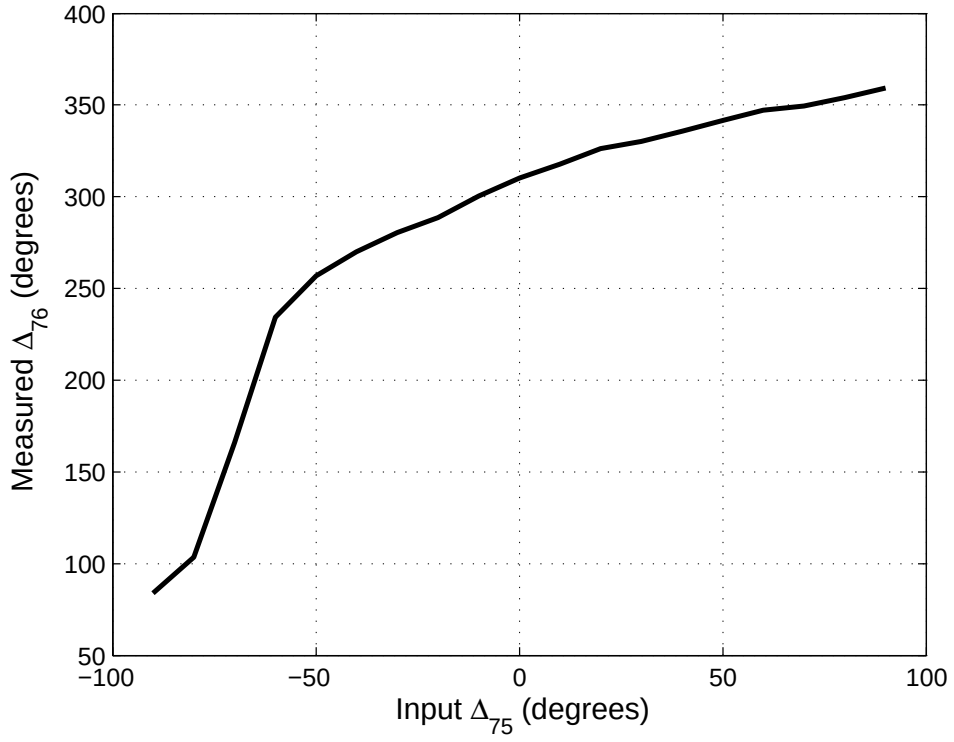


Figure 5.29: Relation between Δ_{75} and Δ_{76} for left wing

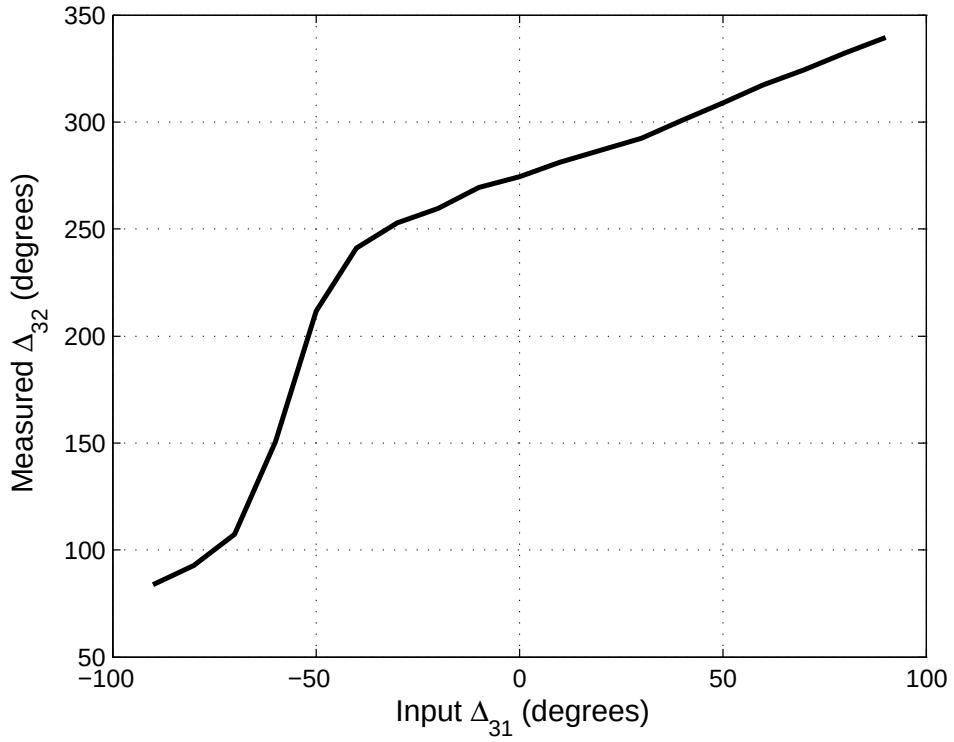


Figure 5.30: Relation between Δ_{31} and Δ_{32} for right wing

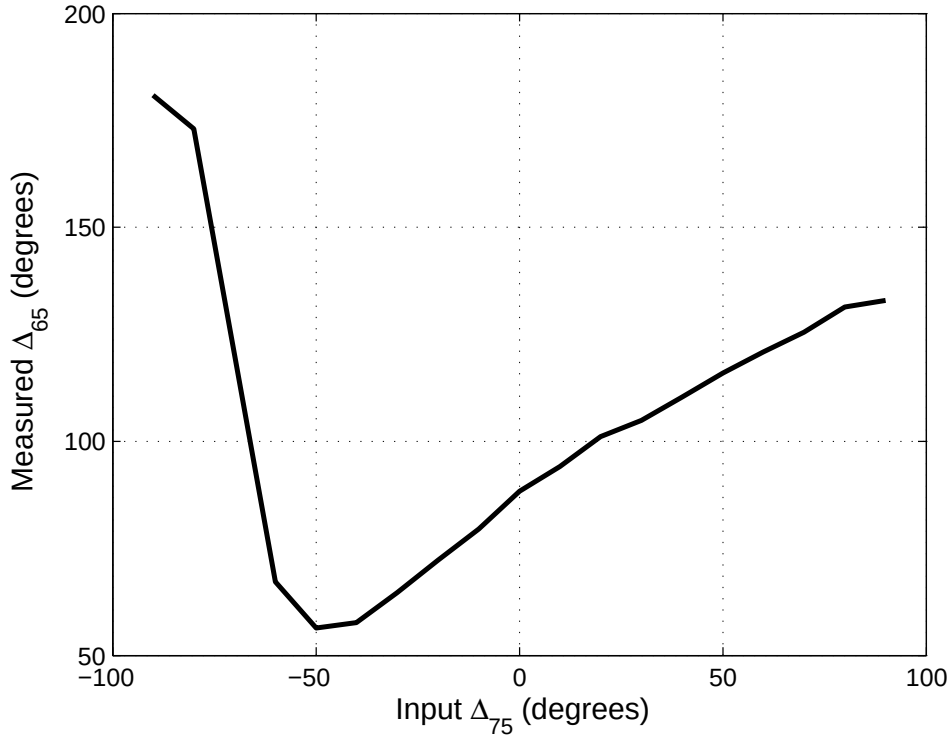


Figure 5.31: Relation between Δ_{75} and Δ_{65} for left wing

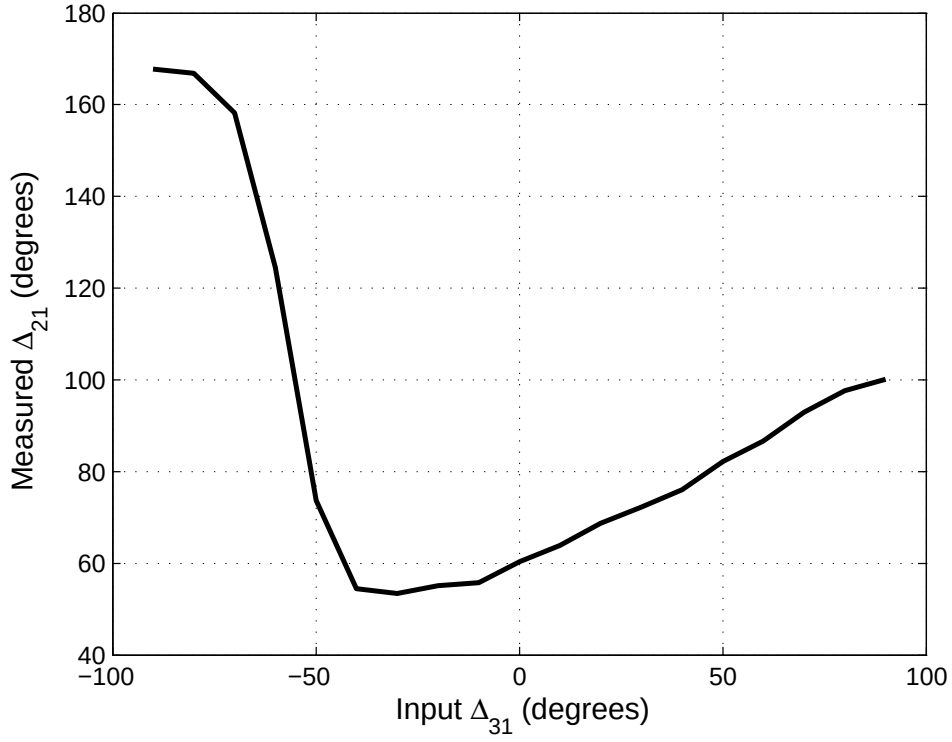


Figure 5.32: Relation between Δ_{31} and Δ_{21} for right wing

phase difference would stay constant at 90 degrees. However, this is not currently the case due to coupling issues. To decouple these motions, we traditionally would use forward kinematics to derive the measured pitch angle of the wing with respect to rotations of the servos and motors. This can be done by approximating the servos and motors as rotating crackshafts similar to a crackshaft-cylinder assembly found in engines. The position of the wing joint where the lead-lag and flapping pushrods could be found, as could the rotation of the wing joint which equates to the pitching angle of the wing. Then, we would use inverse kinematic techniques to find a function of the position of the end effector (in this case the position of the wing joint and the pitch angle) as a function of the inputs of its actuators (the rotations of the servos and motors). However, simply doing the forward kinematics results in an extremely long function involving transcendental functions such as trigonometric functions, as well as higher order polynomials. Even making some oversimplifications such as assuming small angle rotations, the end result is an equation that would be easily solved, if at all, in real time using an online controller. Moreover, such an effort would be limited only to our test bed and not applicable for other models. Thus, going the software route to decouple the motions is desired.

Note that figures 5.31 and 5.32 are only for a desired phase difference of 90 degrees. One method using the Vicon system to decouple the system is to change the desired pitch/flapping phase difference in set increments while repeating the entire procedure of measuring phase differences while varying the flapping/lead-lag phase difference, for each individual pitch/flapping phase difference value. Of course, this would be very ad hoc and require potentially hundreds of data captures depending on the size of the increments, and again only applicable to this current model. Simple mechanical adjustments could potentially alter the characteristics of the wing motions and nullify a priori assumptions. Therefore, finding optimal phase difference values through experimentation and data driven models such as reinforcement learning is likely to yield more appreciable results in a more efficient manner.

Chapter 6

Supplemental Implementations

6.1 Introduction

Some extra possibilities regarding research with flapping flight MAVs using the test bed, as well as research with other related technologies described in this thesis are described here.

6.2 Implementation of CPGs on FPGAs

Field programmable gate arrays (FPGA) provide a mechanism to prototype chips and other logical circuits. They are integrated circuits which can be reprogrammed by the customer to match the customer's needs and requirements. These FPGAs can provide a low level implementation for a computer or controller, using only the bare minimum amount of digital logic required. Because of this potential minimal implementation, there is some interest of implementing biological systems onto FPGAs. There is also interest in using FPGAs as the implementation of a robot controller. Some research can be found in [41], [42], and [43].

So far, only basic work involving FPGAs has been accomplished with our research using a Xilinx Spartan-6 FPGA. In experiments conducted so far, the ability to use FPGA circuits to generate control signals for robotic motion is explored. As preliminary project, generating a simple signal using the Forward Euler Method and the FPGA digital circuits was done to control a robotic arm. The signal used was a simple sine wave and the output signal was a PWM signal. Using floating point algorithms and a converter, it is possible to quickly simulate the governing differential equation through a digital circuit. Using an FPGA allows

for rapid prototyping and development of complex digital circuits, and have become quite popular in the field of digital signals and embedded systems.

The set up was simply connecting a robotic arm to the FPGA and a power source. Then, connecting the FPGA to a computer via USB port, and then loading the program into the FPGA from the computer and running the program. The FPGA circuit was designed to provide the three signals need by the PWM converter box: signal, +5V, and ground. The PWM signal comes in bursts ranging from 1 to 2 ms at a rate of one new signal every 20 ms (or 50 Hz, a standard frequency for most servos). This provided the rotation from 0 degrees to 180 degrees in the wrist of the robotic arm.

Designing the digital circuit is the real challenge. In order to ensure correct functionality and to generate a basic understanding of the circuit before more challenging signals were used, a simple sine wave was developed. This is the result of the differential equation in the time domain equation in Eqn. (6.1).

$$\ddot{X} + \omega^2 X = 0$$
$$X = A\sin(\omega t) + B\cos(\omega t) \tag{6.1}$$

Where A is the initial velocity, and B is the initial position. Using the Forward Euler Method, this equation can be reduced into two simple linear relations which recursively relate the next value for the position and velocity to the previous corresponding values.

$$X^+ = X + \Delta t \dot{X}$$
$$\dot{X}^+ = \dot{X} + \Delta t \ddot{X}$$

Or, using governing differential equation in equation (6.1)

$$\dot{X}^+ = \dot{X} - \Delta t \omega^2 X \tag{6.2}$$

The key here would be to have a valid time step, Δt. This varies based on the digital circuit, and the individual components of the circuit. With the Xilinx Spartan-6 FPGA, the operating frequency is 27 MHz, with all of the components able to react within one clock cycle. This means that the time step Δt, must be equal to the time between clock cycles. This is simply the inverse of the operating frequency, which in this case is 37.037 nanoseconds (0.000000037037 seconds). This is a very small number, and thus a double precision floating point number is used to properly represent this number. This uses 64 bits to store a number.

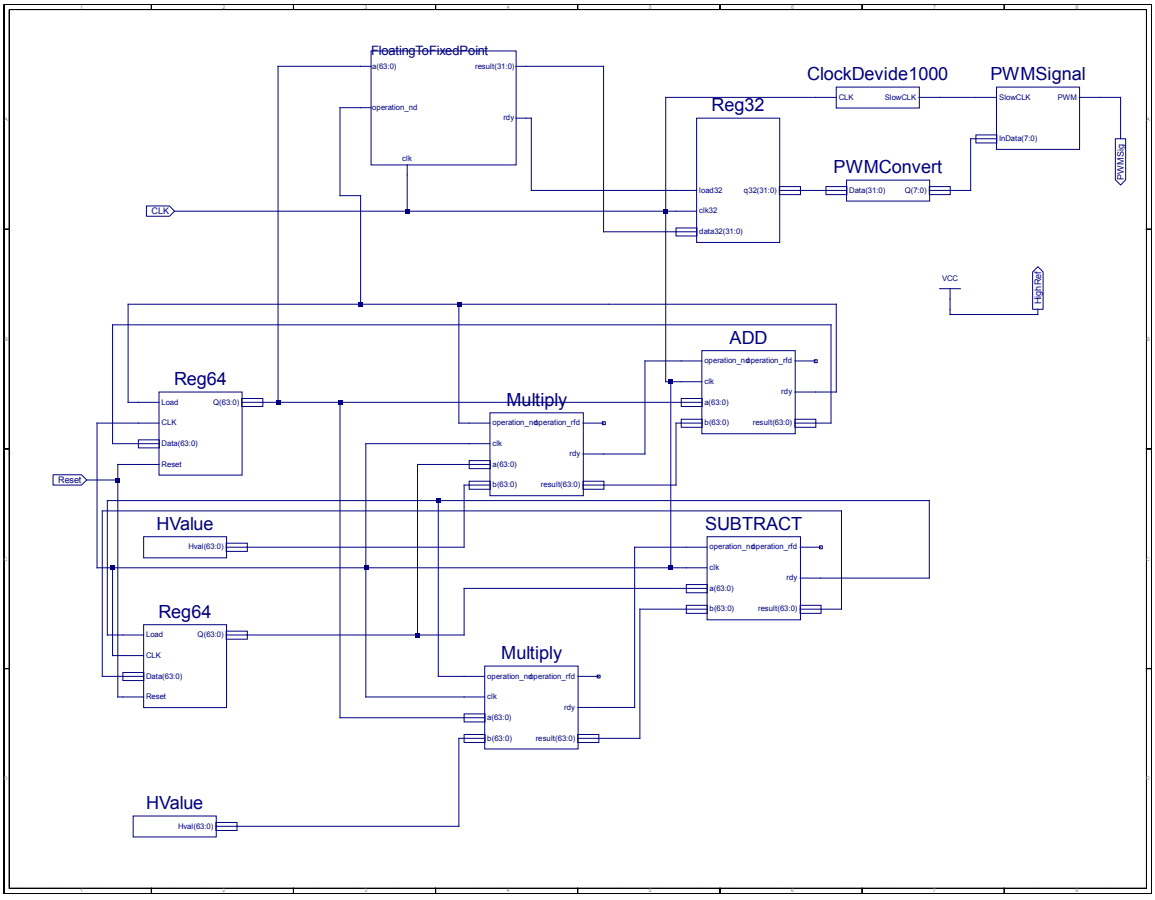


Figure 6.1: Schematic for sine wave PWM signal generation

The basic schematic for this is shown in Fig. 6.1. The main idea is that the position and the velocity are updated every clock cycle. The position is added to the multiplication of the velocity and the time step and then stored back into position, giving the new position. The

velocity is similar except that the new velocity is the old value, minus the multiplication of the time step and the frequency squared. Then the position data is converted from floating point to a fixed point notation, which is then stored for 20 ms; the time when the next pulse comes in. This is done by using the clock divide block, that creates a new clocking signal at a much lower rate by taking the 27 MHz signal and reducing its frequency by a factor of 1000 to 27 KHz. This will allow the PWM signal to generate different signals for different lengths of time depending on the data input, a range from 0 to 26. This effectively gives 27 different positions that can be outputted. This number is not significant; it was used because it was shorter to do and shows a wide range of positions.

While this circuit works, the floating point units, the two multipliers, the adder, the subtraction, and the converter blocks were implemented using DSP slices. While they worked in this application, they will not be sufficient for more complicated signals. Also, there are only six available DSP slices on the FPGA board currently used. In addition, the timing of these blocks are dependent on how the manufacturer made them. In order to implement more advanced signals, the code for these blocks must be manually created. Once the basic floating point operations have been created purely in digital logic, outside of DSP slices, just about any signal can be created using this set up, especially if the signal is a digital controller. If the signal needs varying amplitude, a digital to analog converter can be created as well.

One of the major end goals of reproducing CPG signals on FPGAs would be to implement our CPG controller on a microchip for use in an embedded system. One of the benefits of CPGs is the reduced dimensionality for control and thus reducing the top level computation needed in the main controller, the "brain", of the system. Thus, theoretically the added power of a microcontroller or computer would not be necessary for a controller as simple as one based on CPGs. This can help reduce the physical weight of avionics on a flapping flight MAV and also reduce overhead maintenance and computations that would be involved with something more powerful such as a microcontroller or even a small computer. Another

benefit is that while computers are usually serial by nature and thus perform computations and operations sequentially, an FPGA executes operations in parallel with one another.

A possibility for implementing CPGs would involve using System Generator from Xilinx, which would allow the integration between the FPGA hardware description language and Simulink. There is also HDL Coder, a toolbox which allows the generation of HDL code from Simulink to program FPGAs.

Chapter 7

Conclusions and Future Work

7.1 Conclusion

The objectives of this thesis were to show some ongoing research into controlling flapping flight MAVs, the challenges presented by such a problem, and to provide insight into possible future developments given current technological availabilities. Flapping flight MAVs inspired by bats still have a long way to go, however, current progress and accomplishments were demonstrated here. We first examined the current state of the art of flapping flight by looking into previous work, models, and hardware developed, and then derived some basic governing equations to provide a theoretical framework of our model and controller.

The low level implementation of a testbed for flapping flight test bed was described from a mechanical, electrical, and software perspective. We also described the many other pieces of hardware and equipment necessary to operate and control such a testbed, as well as other equipment which can supplement our research.

From an experimental perspective, we demonstrated control algorithms for flapping flight MAVs and tested them with our own constructed flapping flight testbed. We showed it is possible to control the longitudinal motion of flapping flight MAVs simply by modulating the phase difference in between wing motions. Our use of CPGs also showed it is possible to reduce the dimensionality of a system as complex as flapping flight aerial vehicles to reduce top level computation required. With CPGs, we can use top level controllers as simple as PID controllers to control the pitch and velocity of flapping flight MAVs. We also showed that going from flapping flight to gliding flight is as simple as flipping a switch. We also

showed how we can use our current dynamic model to approximate forces and moments generated during flapping flight. The exact motions of flapping wings can be analyzed using a motion capture system and we can use this data to tune our hardware and controller. We also looked into other implementations of a flapping flight MAV controller beyond our current testbed.

Bat-like flight is a challenging problem that cannot be solved via averaging or with traditional tail-derived stability. We have demonstrated the ability to stabilize and control longitudinal motions via CPGs with the RoboBat. As expected, the top-level controllers are of low dimension and can be made very simple, because most of the hard work is done by the CPGs. Given the mechanical coupling shown in Section 5.6, it is quite remarkable that such control was immediately as effective as it was. Further work can still be done to create a pattern generator layer so to optimize the output waveforms. Additionally, we expect to better quantify the forces and moments actually produced via the dynamic model, so that we can make better predictions for a free-flying robotic bat.

7.2 Future Work

While progress is expected towards the development of autonomous flapping flight, current technological limitations are still far too great to accurately mimic biological flight systems and thus more research and development is required in several different areas of study in order for significant progress to be made in this area. Examples include material science, where lightweight actuators that mimic muscles are currently being developed in hopes of eventually replacing servos and motors which can be both bulky and heavy. The mechanical structure would also need to be made of materials that are lightweight yet durable enough to withstand aerodynamic loading. Another example is smaller, lightweight computing platforms that are powerful enough to implement our controllers, even though our controllers already attempt to minimize required computational power. Electronics such as an onboard power source

which is constrained by weight and volume would be a necessity for a free flying MAV, as well as communication devices and cameras for sight. From a mathematical and software perspective, algorithms for navigation and path planning would be applied for a flapping flight MAV.

The testbed itself, and any other future implementations, will likely always be in need of continuous improvements. Particularly, the mechanical structure of the robotic bat will eventually need to be redesigned to further reduce the high moment of inertia that is currently present in the testbed to closely mimic what is found in a freely flying situation. This will allow more experimentation for the transition between flapping and gliding flight, as well as other maneuvers in gliding flight such as perching. Different actuators may be used to control the wing motions, and for testing certain models it may be necessary to change the total number of actuators used. As mechanical design of actuators develops, we expect robotic fliers in free flight to be able to utilize the key feature of phase synchronization and control of phase differences in stability and control of body motions. The major problem of identifying a method of proving such stability analytically is still open. However, this paper has demonstrated the result experimentally. Since this CPG controller design also features rapid inhibition of oscillation, it leads to the important problem of gliding flight and maneuvers while gliding. Two more topics without much current work in our research include takeoff and landing.

The wings currently used in our model are assumed to be rigid; in reality bat wings have many joints similar to that of a human hand. Wings with multiple joints would improve our design, but also increase the complexity of both our dynamic modeling and our controller, as well as require a more clever electro-mechanical implementation.

In the future, we would like to implement CPGs via field programmable gate arrays to truly demonstrate decentralized control with CPGs and reduced dimensionality and complexity of CPG based controllers. This will require a deeper understanding in discrete logic and computer engineering, as well as an understanding of the continuous dynamics which

govern our understanding of CPG based flapping flight control. With an FPGA implementation we could potentially fabricate a microchip of our controller which would tremendously reduce required power and volume of an onboard controller for a flapping flight MAV.

Our controller implementations are mostly based on dynamical systems and other more traditional control methods such as PID. We have to truly explore artificial intelligence and path planning control methods in detail. Reinforcement learning will hopefully provide a suitable alternative into flapping flight control while providing new research opportunities to bridge the gap between more computer science oriented topics such as AI and machine learning with traditionally more mechanical topics such as dynamics and control.

An attempt to describe the mechanics flapping flight in a quantitative manner is considerably difficult, involving a broad range of scientific fields such as aerodynamics, aero-elasticity, kinematics, dynamics, and control. To describe flapping flight in a manner such that a computational controller could understand the concept of flapping flight is even more difficult. Thus, there is ongoing research relating UAVs and MAVs and various types of machine learning, such as reinforced learning.

The fundamental methods of reinforcement learning include dynamic programming, Monte Carlo methods, and temporal-difference learning. Dynamic programming involves finding an optimal policy by assuming a perfect model of the environment as a Markov decision process and solving the Bellman equation, and example algorithms include policy iteration and value iteration [44]. Monte Carlo methods refer to methods that involve learning from only experience based on averaging complete returns and do not assume complete knowledge of the environment. Temporal-difference learning involves learning from direct experience like Monte Carlo methods, but also updates estimates based on previous estimates like dynamic programming methods. An example of a temporal-learning method is Q-Learning, which attempts to calculate the optimal action-value function (the return for taking a certain action at a certain state), independent of policy [44].

Data-driven models may be more contributing than physics based models due to the

computational burden of CFD on low Reynolds number flow. Simply put, birds, bats, and insects do not perform complex nonlinear calculations while flying and do not solve the Navier-Stokes equations in real time; they simply flap their wings and adjust as necessary to get where they need to go[45].

More experimentation regarding reinforcement learning and its application to flapping flight MAVs can be found in [46]. By modeling the dynamics of a the wings of a particular kind of fly, and by using what is known as the Q-learning algorithm, the optimal policy for the motion of a flapping wing is found to maximize lift using rewards and punishments.

Beyond flapping flight, or MAVs or UAVs in general, reinforcement learning is of interest to those in the robotics community in general for various tasks. In [47], reinforcement learning is used for a robotic arm to acquire the necessary motor skills to flip a pancake in a frying pan; something which would be considerably difficult to model analytically.

For the setup described in this thesis, reinforcement learning could be used to attempt to find the optimal wing trajectory in terms of optimal phase differences to extremize a given cost function. This could include maximizing propulsive efficiency, maintaining a desired pitch and/or angle of attack, maximizing forward velocity, or any number of other parameters. In Chapter 5, we described closed loop experiments which used PID controllers to find suitable beating frequency and input phase difference values for given desired forward velocity and pitch values. Reinforcement learning could potentially find a wider range of suitable values for several more desired input values such as elevation and flight path angle when gliding. Some challenges for this would include defining a suitable cost function for the robotic bat to learn from rewards and punishments, and coming up with a policy that would hopefully converge to some optimal value after a number of trials. However, it may also be desirable to use a reinforcement learning method which does not rely on a model of the environment simply due to the complexities and difficulties that currently exist when modeling flapping flight.

References

[1] S.-J. Chung and M. Dorothy, "Neurobiologically inspired control of engineered flapping flight," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 33, no. 2, pp. 440–453, 2010.

[2] T. J. Mueller, *Fixed and Flapping Wing Aerodynamics for Micro Air Vehicle Application*. Progress in Astronautics and Aeronautics, AIAA, 2001.

[3] S.-J. Chung, M. Dorothy, and J. R. Stoner, "Neurobiologically inspired control of engineered flapping flight," in *AIAA Infotech @ Aerospace and Unmanned Unlimited Conference and Exhibit*, Seattle, WA, Apr. 2009, aIAA Paper 2009-1929.

[4] B. Sanders, R. Crowe, and E. Garcia, "Defense advanced research projects agency: Smart materials and structures demonstration program overview," *Journal of Intelligent Material Systems and Structures*, vol. 15, 2004.

[5] X. Deng, L. Schenato, W. C. Wu, and S. S. Sastry, "Flapping flight for biomimetic robotic insects: Part i-system modeling," *IEEE Trans. on Robotics*, vol. 22, no. 4, pp. 776–788, 2006.

[6] R. J. Wood, "The first takeoff of a biologically-inspired at-scale robotic insect," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 341–347, 2008.

[7] A. A. Paranjape, S.-J. Chung, and M. S. Selig, "Flight mechanics of a tailless articulated wing aircraft," *Bioinspiration and Biomimetics*, vol. 6, 2011.

[8] A. J. Bergou, L. Ristroph, J. Guckenheimer, I. Cohen, and Z. J. Wang, "Fruit flies modulate passive wing pitching to generate in-flight turns," *Physical Review Letters*, vol. 104, 2010.

[9] X. Deng, L. Schenato, W. C. Wu, and S. S. Sastry, "Flapping flight for biomimetic robotic insects: Part ii-flight control design," *IEEE Transactions on Robotics*, vol. 33, no. 4, pp. 789–803, 2006.

[10] D. B. Doman, M. W. Oppenheimer, and D. O. Sigthorsson, "Wingbeat shape modulation for flapping-wing micro-air-vehicle control during hover," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 33, no. 3, pp. 724–739, 2010.

[11] L. Schenato, "Analysis and control of flapping flight: from biological to robotic insects," Ph.D. dissertation, University of California at Berkeley, 2003.

[12] S. Grillner, A. Koslov, P. Dario, C. Stefanini, A. Menciassi, A. Lansner, and J. Kotaleski, "Modeling a vertebrate motor system: pattern generation, steering and control of body orientation," *Progress in Brain Research*, vol. 165, pp. 221–234, 2007.

[13] O. Kiehn, "Locomotor circuits in the mammalian spinal cord," *Annual Review of Neuroscience*, vol. 29, pp. 279–306, 2006.

[14] M. H. Dickinson, F. O. Lehmann, and S. P. Sane, "Wing rotation and the aerodynamic basis of insect flight," *Science*, vol. 284, pp. 1954–1960, 1999, june.

[15] A. Azuma, *The Biokinetics of Flying and Swimming*, 2nd ed. AIAA, 2006.

[16] X. Tian, J. Iriarte-Diaz, K. Middleton, R. Galvao, E. Israeli, A. Roemer, A. Sullivan, A. Song, S. Swartz, and K. Breuer, "Direct measurements of the kinematics and dynamics of bat flight," *Bioinspiration and Biomimetics*, vol. 1, 2006, s10-S19.

[17] S. M. Swartz, K. L. Bishop, and M.-F. Ismael-Aguirrre, "Dynamic complexity of wing form in bats: Implications for flight performance," in *Functional and Evolutionary Ecology of Bats*. Oxford, UK: Oxford University Press, 2005.

[18] G. K. Taylor and R. Zbikowski, "Nonlinear time-periodic models of the longitudinal flight dynamics of desert locusts schistocerca gregaria," *J. R. Soc. Interface*, vol. 2, pp. 197–221, 2005.

[19] Z. J. Wang, "Aerodynamic efficiency of flapping flight: Analysis of a two-stroke model," *The Journal of Experimental Biology*, vol. 211, pp. 234–238, 2008.

[20] G. Sachs, "Why birds and miniscale airplanes need no vertical tail," *Journal of Aircraft*, vol. 44, no. 4, pp. 1159–1167, 2007.

[21] W. J. Crowther, "Perched landing and takeoff for fixed wing uavs," in *NATO Symposium on Unmanned Vehicles for Aerial, Ground, and Naval Military Operations*, 2000.

[22] A. Wickenheiser and E. Garcia, "Longitudinal dynamics of a perching aircraft," *Journal of Aircraft*, vol. 43, no. 5, pp. 1386–1392, 2006.

[23] A. Wickenheiser and E. Garcia, "Optimization of perching maneuvers through vehicle morphing," *AIAA Journal of Guidance, Control, and Dynamics*, vol. 31, no. 4, pp. 815–823, 2008.

[24] J. W. Roberts, R. Cory, and R. Tedrake, "On the controllability of fixed-wing perching," in *Proc. American Control Conference*, St. Louis, MO, 2009.

[25] R. Cory and R. Tedrake, "Experiments in fixed-wing UAV perching," in *Proc. AIAA Guidance, Navigation and Control Conference, Honolulu, HI*, 2008, AIAA Paper 2008–7256.

[26] M. Dorothy, A. A. Paranjape, P. D. Kuang, and S.-J. Chung, "Towards bio-inspired robotic aircraft: CPG-based control of autonomous flapping and gliding flight," in *Intelligent and Autonomous Aerospace Systems*. Reston, VA: American Institute of Aeronautics and Astronautics (AIAA), 2012, J. Valasek (Editor).

[27] S. M. Swartz, M. S. Groves, H. D. Kim, and W. R. Walsh, "Mechanical properties of bat wing membrane skin," *Journal of Zoology*, vol. 239, pp. 357–378, 1996.

[28] F. Herrero-Carrón, F. B. Rodríguez, and P. Varona, "Bio-inspired design strategies for central pattern generator control in modular robotics," *Bioinsp. Biomim.*, vol. 6, 2011.

[29] S. Ho, H. Nassef, N. Pornsinsirirak, Y.-C. Tai, and C.-M. Ho, "Unsteady aerodynamics and flow control for flapping wing flyers," *Progress in Aerospace Sciences*, vol. 39, 2003.

[30] J. Birch and M. Dickinson, "Spanwise flow and the attachment of the leading-edge vortex on insect wings," *Nature*, vol. 412, pp. 729–733, Aug. 2001.

[31] S. Ho, H. Nassef, N. Pornsinsirirak, Y.-C. Tai, and C.-M. Ho, "Flight dynamics of small vehicles," in *Proceedings of the International Council of the Aeronautical Sciences (ICAS)*, Toronto, Canada, 2002, pp. pp. 551.1 – 551.10.

[32] T. Pornsinsirirak, S.-W. Lee, H. Nassef, J. Gransmeyer, Y.-C. Tai, C.-M. Ho, and M. Keennon, "Mems wing technology for a battery-powered ornithopter," in *Proceedings of the International Conferences on MEMS*, Miyazaki, Japan, 2000, pp. 709–804.

[33] G. Jadhav, *The Development of a Miniature Flexible Flapping Wing Mechanism for Use in a Robotic Air Vehicle*, 2007, m.S. Thesis, Georgia Institute of Technology, Atlanta, GA.

[34] S. M. Swartz, J. Iriarte-Diaz, D. K. Riskin, A. Song, X. Tian, D. J. Willis, and K. S. Breuer, "Wing structure and the aerodynamic basis of flight in bats," in *Proc. of the 45th AIAA Aerospace Science Meeting*, Reno, NV, 2007.

[35] J.-J. E. Slotine and W. Li, *Applied Nonlinear Control*. Prentice Hall, 1991.

[36] M. Goman and A. Khrabrov, "State-space representation of aerodynamic characteristics of an aircraft at high angles of attack," *Journal of Aircraft*, vol. 31, no. 5, pp. 1109–1115, 1994.

[37] D. Peters, S. Karunamoorthy, and W. Cao, "Finite state induced flow models part i: Two-dimensional thin airfoil," *Journal of Aircraft*, vol. 32, no. 2, pp. 313–322, 1995.

[38] J. DeLaurier, "An aerodynamic model for flapping-wing flight," *Aeronautical Journal*, vol. 97, no. 964, pp. 125–130, 1993.

[39] "Q4 hardware in the loop (h.i.l) board," http://www.quanser.com/net/industrial/Systems_and_Products/Prod_Q4_Hardware.aspx, 2009.

[40] P. D. Kuang, M. Dorothy, and S.-J. Chung, "Robobat: Dynamics and control of a robotic bat flapping flying testbed," in *AIAA Infotech @ Aerospace Conference and Exhibit*, St. Louis, MO, Mar. 2011, aIAA Paper 2011-1435.

[41] W. Zhao, B. H. Kim, A. C. Larson, and R. M. Voyles, "Fpga implementation of closed-loop control system for small-scale robot," in *Advanced Robotics, 2005. ICAR '05. Proceedings., 12th International Conference on*, Seattle, WA, Jul. 2005, pp. 70–77.

[42] F. Aubpart and N. Franceschini, "Bio-inspired optic flow sensors based on fpga: Application to micro-air-vehicles," *Microprocessors and Microsystems*, vol. 31, pp. 408–419, Sep. 2007.

[43] S. Murthy, W. Alvis, R. Shirodkar, K. Valavanis, and W. Moreno, "Methodology for implementation of unmanned vehicle control on fpga using system generator," in *Devices, Circuits and Systems, 2008. ICCDCS 2008. 7th International Caribbean Conference on*, Cancun, Mexico, Apr. 2008, pp. 1–6.

[44] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[45] R. Tedrake, Z. Jackowski, R. Cory, J. W. Roberts, and W. Hoburg, "Learning to fly like a bird."

[46] M. Motamed and J. Yan, "A reinforcement learning approach to lift generation in flapping mavs: Experimental results," in *Robotics and Automation, 2007 IEEE International Conference on*, Roma, Italy, Apr. 2007.

[47] P. Kormushev, S. Calinon, and D. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, Taipei, Taiwan, Oct. 2010, pp. 3232–3237.