Replicating Extreme Maneuvers of Anna's Hummingbirds Using a Small Indoor Quadrotor: A Simulation and Proof-of-Concept Approach

Puthalapattu Sukanya
puthalapattusukanya@gmail.com

***Abstract*—This research investigates the capability of a small indoor quadrotor (Crazyflie 2.1) to autonomously execute extreme aerial maneuvers inspired by the courtship display dives of Anna's Hummingbirds (Calypte anna). Utilizing a multi-camera motion tracking system (OptiTrack), the study aims to replicate these high-speed, high-acceleration dives through precise trajectory tracking and control. The primary objective is to assess the quadrotor's ability to reproduce these maneuvers by minimizing the time-scaled root mean square error (RMSE) between the hummingbird and quadrotor trajectories to less than 10 cm, with a standard deviation of 5 cm. The research explores the engineering challenges associated with replicating such complex movements, considering factors like speed, acceleration, and physical constraints. By leveraging simulation tools (Matlab and Simulink) and subsequent proof-of-concept demonstrations, the study seeks to advance the field of unmanned aerial systems (UAS) by improving their maneuverability and understanding the biomechanical limits of extreme aerial maneuvers. Potential applications include enhanced autonomy in navigation, obstacle avoidance, and behavioral playback studies for biological research.**

I. Background and motivation

Animals moving in a real environment face many challenges that are currently unsurmountable for typical engineered devices, including small size, difficult-to-achieve power-to-weight ratios, ability to operate in multiple environments, and the ability to control complex maneuvers even in the presence of environmental disturbances such as wind, water flow, or turbulence, variable lighting, or even attack from predators or conspecific rivals. These alone make them worthy of engineering study, but such interdisciplinary work is not a one-way street.

Biologists may also wish to use engineering to learn how an organism does what it does. As an example, consider the aerodynamic principles of "helicopter" samara seeds such as in maples (*Acer sp.*) and convergently evolved in many other groups. Samaras are able to slow their descent to the group using a spanwise asymmetric weight distribution and wing structure to facilitate autorotation, allowing them longer hang time during which, on rare occasions, they are swept quite far by a lucky gust of wind. The resulting long dispersal distances are quite advantageous for the saplings, who no longer have to compete with the mother tree. The study of the first autorotating seeds in the fossil record made heavy use of engineering techniques such as autorotation theory, consideration of stability from the vertical separation of the center of gravity and the center of pressure, and nondimensional coefficients governing flight . As for applications, researchers in developed a monocopter based off of this concept, and were able to develop control algorithms for actively steering such a device.

Robotic systems are potentially a way to examine form and function in organisms, and organisms are potentially a source of inspiration for improving the robots. In sidewinder-type locomotion, observed in all snakes but most characteristic of the sidewinder rattlesnake (*Crotales cerastes*), was studied by creating a robot to imitate it. In sidewinder locomotion, snakes move across granular media like hot desert sand in a direction lateral to their main body axis; this is accomplished using a combination of body twisting and bending to create moving points of contact with the ground as the body bends. The robot was developed like a snake, and was programmed to copy as closely as possible the movement of the sidewinder. The initial test had success on level ground, but no such luck on steeper slopes. On closer observation of the live snake, it was discovered that the sidewinder used two independent wavelike motions to move instead of just a single wave. Once this was implemented in the robot, the robot was able to navigate just like the sidewinder, and a new form of ground travel was made attainable by robotic machines. This discovery has applications of movement over a variety of different terrain that a typical wheeled robot would not be able to navigate.

I will be attempting to enhance a quadrotor's maneuverability by studying the flight patterns and maneuvers of Anna's Hummingbirds , a 5 hummingbird native to western North America. Male Anna's Hummingbirds perform a display dive in order to win matings with choosy females. The dive reaches speeds above 27.3 and ends with a 9G pullout maneuver (full 3D field kinematics determined in) culminating in a display of red iridescent feathers on his gorget (throat) and a high frequency tweeting noise made by flow-induced vibration of the distal two retrices (outer tail feathers). Such a maneuver would normally cause G-induced loss of consciousness in a human pilot without a G-suit. The difficulty of the maneuver makes it an honest signal of mate quality to Anna's Hummingbird females as it requires skill and power to complete. Thus, it is expected to be difficult for a 27 quadrotor to imitate and a worthy challenge. I intend to discover how feasible the dive maneuvers actually are for quadrotors and explore their operational limits in this respect.

Unmanned aerial systems are becoming increasingly more autonomous for tasks with position control at comparatively low speeds; a guiding research question is how extreme maneuverability can augment their capabilities by building on previous work in quadrotor control. The applications of this research are twofold: it will both help in developing a greater understanding of the physical limitations of extreme maneuverability on quadrotors, the primary engineering problem, and it will also provide a greater idea of the effects of such maneuvers on flying animals. The ability to automate the hummingbird dive maneuvers means that other forms of extreme maneuvers might also be autonomously executed, perhaps providing a toolbox or set of skills to use when a UAS is presented with a maneuvering challenge during a mission. If combined with enhanced sensing abilities, autonomous MAVs would be able to navigate through obstacle-heavy environments with greater speed and ease, allow for evasion or penetration/infiltration, or simplify recovery.

In addition to this, a successful device could be used in behavioral playback studies in which live hummingbirds are presented with a controllable stimulus mimicking the male display dive. If a quadrotor were to execute this dive for a female Anna's Hummingbird, could the quadrotor generate a favorable response from her? If so, it may be possible to alter the pattern to present a super-normal stimulus or probe which aspects of the display are most appealing to her. Such a study in hummingbirds would be novel as biologists have not been able to produce the maneuvers themselves.

II. Problem statement

I aim to execute aggressive maneuvers similar to the display dives of male Anna's Hummingbirds with an autonomous quadrotor platform.

A successful maneuver is defined as a root mean square error (RMSE) calculation of less than 10cm between the time-scaled hummingbird trajectory and the quadrotor trajectory, with a standard deviation gma of 5cm over the entire dataset. The RMSE will be calculated between the x, y, z position data for each pair of data points that share the same time value, t. The hummingbird trajectory will be scaled by a constant factor in time to allow for the physical limitations of the quadrotor–e.g. the quadrotor may only have to travel the desired trajectory at half the speed of the hummingbird to still achieve a successful maneuver. The hummingbird trajectory will also be geometrically dilated to a smaller overall trajectory path in order to physically fit the trajectory inside the available lab space. These are necessary adaptations due to the impressive speed and acceleration capabilities of the Anna's hummingbirds relative to their size , and of course the physical size constraints of the Maury 201 labroom.

As a stretch goal, I will also analyze and replicate a variety of different types of hummingbird dives and maneuvers, to include evasive aerial maneuvers [2], [3]. Success will be determined in the same fashion as described above.

Fig. 1. (a) The five stages of an Anna's Hummingbird dive maneuver, from . (b) A quadrotor using path planning to fly through a thrown hoop, from [1].

III. Literature review

The work in [3] determined the trajectory and body kinematics of four different hummingbird species in an evasive maneuver. This was done by startling the birds while they were hovering, and observing their movements with three high definition cameras to provide a 3D position [4]–[6]. Birds were marked with water-soluble white paint to ease digitization and tracking. I will use a similar optical tracking methods as in [3]–[6], with the added simplifications of being able to install known, infrared (IR) relfective markers for automatic tracking and using a RANSAC algorithm to estimate pose. In addition, the evasive maneuvers studied in [2], [3] could be useful for quadrotors attempting to avoid capture or a counter-UAS drone denial system (e.g. USNA Project Midknight or similar).

The work in obtained fully 3D field kinematics of courtship dives in *C. anna* in order to study extreme locomotor performance in animals. used multiple calibrated high speed cameras and manual digitization to reconstruct the birds' 3D position during dives. Splines were used to estimate acceleration and velocity from positions without undue amplification of noise. also examined wing and tail movements and sounds produced during dives. As a measure of the biomechanical difficulty of the maneuver, estimated the maximum stresses in the humerus during dive pullout; such a quantity would be extraordinarily difficult to measure *in vivo* but in my work there is a possibility of directly instrumenting the quadrotor to examine forces, torques, stresses, and engineering limits.

After obtaining these hummingbird trajectories, an effective method of modeling a quadrotor and conducting trajectory planning is required. In [7], the authors developed metrics and constraints for quadrotor maneuvering performance and

a machine learning workflow. This was achieved through the solving of an optimal control problem offline, and then using a machine learning technique to learn these trajectory solutions with the given constraints. The result was then used to develop online solutions for near-optimal trajectories for a quadrotor. This was done in the x-z plane for point to point and perching maneuvers, as well as joint trajectories. To validate their solution, they flew these optimal trajectories using both Simulink simulations, and proof of concept demonstrations. Since I will be using a quadrotor platform, this paper directly applies to my problem statement as a good reference base that I can use to springboard my exploration into more complex extreme maneuvering. The basis of this work will give me a much more quantitative measure of success in terms of how close my developed trajectories are to an optimal path. [7] is very thorough and provides a clear distinction and improvement on previous work in quadcopter trajectories, especially with regard to the joint trajectory problem. I can build on this by expanding into 3D trajectories instead of just working in a 2D plane, and I can also try to utilize their proxy-based joining method to create a desired path curvature.

In [8], the authors developed a linearized model of a quadrotor in planar motion. The careful process by which the quadrotor dynamics are identified and modeled will be helpful in my own research as I develop my own model for the quadrotor that I will be using. In [8], their modeling method is done for three different linearization methods and each of these is compared to each other by running a Simulink simulation with each controller. Quantities compared include several attributes of the step response, and the actual trajectory of the quadrotor compared to the desired trajectory. This comparison method between the different trajectories is similar to the validation work that I will need to do on my own simulation. As such, this work will help me to better understand ways of determining the accuracy of my trajectory testing in simulation, and in proof of concept demonstration. This paper, while a good starting point for my work, does not attempt to go into more complex maneuvers. These are discussed in greater detail in the following works.

In [9], the development of trajectories and path planning for UAVs was accomplished. They did this by determining the maximum overload, minimum turn radius, and maximum flight endurance of the experimental quadrotors in order to come up with feasible aggressive trajectories. Trajectories had the constraint that they had to follow a sixth order (or lower) polynomial trajectory. Much like my proposed concept, this work develops an attitude and trajectory controller with appropriate initial and final conditions, as well as a boundary "tube" which the quadrotor must stay within for every trajectory. The work in this project is heavily relevant to my proposed work, as they achieve a working simulation of aggressive trajectories with their path-planning algorithm and onboard controllers. I would like to expand on this work by flying a shorter trial with hummingbird-like flight patterns.

Finally, [1] offers some of the closest work to exactly what I am proposing for my own work. The main focus of this paper is to create trajectories for quadrotors in real time in an indoor or constrained environment. They also pay particular attention to the velocity and acceleration vectors of the quadrotor throughout its maneuver. I will also need to be able to achieve these types of measurements from my system, and be able to change my controller to affect them in an appropriate manner in order to fully achieve a trajectory flight path that replicates a hummingbird maneuver. [1] also uses temporal scaling to fly their trajectories at different speeds, which is exactly what I will need to do when and if I find that flying the hummingbird trajectory at full speed is either not possible or extremely dangerous.

IV. Materials and methods

To demonstrate the objective of replicating *C. anna* display dives, I will simulate the system in Matlab and Simulink. This will be followed up with a proof-of-concept demonstration, provided I am able to achieve successful trials in simulation first. The simulation will be more general and will allow easy changing of parameters and control methods to more completely explore the space; simulations will also allow examination of behavior at actual hummingbird flight speeds without risking excessive damage to the hardware. Simulations make simplifying assumptions by necessity, so actual testing and demonstration using real quadrotors–namely the Crazyflie 2.1–were also planned. Unfortunately, due to the inaccessibility of hardware brought about by the stay-at-home orders issued by the DoD in March of 2020, a final hardware demonstration of trajectory flight was never executed. Progress made towards achieving a hardware demonstration is described later in this report.

For my concept demonstrations, I will be modeling the rigid body dynamics of the quadrotor using parameters obtained from the Bitcraze website, and distance measurements taken by calipers. For initial attempts at control, the hover condition is assumed. This is a control linearization region which assumes the attitude of the quadrotor is completely level, i.e. the pitch and roll angles are assumed to be approximately zero degrees during all stages of flight. A basic waypoint position controller will be utilized first in order to ensure controller functionality in simulation and on the hardware itself. A PD controller will be used initially for position control, as this is a ubiquitous controller well understood by students and control theorists. Initial gains will be taken from the simulation [10]

and later adjusted to fine-tune the quadrotor response. Additionally, in order to avoid an excessive attitude command in the event of large position errors, saturation limits will be placed on the attitude command of +/- 45 degrees in order to ensure the quadrotor doesn't over-rotate and accidentally flip over.

The functional block diagram of the overall Crazyflie control system is shown in 2. The laptop is the central node in the control scheme, obtaining position and orientation information from the OptiTrack system, and then relaying this data to the Crazyflie in addition to the next desired position on the trajectory path. The Crazyflie then conducts onboard error

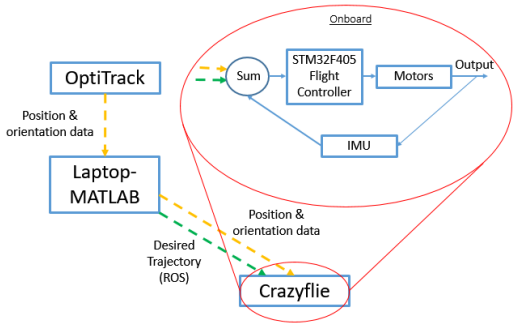

Fig. 2. Functional Block Diagram of the data communication flow concept for the autonomously controlled Crazyflie, with an included onboard controller feedback loop.

calculation and command processing in order to control its position in 3D space.

A. Simulation of bio-inspired dive pullout maneuvers in Matlab

Using the model of the Crazyflie quadrotor described in the Matlab Simulink simulation [10], and hummingbird trajectory data obtained from , several trajectory comparisons were conducted in the x, y, z Cartesian coordinate frame between the true hummingbird trajectory (desired trajectory) and the actual flown trajectory by the simulated Crazyflie. Since the dive trajectories have little deviation in the y coordinate axis, they are compared only in the x and z axes for all relevant error calculations. Simulations were conducted from two different initial quadcopter states. The first simulation was run with the quadcopter starting from rest on the ground. This requires the quadcopter to ascend to the top of the trajectory first and then begin the trajectory flight, which simulates what I would have to replicate for the hardware demonstration. The second initializes the quadcopter at the beginning of the hummingbird trajectory, with an initial velocity determined by calculating the instantaneous velocity of the second data point $\delta x/\delta t$, and $\delta z/\delta t$ (both x and z directions) of the original hummingbird trajectory, where $\delta x = x(t = t_1) - x(t = t_0)$. These velocities were then multiplied by a scale factor based on the simulation speed (e.g. the initial velocities were reduced by a factor of 20 for a simulation trajectory at 20 times reduced speed) in order to maintain a realistic starting velocity at each of the different trajectory speeds.

To determine the accuracy of the trial I will compare the actual flight path of the quadrotor with the desired flight path, and determine a time-scaled root mean square error between the two position vs time datasets. This error will be calculated using the distance formula 1 below:

$$e_p(t) = \sqrt{(x_a(t) - x_d(t))^2 + (z_a(t) - z_d(t))^2} \qquad (1)$$

where $e_p(t)$ is the position error at a specific time step (time t) in the trajectory, and the subscripts a and d represent the actual traveled trajectory and the desired trajectory respectively. The error in the y axis is omitted since the trajectory is 2D in the x-z plane and errors in the y axis are therefore negligible for a stable controller, as is assumed. The error for every time step will be averaged together to determine the root mean square error of the data. Additionally, the standard deviation gma of the position error was calculated using the `std()` function in Matlab. Without any other indications, a successful trial will be considered a root mean square error of less than ten centimeters over the entire trajectory, with a gma value of less than 5cm. Since the hummingbird trajectory is time-scaled down to only a fraction of its true speed, the quadrotor will be responsible for marking every position at the time it is supposed to be located at that position, therefore limiting the effects that motor operation constraints, e.g. thrust saturation, as a possible source of error in the trajectory flight.

After the initial simulation runs there will likely be large position errors over the trajectory flight. As such, it will be desirable to tune the control gains in order to achieve a lower root mean square error. The simulation consists of two separate levels of control. The first level is a high-level PD controller concerning the x and y position of the quadcopter, and the second level is the lower-level PID controller that actually determines the pitch and roll angle of the quadcopter, as well as its vertical (z) position. The output of the first level controller is factored as input into the second level controller, along with the quadcopter state. Separate control gains are used for each degree of freedom, i.e. the control gains for the pitch angle of the quadcopter are completely independent of those for the roll angle, altitude, yaw angle, etc. In all, there are eight different control gains that characterize the behavior of the quadcopter through the 2D x-z hummingbird trajectory: the proportional and derivative control gains for the x position (K_{px}, and K_{dx}), and the proportional, derivative, and integral control gains for both the pitch θ and altitude z (respectively, $K_{p\theta}$, $K_{d\theta}$, $K_{i\theta}$, K_{pz}, K_{dz}, K_{iz}).

In general, the effects of changing the proportional control gain will increase the responsiveness of the system with regard to the parameter effected by the controller. As the proportional gain is increased, overcontrolling is noted as oscillations around the desired state grow with a continued gain increase. An increase in the derivative control gain will help to counterbalance this effect to improve the overall stability of the system and reduce overshoot, while an increase in the integral control gain will reduce/eliminate oscillations around the desired settling state, bringing the steady state error to zero. As such, with regard to the effect of the control gains on the simulation output, I expect that increasing both the proportional and derivative gains should help to quicken the response of the quadrotor, and achieve a smaller root mean square error when flying the desired trajectory while remaining stable.

Several methods of gain tuning were attempted, with varying levels of success, to reduce the root mean square error of the quadrotor flight. These methods include manual gain tuning, nonlinear optimization routines, and marginal analysis. Manual gain tuning was attempted first. Initially, the proportional gain was increased separately for each of the

three characteristic control loops (x, z, and θ) until signs of overcontrol/instability were noticed in the response (mostly by large oscillations around the desired control point). Then the derivative control gains were increased for each of these controllers to quell the unstable oscillations. Manual gain tuning was successful at reducing the root mean square error, and the detailed/numerical results are recorded in the Results section of this report.

After the manual gain tuning was unable to achieve the desired performance for the quadrotor, I began looking into different ways to attempt to optimize the controller gains to achieve a minimum root mean square error. I decided to use the Matlab function, `fmincon()` to optimize the controller gains, which finds the minimum value of a constrained nonlinear multivariable function through an iterative process. Before attempting to use the function for my problem, I completed the example problem listed in the Matlab documentation for `fmincon()` to minimize Rosenbrock's function. This test allowed me to orient myself with using the `fmincon()` tool, and helped to build my objective function for use on my simulation.

In defining my optimization problem, the eight characteristic control gains (K_{px}, K_{dx}, $K_{p\theta}$, $K_{d\theta}$, $K_{i\theta}$, K_{pz}, K_{dz}, and K_{iz}) are the decision variables, using the values I had determined during my manual gain tuning as the initial point $x0$. My output variable (what I'm trying to optimize) is the root mean square error between the desired and actual trajectory flown by the quadcopter in the simulation. My objective function was the piece that was a little less straightforward, as it required multiple lines of Matlab code to achieve. I first set up the appropriate simulation parameters based on the changes to the decision variables made by the `fmincon()` optimization routine. Then, I ran the simulation for a set time of 20 seconds to allow sufficient time for the quadcopter to complete the trajectory flight. Following this, I had to determine the start and end points of the trajectory flight. The start point was simply the first index of the simulation output, since the quadcopter started at the top of the trajectory. The endpoint was determined by the index of the output where the commanded position at that index was the exact same as the commanded position two iterations after that point. (The simulation continues to command the last timeseries position provided by the input until the simulation time runs out, therefore a repeated position command in adjacent time steps means that the dive trajectory has ended.) Once the starting and ending index of the simulation output were determined, I was then able to use the error calculation described by 1 to determine the root square error of each time step, and average these values to find the root mean square error, completing my objective function.

In my first attempts at running the optimization routine I left the decision variables only constrained by the basic requirement that they had to be nonzero and positive. The optimizer would complete the first iteration at the initial point with no problems, but as soon as it changed the gains for the second iteration, it caused the system to become unstable and the simulation crashed with multiple zero-crossing errors. In attempts to remedy this, I changed several different simulation settings to ignore zero crossing errors and change the simulation step size, all to no avail. I then tried to add constraints to the upper bound of the decision variables (they were already constrained on the lower bound to be greater than zero), in an attempt to prevent the simulation from becoming unstable while `fmincon()` tweaked the control gains. This was successful in that the optimization routine was able to run its course; however, the result only made small changes to the initial control gains (less than a tenth of a percent change) which resulted in no significant effect on the output variable, the root mean square error. The simulation is therefore seemingly unable to support optimization routines as it is currently written.

After the attempt at control gain optimization proved unsuccessful, I decided to conduct a marginal analysis in order to determine which variables would contribute most to the reduction of the root mean square error. A marginal analysis test simply makes a small change to a decision variable, and then records the overall benefit gained and cost incurred from making that small change. Running this analysis provides insight into how this current controller scheme could be tuned further to achieve greater success at flying the quadcopter through the desired trajectory. I again used the gain values obtained from manual tuning as my initial point, and wrote a Matlab script to iterate through each gain value and run the simulation on three separate cases for each: a run with the gain at the initial point, a run with the gain decreased by five percent, and a run with the gain increased by five percent. After each simulation, error data was collected and saved for processing.

B. Demonstration of bio-inspired dive pullout maneuver using Crazyflie quadrotor

Hardware implementation is essential to verifying that the findings of the simulation are accurate. The path to a successful hardware demonstration was a multi-step process. We needed to test the capability of the crazyflie to operate under manual control, open loop control, and then finally introduce closed-loop control of the crazyflie using an optical motion tracking system. As such, the hardware components necessary for this experiment include a fully functional Crazyflie quadrotor (Bitcraze, Malmö, Sweden), and an OptiTrack system (NaturalPoint Inc., Corvallis, OR). The quadrotor will be the object of the experiment, and the OptiTrack system serves as a highly accurate way to obtain position data for the Crazyflie in flight. It achieves this using visual information from nearly 20 cameras staged around the outside of the testing area. Initial experimentation will be conducted indoors to minimize any aerodynamic noise, although future trials could test controller robustness by introducing environment disturbances. Several batteries for the Crazyflie are needed to ensure sufficient trial and testing periods, and approximately five OptiTrack visual markers must be affixed asymetrically on each Crazyflie drone to ensure that it is able to be detected by the OptiTrack system

and that its state is measured appropriately. These marker additions will be taken into account in simulation first in order to ensure readiness to counteract any effect they may have on the dynamics of the quadrotor in the feedback loop.

As an initial test of hardware, I used the `cfclient` software to link to the crazyflie from a linux machine, and test manual control of the crazyflie using a Logitech USB gamepad (Logitech F310; Lausanne, Switzerland) and a Crazyradio sending velocity and thrust commands to the Crazyflie. This allowed me to gain a basic understanding of how the crazyflie communicated with the linux machine over radio control, and how this might be implemented in a python script. I then tested the motion tracking system capabilities by obtaining automatic 3D position tracking data via the OptiTrack. The OptiTrack system worked independently from this workstation, and recorded x, y, z position data while logging the time t that each data point was recorded at.

The next step towards a full hardware demonstration was to test autonomous control of the Crazyflie. To support this, I cloned the whoenig crazyflie_ros github repository which utilizes a ROS framework to run multiple python and C++ scripts that send appropriate commands to the Crazyflie based on the tasking in the files that are run. The demo package of the repository contained a plethora of different example scripts and launch files that would provide the capability needed to complete the hardware testing that I desired, including options for manual joystick control, open-loop hovering control, and closed-loop control using both vicon (an optical motion tracking system) and vrpn connection to an alternate motion tracking system such as OptiTrack. I progressively ran through several of the lower level control demo packages, testing first manual control with the USB gamepad using ROS, followed by open-loop control (which ultimately resulted in the Crazyflie crashing into the ceiling due to no closed-loop feedback). Through adaptation of the vrpn control demo scripts, I was able to achieve a successful implementation of a closed-loop hover at a desired 3D location. This worked similar to waypoint control, but only consisting of two waypoints (i.e. the starting and ending point). This meant that waypoint trajectory flight would be possible in the Maury lab with the provided controller. Unfortunately, due to the conversion to online classes, this was as much as we were able to achieve in terms of a hardware demonstration of capability. My work up to this point has been saved in a github repository, and should be easily accessible and run-able by any student wishing to use the combined Crazyflie OptiTrack system to conduct waypoint flight hardware demonstrations. I have also left fairly detailed instructions in a google document, which is available on the marcello-2020 shared drive.

V. Results

A. Initial simulations of bio-inspired dive pullout maneuvers in Matlab

Using the modeling framework of [10], 4 trials were run at varying speed reductions: a trial run at the actual hummingbird trajectory's speed, a trial at one fifth the hummingbird trajectory's speed, a trial at one tenth speed, and a trial at one twentieth speed relative to the full scale, full speed trajectories in . The results of those tests are shown in 3, 4, 5, and 6. Distance error calculations for the four trials are shown in I.

For these experiments, the attitude angle command saturation limits were increased to allow for larger command inputs to take advantage of a wider range of pitch angles. The simulated quadrotor was able to follow the one twentieth speed reduction the closest, and had the smallest RMSE, while the trial at one fifth the speed was incredibly behind the desired trajectory path. The trial at normal speed showed how inferior this control method is at executing high-speed extreme maneuvers, as the quadrotor was barely able to begin accelerating by the time the simulation called for the quadrotor to be completing the trajectory. As such, further experimentation with this controller at actual speed was discontinued, as it seems more relevant to focus on improving the control at slower speeds first before attempting improvment at full speed.

TABLE I
DISTANCE ERROR CALCULATIONS FOR EACH TRIAL (ACTUAL SPEED, ONE FIFTH SPEED, ONE TENTH SPEED, AND ONE TWENTIETH SPEED). POSITION ERROR IS REPRESENTED BY THE VARIABLE e_p

Trial	$\bar{e_p}$,	gma_{e_p},	$\max(e_p)$,
1	6.6888	4.3926	13.5984
1/5	3.9415	1.8710	6.3231
1/10	2.1757	0.7700	2.9967
1/20	1.0443	0.3328	1.3603

B. Improved control simulations of bio-inspired dive pullout maneuvers in Matlab

As discussed in the Methods section, several different attempts were made at improving the control of the quadrotor. The only method that proved successful enough to show meaningful visual results was the manual gain tuning. The results of which, for trials at one fifth speed, one tenth speed, and one twentieth speed, are compiled into the three figures shown below in 7, 8, and 9. The new error calculations for these trials are also displayed in II. Additionally, the results of marginal analysis at each speed trial are displayed in III. Of note, the marginal analysis indicated that, for the trial at one twentieth the speed of the hummingbird trajectory, any changes made to the gain values (of five percent) only increased the overall root mean square error. This is represented in the table by displaying tacs across the row.

C. Initial hardware testing of Crazyflie manual control and OptiTrack tracking and pose estimation

Initial hardware testing for flying the Crazyflie and obtaining automatic 3D position tracking data via the OptiTrack was successful, as shown in 10. In general, the next step would be to upgrade from manual flight of the Crazyflie to autonomous flight, as discussed in the Methods section. Due to the abrupt and unexpected inaccessibility of lab equipment, I was also

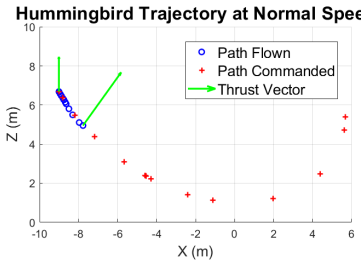

Fig. 3. Crazyflie attempted trajectory flight of the hummingbird trajectory at its actual speed.

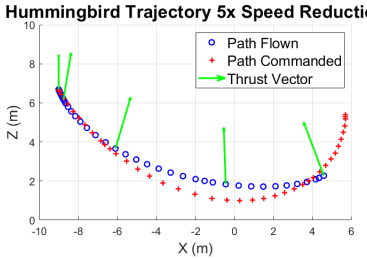

Fig. 4. Crazyflie attempted trajectory flight of the hummingbird trajectory at one fifth its actual speed.

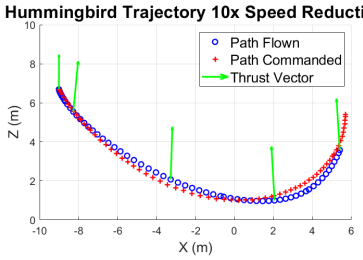

Fig. 5. Crazyflie attempted trajectory flight of the hummingbird trajectory at one tenth its actual speed.

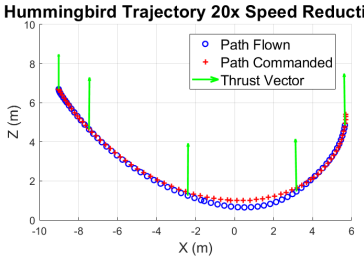

Fig. 6. Crazyflie attempted trajectory flight of the hummingbird trajectory at one twentieth its actual speed.

unable to record data for the waypoint-hover flight demos to display in this report.

VI. Discussion

A. Simulations show quadcopter limitations when attempting extreme maneuvers at full speed

4–6 show that as the speed was reduced, the quadrotor was able to more closely follow the desired hummingbird trajectory. A look at the RMSE calculations in I and II shows that this was indeed the case. The quadrotor was unable to obtain an RMSE (the mean distance error value) of less than one meter before gain tuning, even when flying at one twentieth the hummingbird's true speed. At this slower speed, the error is still exorbitantly high at a value ten times greater than my definition of a successful trajectory flight, and showed a standard deviation that was more than five times my defined value for a successful flight. After gain tuning, small improvements were made to the overall reduction of the RMSE value at each trial; however, they were still several times higher than I had defined for a successful trajectory flight. Although control gain optimization may lead to a successful trial at slower speeds, several changes will likely need to be made to the structure of the controller in order to obtain desireable results at the higher speed trials. This simulation assumes a linearization region about the "hover" state, which doesn't lead it well to controlling the quadrotor accurately through this extreme maneuver.

B. Improved control in simulation

The purpose of improving the controller given by the simulation is to find a way to reduce the root mean square error between the desired trajectory path and the actual trajectory path flown by the quadrotor. Several attempts were made at error reductions, with limited success. The changes that I made to the control gains did improve the system, and the results of the marginal analysis even suggested that it could be enhanced further; however, this controller is simply not sufficient for executing the extreme maneuvers outlined in this report. The main issue with the current controller is that it only takes into account one timestep at a time and iteratively makes adjustments based off of each new incoming positoin command. In this way, it fails to anticipate the necessity of large acceleration requirements (namely at the top of the dive when the hummingbird is diving downward, and bottom of

TABLE II
Distance error calculations with improved control using manual gain tuning for each trial (one fifth speed, one tenth speed, and one twentieth speed). Position error is represented by the variable e_p

Trial	$\bar{e_p}$,	gma_{e_p},	$\max(e_p)$,
1/5	3.1216	1.5601	4.8006
1/10	1.6744	0.6758	2.5373
1/20	0.8251	0.3448	1.3153

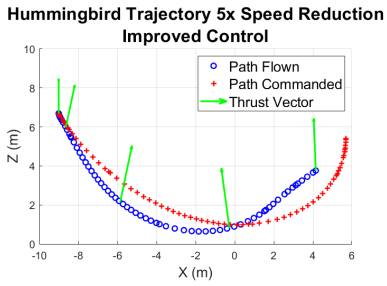

Fig. 7. Crazyflie attempted trajectory flight of the hummingbird trajectory at one fifth its actual speed, with improved control provided by manual control tuning.

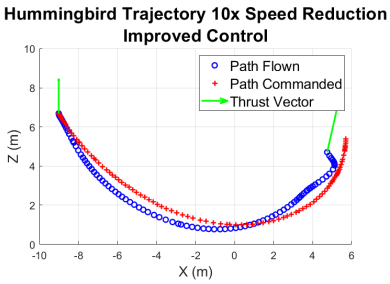

Fig. 8. Crazyflie attempted trajectory flight of the hummingbird trajectory at one tenth its actual speed, with improved control provided by manual control tuning.

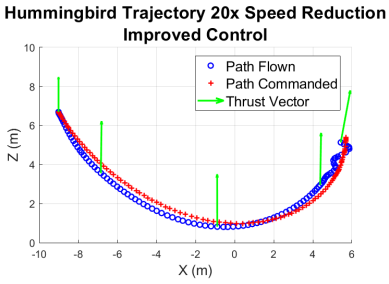

Fig. 9. Crazyflie attempted trajectory flight of the hummingbird trajectory at one twentieth its actual speed, with improved control provided by manual control tuning.

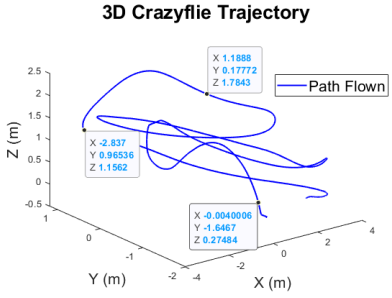

Fig. 10. 3D position data of a manually-controlled flight with the Crazyflie, captured with OptiTrack, and plotted in Matlab.

the dive when the hummingbird pulls 9gs to soar up out of the dive) to properly pull the quadcopter out of the dive and follow the hummingbird trajectory. As such, a completely new controller would need to be outfitted on the simulation in order to get the results that I desire.

C. Experimental flight demonstration next steps

The next step to the autonomous experimental demonstration will involve the conversion of the controller code used in the Matlab simulation to something that can be used on the Crazyflie hardware. The controller will therefore be written in Python, as it is similar to Matlab and can be used on the Crazyflie with the ROS package. The other option would be to use C programming language, but since the ROS demo I'm using is in Python it would be wiser to stay consistent with that language for the hardware. Therefore Python will be used to implement the path-planning control algorithm on the Crazyflie. To fly the trajectory, the quadrotor system will follow the feedback loop portrayed in 2 using ROS as the primary communications manager between the different nodes. The laptop will be running ROS in linux, which will establish the OptiTrack System as a node, itself as a node, and the Crazyflie as the third node. The laptop node will recieve position and orientation data of the Crazyflie from the OptiTrack node accurate to $\approx 0.5mm$, and it will send this information along with the next desired trajectory position to the Crazyflie node. The Crazyflie will then process this information onboard, along with pose feedback from the IMU accelerometers and gyros to determine a position error and the correct control response due to both the position error, and the derivative of that error–i.e. velocity error (not shown in the functional block diagram). The position controller will be designed using the same PD controller as in the simulation (for the higher level) and PID controller (for the lower level control), and the controller gains will be tuned appropriately based on the gain tuning done in simulation to reduce the overall error in position over the entire trajectory. Once the Crazyflie has completed flying the trajectory, the saved position data obtained from the OptiTrack readings will be compared to the desired trajectory path in post-processing to determine the flight accuracy, using the same calculations as in the analysis of the simulated flights.

TABLE III
MARGINAL ANALYSIS OF CONTROL GAIN TUNING FOR EACH TRIAL (ACTUAL SPEED, ONE FIFTH SPEED, ONE TENTH SPEED, AND ONE TWENTIETH SPEED). EACH ROW SHOWS WHICH GAIN REDUCED THE ROOT MEAN SQUARE ERROR THE MOST, THE PERCENTAGE CHANGE IN THAT GAIN TO ACHIEVE THIS RESULT, AND THE PERCENT CHANGE IN THE ROOT MEAN SQUARE ERROR THAT RESULTED FROM MAKING THAT CHANGE.

Trial	Gain	$\Delta, \%$	$\Delta\bar{e_p}, \%$
1	K_{px}	$+5$	-1.266
1/5	$K_{p\theta}$	-5	-2.201
1/10	K_{pz}	$+5$	-1.188
1/20	-	-	-

D. Additional thoughts and intermediate conclusions

The research discussed in this paper is concerned with the replication of the highly aggressive aerial maneuvers of the Anna's Hummingbird on an autonomous quadrotor platform. Specifically, this research aims to determine the extent to which these maneuvers are possible through analysis of a time-scaled root mean square error between the bird and quadrotor trajectories. The analysis and experimentation discussed previously in both simulation and as hardware demonstration support that flying mock hummingbird trajectories with the Crazyflie quadrotor is indeed possible at lower speeds, however the extent to which this is feasible is still unknown. Additional controller tweaking and design will be necessary to enhance the ability of the Crazyflie to more closely follow a *C. anna* trajectory at speeds closer to the actual hummingbird speed. The final proof of concept demonstration will utilize an OptiTrack motion capture system to ensure a high degree of accuracy is recorded for the quadrotor trajectory, and the experiment will be conducted inside to minimize any possible disturbances in the air and therefore view maximum controller performance and future potential in less-controlled environments.

Novelty: This research is the first attempt to autonomously mimic a hummingbird dive trajectory, and it has promise in being successful at developing a greater understanding of the limits of extreme maneuverability in our currently utilized UAV quadrotors. Due to resource constraints encountered towards the end of the semester, and difficulties with running optimization routines with the developed simulation, this research has been inconclusive in determining whether or not a crazyflie quadrotor can successfully fly the same dive trajectory as the *C. anna* .

Fig. 11. Budget. Disclaimer: Labor and overhead costs are estimated only for EW502 training purposes and do not actually reflect real costs that would be supported by project sponsors. Additionally, there are no new costs predicted for the spring semester.

LABOR	Category	Hours	hourly rate	Cost
	Midshipman	336	$25	$8,400
	Faculty	64	$60	$3,840
	Staff	45	$40	$1,800
Labor Sub-total				$14,040
OVERHEAD	**Category**	**Base Amount**	**Rate**	**Cost**
	Fringe Benefits	$14,040	35%	$4,914
	Facilities	$14,040	50%	$7,020
	General Services	$14,040	15%	$2,106
Overhead Sub-total				$14,040
MATERIALS	**Category**			**Cost**
	In-stock Items			$817
	New Items			$820
Materials Sub-total				$1,637
TOTAL COST				$29,717
OUT-OF-POCKET COST				$820

VII. Conclusion

In current standing, my research has fallen behind the original schedule I had set out to complete in my EW502 proposal. This is due mostly to the fact that I was unable to utilize most of MIDN Canlas' work to jumpstart the hardware aspect (with the exception of the Matlab code that interfaces with the OptiTrack motion capture system, which has been a huge timesaver). Therefore, I have been forced to learn more about ROS than I had anticipated in order to start autonomous trajectory flying, and some of my research time had to be repurposed to searching for a suitable demo code to try and autonomously fly the Crazyflie. Even though I am behind schedule, I am still projected to complete my research aims by capstone day, as my original proposed research left plenty of buffer room to account for potential delays and setbacks in the early stages of development.

The biggest hurdle I will face going into next semester will be using ROS to integrate the autonomous hardware control of the Crazyflie. This will require communications between OptiTrack, the linux laptop, and the Crazyflie itself, with the linux machine running the ROS code and acting as the main hub. This process will not be easy to achieve, as different communication protocols are likely required to integrate these devices. Additionally, development of a new controller design for the Crazyflie will also be a highly time-intensive aspect of this project. It will involve a lot of sifting through previous research and replicating that same design process for my own application with the Crazyflie.

A. Remarks on budget

I have purchased and recieved all items that cover the originally proposed $570 new item cost with the exception of a second additional Crazyflie 2.1 that was originally budgeted for, but I have not requested it yet for my research as I don't predict I will need it considering I have four old Crazyflie 2.0 models to use in addtion to the new 2.1 model I obtained with research funds. The increase in the total out-of-pocket cost is due to an approximately 50 increase added on to my original anticipated budget to account for any last-minute must-haves such as a battery charger that can charge multiple Crazyflies at once (will become more applicable as I get into more hardware testing). The in-stock item cost is derived from an approximated worth of $198 for each of the four Crazyflie 2.0's that I'm using, as well as an additional $25 for the use of old OptiTrack markers and the 3D printed rotor-guards.

References

[1] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011, pp. 2520–2525.

[2] K. M. Sholtis, R. M. Shelton, and T. L. Hedrick, "Field flight dynamics of hummingbirds during territory encroachment and defense," *PLOS ONE*, vol. 10, no. 6, pp. 1–20, 06 2015.

[3] B. Cheng, B. Tobalske, D. Powers, T. Hedrick, S. Wethington, G. Chiu, and X. Deng, "Flight mechanics and control of escpae manoeuvres in hummingbirds. i. flight kinematics," *Journal of Experimental Biology*, vol. 119, no. 22, pp. 3518–3531, 2016.

[4] T. L. Hedrick, "Software techniques for two- and three-dimensional kinematic measurements of biological and biomimetic systems," *Bioinspiration & Biomimetics*, vol. 3, p. 034001, 2008.

[5] D. H. Theriault, N. W. Fuller, B. E. Jackson, E. Bluhm, D. Evangelista, Z. Wu, M. Betke, and T. L. Hedrick, "A protocol and calibration method for accurate multi-camera field videography," *Journal of Experimental Biology*, vol. 217, pp. 1843–1848, 2014.

[6] B. E. Jackson, D. J. Evangelista, D. D. Ray, and T. L. Hedrick, "3d for the people: multi-camera motion capture in the field with consumer-grade cameras and open source software," *Biology Open*, vol. 5, pp. 1334–1342, 2016.

[7] T. Tomić, M. Maier, and S. Haddadin, "Learning quadrotor maneuvers from optimal control and generalizing in real-time," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 1747–1754.

[8] F. Sabatino, "Quadrotor control: modeling, nonlinear control design, and simulation," Master's thesis, KTH Royal Institute of Technology in Stockholm, 2015.

[9] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-D complex environments," *IEEE Robot. Autom. Lett.*, vol. 2, no. 3, pp. 1677–1695, 2017.

[10] D. Hartman, K. Landis, M. Mehrer, S. Moreno, and J. Kim, "Quadcopter dynamic modeling and simulation (quad-sim) v1.00," https://www.mathworks.com/matlabcentral/fileexchange/48053-quad-sim, 2014.