BIO-INSPIRED SENSING AND CONTROL OF MICRO AERIAL VEHICLES (MAVS)

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Hengye Yang

May 2023

BIO-INSPIRED SENSING AND CONTROL OF MICRO AERIAL VEHICLES (MAVS)

Hengye Yang, Ph.D.

Cornell University 2023

Natural phenomena, such as thermal soaring of birds, gravitational settling of inertial particles in turbulent flows and acrobatic feats performed by flying insects, can inspire interesting and sometimes useful solutions to the sensing and control problems of micro aerial vehicles (MAVs). This dissertation covers novel biologically inspired MAV sensing and control approaches that are efficient and robust in the presence of unforeseen wind disturbances and modeling uncertainties. At first, this dissertation presents a new feedback control approach inspired by experimental studies on particle transport that have recently illuminated particles' ability to traverse homogeneous turbulence through the so-called fast-tracking effect. While in nature fast-tracking is observed only in particles with inertial characteristics that match the flow parameters, the new fast-tracking feedback control approach employs available propulsion and actuation to allow the vehicle to respond to the surrounding flow in the same manner as ideal fast-tracking particles would. The resulting fast-tracking closed-loop controlled vehicle is then able to leverage turbulent flow structures, such as sweeping eddies, to reduce travel time and energy consumption. The fast-tracking approach is shown to significantly outperform existing optimal control solutions, such as linear quadratic regulator and bang-bang control, and to be robust to changes in the vehicle characteristics and/or turbulent flow parameters. Furthermore, since this fast-tracking control design requires prior knowledge of the

turbulent flow parameters, this dissertation presents a novel approach of using noisy on-board measurements to estimate the flow parameters via the sparse identification of nonlinear dynamics (SINDy) method.

In addition to particle transport theory, MAV sensing and control strategies can be extracted from biological neural systems. Since spiking neural networks (SNNs) encode information in sequences of spike times, spike train decoding is considered one of the grand challenges in reverse-engineering neural control systems as well as in the development of neuromorphic controllers. Therefore, this dissertation presents a novel relative-time-kernel-based spike train decoding approach that accounts for not only individual spike train patterns, but also the relative spike timing between neuron pairs in the population. Using the data collected in hawk moth's flower tracking experiments, the new spike train decoding method allows us to uncover the precise mapping from the spike trains of ten primary flight muscles to the resulting forces and torques on the moth body. The new relative-time-kernel-based spike train decoder significantly improves the prediction of the resulting forces and torques when compared to the existing instantaneous-kernel-based and rate-based decoders.

Finally, inspired by the insect's flapping flight control strategies, this dissertation presents a novel two-phase adaptive full-envelope SNN control design for flapping-wing micro aerial vehicles (FWMAVs) that is able to learn and adapt to unmodeled uncertainties online. During the offline learning phase, populations of spiking neurons are trained by supervised learning to approximate a gain-scheduled proportional-integral-filter (PIF) compensator developed to stabilize the ideal vehicle dynamic model. The online learning phase improves the performance subject to actual vehicle dynamics by incrementally updating the neural connection weights via policy gradient reinforcement

learning (PGRL). This two-phase adaptive SNN control design is then implemented for the control of a simulated insect-scale flapping-wing robot known as RoboBee over its full flight envelope. The adaptive SNN controller is shown to outperform a benchmark non-adaptive SNN controller when the RoboBee is commanded to conduct a full range of maneuvers in the presence of significant uncertainties, such as parameter variations, unmodeled dynamics and measurement errors, as well as actuator failures. The bio-inspired sensing and control approaches presented in this dissertation can be potentially implemented on the next generation of smart, agile and highly adaptive MAVs.

BIOGRAPHICAL SKETCH

Hengye Yang is a PhD student in the Laboratory for Intelligent Systems and Controls (LISC) at Cornell University. He received his B.S. degree in Energy and Power Engineering from Huazhong University of Science and Technology (HUST) in 2017, where he was an undergraduate researcher in the State Key Laboratory of Coal Combustion (SKLCC). In LISC, he functions as a leader in collaborations with other students and faculties at Cornell, Georgia Tech and Harvard to implement his developed control algorithms on real flapping-wing robots and drones. His research interests include optimal and adaptive control theory, spiking neural networks, and unmanned aerial vehicle (UAV) navigation and control, focusing particularly on the intelligent sensing and control of airflow-robot interactions. As a 2021 Cornell Commercialization Fellow, he was selected as the Entrepreneurial Lead (EL) for a National Science Foundation (NSF) Innovation Corps team to explore the commercialization potential of the UAV control techniques developed by him in LISC.

Dedicated to my family.

ACKNOWLEDGEMENTS

First of all, I would like to thank all my lab mates in the Laboratory for Intelligent Systems and Controls (LISC) at Cornell University for the unconditional support and enlightening discussions on my research. I also want to thank my advisor, Prof. Silvia Ferrari. I wouldn't have been able to get this far without her guidance and encouragement. Additionally, I would like to thank the collaborators, Prof. Gregory Bewley at Cornell University and Prof. Simon Sponberg at Georgia Tech, for their timely feedback on our cross-disciplinary research projects. Furthermore, I want to thank Prof. Qi Li and Prof. Gregory Bewley again for agreeing to join my PhD committee and giving me useful advice on my research. I would also like to thank my best friends, Junan Chen and Ji Chen, for accompanying and supporting me on the worst days of the pandemic. Finally, I want to thank my loving family, especially my parents and grandparents, for unconditionally supporting me when I made the decision to pursue my PhD at Cornell University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Sensor-laden micro aerial vehicles (MAVs) have been widely used to assist humans to gather information about targets autonomously in hazardous environments [109, 166, 82]. Due to their small size and light weight, MAVs are not only safe enough to operate near humans, but also extremely agile to avoid obstacles and visit confined spaces inaccessible to vehicles of larger sizes [136, 24]. In particular, bio-inspired flapping-wing micro aerial vehicles (FWMAVs) generate lift much more efficiently than typical aerial vehicles such as fixed-wing aircraft and rotorcraft at micro scale [119, 49, 28]. However, these micro-scale aerial vehicles are extremely sensitive to external disturbances, such as wind gusts, and unmodeled uncertainties, such as parameter variations, measurement errors and control failures, and therefore can easily become unstable or even undergo damage on the fly [218, 78, 144]. Consequently, robust and efficient sensing and control approaches that can achieve desired performance over the full flight envelope in the presence of unforeseen wind disturbances and unmodeled uncertainties have yet to be developed for MAVs.

There is significant precedent for tackling air-vehicle navigation and control problems in constant winds [127, 38], Dryden wind turbulence [202, 197] and thermals [158]. However, most existing approaches rely on accurate modeling of aerodynamic effects, whereas the influences of wind on the vehicle are treated as disturbances. These control designs focus on rejecting such disturbances in the closed loop, but will fail when the wind disturbance exceeds certain physical threshold limit especially during aggressive and rapid air-vehicle maneuvers. In addition to wind disturbance compensation, air vehicles, such

as gliders and fixed-wing aircraft, have been demonstrated to be capable of harvesting energy from surrounding air flows and navigating in highly turbulent environments using less time and energy [158, 7, 68]. However, these existing energy-harvesting flight control designs focus primarily on developing optimal and adaptive flight policies in the presence of different flow regimes, which not only rely heavily on the precise prior prediction of turbulent flow structures, but also require large computational cost [105].

At first, this dissertation presents a new feedback control approach inspired by turbulent particle transport theory [18] that allows the air vehicle to follow the ideal response prescribed by the fast-tracking effect in the closed loop via implicit model following (IMF) in Chapter 2 [223]. The fast-tracking control (FTC) approach has the distinct advantage of enabling the vehicle to fly within advantageous tailwinds more often than with existing control methods and avoid adverse headwinds automatically, thus reducing the cost of time and energy to traverse the turbulent flow. The energy-harvesting potential of this new control design is demonstrated by considering two benchmark control problems known as minimum-energy and minimum-time problems. The FTC approach is shown to outperform two classic optimal control solutions obtained using linear quadratic regulator (LQR) and bang-bang control (BBC) theory. The novelty of this fast-tracking-particle-inspired flow-aided control approach is that it only requires prior knowledge of a few key flow parameters, including the mean velocity, vortex length scale and vortex time scale, that characterize the turbulent flows instead of relying on the global knowledge of the entire wind velocity field [160, 8].

As an essential further step, this dissertation presents a new approach to

estimate the flow parameters that the aforementioned FTC approach requires prior knowledge of using the onboard measurement data obtained from the aerial vehicle traversing turbulent flows in Chapter 3 [224]. Estimating the flow structures and velocity profiles of wind and ocean currents helps to guarantee the safety, robustness and efficiency of vehicle navigation and control in highly turbulent environments [217, 106, 66, 156]. People have already successfully estimated the velocity profiles of wind and ocean currents based on vehicle's corresponding dynamic responses [106, 156]. Without prior information of the exact flow parametric model, the novelty of our computationally efficient flow parameter estimation method is to only estimate the key flow parameters by determining and analyzing the weights that represent which user-defined candidate functions are active in the unknown nonlinear vehicle dynamic equations via the so-called sparse identification of nonlinear dynamics (SINDy) method.

In addition to wind disturbances, bio-inspired MAVs especially those with fragile flapping wings are highly sensitive to unexpected variations in their physical parameters, dynamic characteristics and actuator effectiveness [38, 35, 34]. Even though all these uncertainties make the control design extremely challenging, many intelligent flight control algorithms have been developed for FWMAVs in recent years [38, 36, 117, 40, 30, 29, 31, 131, 81]. For example, an adaptive control design consisting of a position feedback controller and a neural-network-based attitude controller is proposed via a hierarchical framework, and allows the controlled FWMAV to accomplish longitudinal trajectory tracking [81]. However, most existing FWMAV control approaches either rely on accurate modeling of the vehicle dynamics or apply to a limited set of maneuvers such as hovering, longitudinal and lateral flight, and therefore are not applicable to the full-envelope control of FWMAVs in the presence of unmod-

eled uncertainties. Significantly, people have already developed reconfigurable fault-tolerant flight control systems for fixed-wing aircraft that can adapt to uncertainties, and achieve desired control performance over the full flight envelope [186, 142, 116, 6]. Neural networks are found to be particularly useful for the reconfiguration of adaptive control systems, because their connection weights can be reflexively updated online according to the observed difference between desired and actual system responses [187, 186, 52, 56, 58, 113]. Spiking neural networks (SNNs) closely mimic natural neural systems by transmitting discrete pulses of information only when the spiking neuron's membrane potential reaches a threshold [72, 198, 114]. Therefore, SNN-based controllers can be potentially implemented on power-efficient, biologically inspired neuromorphic chips that FWMAVs can be easily equipped with, and tend to substitute classical neural-network-based design approach in modern FWMAV control [36, 38, 89, 90, 60].

However, since SNNs encode information in sequences of spike times, the output from the spiking neurons must be decoded to be useful in representing continuous-time functions for the control input [38, 231]. Various spike train decoding approaches, such as rate coding [2, 88], temporal coding [102, 172, 84] and kernel-based algorithms [137, 140, 169, 170, 139], have been proposed in recent years. However, most existing approaches only capture the information encoded in firing rate, spike counts or exact spike timings, and therefore will not perform well when neurons correlate with each other [153]. In Chapter 4, this dissertation presents a novel relative-time kernel design that considers not only the precise spike timing information from individual neurons, but also the relative spike timing information between neuron pairs [226]. Using the data collected in the moth's flower tracking experiment, we demonstrate the impor-

tance of relative spike timing information for neural control, and uncover the precise mapping from the spike trains of ten primary flight muscles to the resulting forces and torques on the moth body. The new relative-time-kernel-based neural decoder is shown to significantly outperform existing instantaneous-kernel-based and rate-based decoders in force and torque predictions.

Finally, inspired by insect's flapping flight control strategies, this dissertation presents a novel two-phase adaptive SNN controller for FWMAV full-envelope flight that can learn and adapt to unmodeled uncertainties in real time via offline supervised learning (SL) and online policy gradient reinforcement learning (PGRL) methods in Chapter 5 [205, 219, 120, 196, 9, 101, 13, 225]. In the offline learning phase, SNNs are trained by SL to approximate a gain-scheduled proportional-integral-filter (PIF) compensator designed to stabilize the ideal FWMAV dynamic model. In the online learning phase, the SNN connection weights are incrementally updated by PGRL in the direction that minimizes the state deviation from desired set points. The distinct advantage of this PGRL-based online adaptation is that it relies on the state measurement rather than prior uncertainty detection or identification, which brings computational efficiency and allows the adaptive SNN controller to account for a wide variety of unexpected circumstances. The performance of this novel adaptive SNN control design is benchmarked by a non-adaptive SNN controller, which is simply a fixed offline SNN approximation of the gain-scheduled PIF compensator. In numerical simulations, both controllers are implemented to control an insect-scale flapping-wing robot known as RoboBee [118]. The adaptive SNN controller is shown to significantly outperform the non-adaptive one when the RoboBee is commanded to hover with wing damage, track a moving flower with wing asymmetry, follow a square trajectory with state measurement errors and con-

duct a coordinated turn with actuator failure. The bio-inspired sensing and control approaches presented in this dissertation can be potentially applied to develop the next generation of MAVs that are highly efficient and robust in the presence of wind disturbances and modeling uncertainties.

CHAPTER 2

FLOW-AIDED CONTROL OF AERIAL VEHICLES IN TURBULENT FLOW

2.1 Introduction

Animals such as soaring birds, migrating insects and swimming fish can traverse turbulent flows efficiently by taking advantage of approximately stationary flow structures [4, 143, 145, 41, 138, 180]. Birds like eagles and storks with large wing span and surface area are able to detect and exploit rising thermals or shear flows to generate lift and therefore save energy for long-distance flight [75, 3, 67]. Migrating insects can adaptively change their headings to harvest energy from atmospheric structures and motions based on their real-time measurements from wind-sensitive hairs and antennas [44, 129, 46, 27]. Fish are found to be capable of detecting their surrounding flow features using the lateral line flow sensory system, and learn to adjust their swimming speed and body undulation while traversing turbulent water currents [94, 22, 17, 181]. Many of these energy-harvesting features discovered in animal flyers and swimmers have also been observed in the characteristic motions of particles and bubbles carried by turbulent flows [126, 132, 108, 18], which have inspired the new flow-aided air-vehicle feedback control design presented in this chapter.

There is significant precedent for tackling air-vehicle navigation and control problems in strong but constant winds [127, 202, 197] and thermals [158]. Despite the prevalence of turbulence, its impact on locomotion, and the potential inherent in its energetic yet organized internal structure [100, 108], most existing approaches either treat wind effects as disturbances to be rejected or require global knowledge of the entire wind velocity field [160, 8]. This global knowl-

edge may be acquired through learning [216, 157, 15, 76] or with environmental prediction, modeling and forecasting [12]. For instance, in [158] global knowledge is acquired by simulating turbulent thermals similar to those arising in the atmospheric boundary layer, and by using model-free reinforcement-learning algorithms to train gliders to soar. Besides requiring prior training, this approach generates more conservative policies than those observed in piloted gliders, and requires gathering information about the fluctuating flow while simultaneously ascending in it. Another approach is to exploit globally known flow structures produced by environmental prediction and forecasting algorithms to generate optimal vehicle trajectories using methods such as mathematical programming, differential evolution, or Lagrangian coherent structures (LCSs) [12, 54, 229]. While this approach is useful for underwater vehicles because ocean currents may be predicted to some extent using oceanographic modeling and prediction tools [162, 164, 163, 83, 178, 21], it is less suited to air vehicles that must navigate rapidly changing winds without knowledge of global turbulent structures [105, 7, 68].

The process of particle transport in turbulence demonstrates that under certain conditions inertial particulates and droplets move quickly through turbulent flows such as turbulent air, water, or flames, without global knowledge of the velocity field [98, 93, 104, 18]. The *fast-tracking effect* is the phenomenon by which inertial particles in turbulent flows exhibit an average settling velocity that is larger in turbulence than in still air [126, 5]. Fast tracking of particles and droplets has been observed and verified in both physical experiments [5, 73], and direct numerical simulations (DNS) of the gravitational settling of inertial particles in complex flow fields, including cellular flow fields [126], Gaussian random flow fields [124], and homogeneous isotropic turbulence [204, 73]. To-

ward exploitation of this phenomenon, [18] analyzes theoretically the energetics of idealized fast-tracking flight vehicles that make only local, instantaneous measurements, revealing an extended parameter regime in which turbulence can decrease flight time or energy consumption in principle.

This chapter presents a new feedback control approach inspired by turbulent particle transport theory [18] that is able to reproduce fast-tracking in air vehicles traversing turbulent flow fields. By viewing the particle dynamics as the ideal response to the surrounding flow, implicit-model following (IMF) can be used to design a fast-tracking control (FTC) system that, by virtue of the onboard propulsion and actuation, induces the vehicle to behave like a particle in the closed loop. As a result, the vehicle flies within advantageous tail winds more often than with existing control methods. The vehicle also avoids adverse headwinds automatically, thereby reducing the energy and time required to traverse a turbulent flow, and it does so without access to global flow information. The energy-harvesting potential of the new FTC control approach is demonstrated through two benchmark control problems known as the minimum-energy and minimum-time problems. The FTC-controlled vehicle performance is compared to two optimal control solutions obtained using linear-quadratic regulator (LQR) and bang-bang control (BBC) theory. The LQR solution to the minimum-energy problem is derived by using information about the flow field to make the vehicle reach and maintain a desired steady-state velocity using minimum control effort. The BBC solution to the minimum-time problem is derived by making the vehicle reach the final desired position in minimum time in still fluid.

Although the FTC approach only requires instantaneous knowledge of vehi-

cle state and local flow, which are easily obtained onboard, it significantly outperforms both LQR and BBC designs. This general approach to flow-aided feedback control can also be applied to other vehicles including fixed- or flapping-wing aircraft [135, 177], rotorcraft, and neutrally-buoyant vehicles such as submarines or balloons, and to non-stationary flow structures such as thermal updrafts or mean shear [4, 201]. The primary advantage of the fast-tracking approach over existing methods is that unsteady turbulent structures can be leveraged without relying on any prediction of the velocity field. In fact, by drawing inspiration from nature, the FTC design only requires approximate knowledge of a few key flow parameters, such as the mean velocity, the typical vortex length scale, and the typical vortex timescale, which can be easily estimated onboard, as shown in [224]. Extensive numerical simulations show that an air vehicle using FTC follows the ideal response of the fast-tracking particle with zero tracking error, regardless of its true inertial characteristics. As a result, the FTC system increases the average horizontal velocity of the vehicle, maintains the desired steady-state velocity with less control effort than the LQR solution, and reaches a desired horizontal position before the BBC solution.

This chapter is organized as follows. Section 2.2 reviews relevant background from transport theory on the fast-tracking effect and cellular flows used here for illustrative purposes. The fast-tracking feedback-control design problem is formulated in Section 2.3, along with its basic assumptions. The FTC control design solution derived using implicit model following is presented in Section 2.4. In Section 2.5, the FTC energy-harvesting ability is demonstrated by comparing its performance to that of the optimal LQR solution on a benchmark minimum-energy control problem. In Section 2.6, the FTC time-saving ability is demonstrated by comparing its performance to that of the optimal BBC solution

on a benchmark minimum-time control problem. Finally, the FTC performance robustness with respect to the vehicle inertial characteristics and turbulent flow parameters is demonstrated through dozens of representative case studies, in Sections 2.5-2.6.

2.2 Background on Transport Theory and the Fast-Tracking Effect

Natural phenomena such as soaring birds exploiting thermal convection [143] and particles or bubbles in steady vortices [132] demonstrate that turbulent air flows can be traversed rapidly and efficiently by leveraging local knowledge of approximately stationary flow structures. In particular, the mechanism known as "fast-tracking effect" has been shown to govern the fast and intelligent motion of inertial particles in homogeneous turbulence [143, 73, 200, 124]. This chapter presents an approach for using the fast-tracking particle dynamic model, known from transport theory, in order to develop high-performance feedback control laws that can be implemented on autonomous vehicles in turbulent flow using local wind measurements and classic state estimation algorithms. The approach is demonstrated for a cellular-flow homogeneous-turbulence model, described in Section 2.2.2, which has been shown effective at capturing vortical structures in natural flows relevant to autonomous vehicles, such as atmospheric circulation [192] and ocean currents [107, 125].

2.2.1 Fast-Tracking Effect

Experimental studies have shown that fast-tracking particles in a turbulent flow field are preferably thrown out of vortices, toward their downward-sweeping sides (Fig. 2.1a), through a mechanism that increases the average speed of particles toward the bottom of the flow [73]. More precisely, the fast-tracking effect causes the mean settling velocity of an inertial particle traversing a turbulent flow to be increased with respect to the still-fluid settling velocity [73]. The particle's mean settling velocity, denoted by v, is the average falling speed of particles subject to drag and gravitational forces when reaching an average force balance in a turbulent flow, or,

$$v = \frac{1}{t_f - t_s} \int_{t_s}^{t_f} v_p(t)\, dt, \tag{2.1}$$

where v_p is the particle's instantaneous velocity, t_s is the settling time required for the velocity to reach and remain within a given error band, and t_f is the terminal time [159]. The still-fluid settling velocity, denoted by v_g, is the terminal falling speed of a particle through still fluid. Therefore, when fast tracking prevails, it can be observed that $v > v_g$.

As shown in [73], fast-tracking particles are characterized by physical characteristics, such as response time and settling velocity, that "resonate" with those of the turbulent flow, as reviewed in the remainder of this subsection. Let m and D denote the mass and diameter of the particle, respectively. Then, when the particle is surrounded by a fluid with dynamic viscosity μ, its motion is characterized by the *inertial response time*,

$$\tau \triangleq \frac{m}{3\pi D\mu} \tag{2.2}$$

as shown in [123]. In particular, the inertial response time represents the time re-

quired to reach equilibrium in response to perturbations in surrounding flows. The nature of the interaction between the particle and the flow depends on τ as well as on the characteristics of the turbulent flow, namely, the root-mean-square fluid velocity u', the vortex length scale l, and the vortex time scale $\tau_w \equiv l/u'$ [73]. The root-mean-square velocity, u', is defined as the standard deviation of the instantaneous flow velocity, u, such that [146]

$$u' = \sqrt{\frac{1}{T}\int_0^T [u(t) - \bar{u}]^2 \, dt}, \tag{2.3}$$

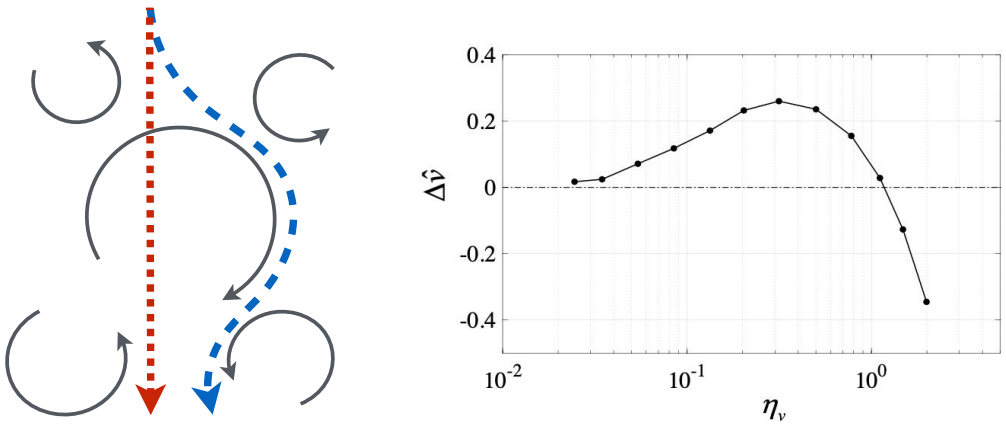
where $\bar{u}$ is the mean flow velocity over the time interval T.

When the particle's inertial response time (τ) and the still-fluid settling velocity (v_g) approach the turbulent flow's vortex time scale (τ_w) and the root-mean-square velocity (u'), respectively, the particle's settling velocity is significantly increased compared to its still-fluid settling velocity. Recently, this fast-tracking effect has been demonstrated experimentally by co-author Bewley using water droplets settling in air turbulence, as shown by the data plotted in Fig. 2.1b and taken from [73]. In particular, this study showed that the (normalized) increase in settling velocity,

$$\triangle\hat{v} \triangleq (v - v_g)/u' \tag{2.4}$$

is positive whenever the particle undergoes the fast-tracking effect.

Consider the dimensionless particle-settling parameter, $\eta_v \triangleq v_g/u'$, which governs the onset of fast tracking observed when η_v is of order one. When $\eta_v << 1$ or $\eta_v >> 1$, the particle's mean turbulent settling velocity (v) is not enhanced compared to v_g. In fact, a sharp decline of the settling velocity is observed for large particle settling parameter, η_v, due to the development of nonlinearity in drag forces on quickly settling particles. Importantly, the normalized



(a) (b)

Figure 2.1: (a) Particles are swept and, thus, readily accelerated into the downward sweeping sides of eddies (blue trajectory) rather than falling straight through turbulence (red trajectory). As a result, (b) water droplets in turbulent air experience increased settling velocity ($\triangle\hat{v}$) when the particle settling parameter (η_v) is of order one (experimental data taken from [73]).

increase in settling velocity, $\triangle\hat{v}$, is *maximum* when η_v is of order one and, therefore, this parameter value can be used as guiding principle in the development of a feedback controller that leverages turbulent flow to accelerate the vehicle similarly to the fast-tracking particle in Fig. 2.1a.

This chapter develops a new control approach by viewing the autonomous air vehicle in a turbulent flow as an inertial point-mass particle driven by constant horizontal thrust that can be adjusted so as to match the desired fast-tracking characteristics of the given turbulent flow. There is considerable precedent for treating vehicles as point masses for navigation and control purposes, whenever their size is small relative to the vortex length scale [68, 59, 141, 207, 32, 79, 148, 147]. Hence, our hypothesis is that by producing a controlled thrust that modifies the vehicle's inertial response time to match the vortex time scale, the vehicle may be accelerated through the turbulent flow similarly to the fast-tracking effect (Fig. 2.1a). Under these conditions, we ex-

pect the vehicle to be preferentially swept toward the sides of vortices pushing in the direction of motion, and to be accelerated along a fast-tracking trajectory, as shown by the simulated comparison in the next section. By taking advantage of beneficial flow structures, the vehicle may achieve a larger average terminal horizontal velocity and travel a longer distance over the same amount of time when compared to other (inefficient) trajectories.

2.2.2 Cellular Flow Fields

For illustration purposes, the control approach presented in this chapter is demonstrated for vehicles traversing a two-dimensional cellular flow field with known characteristic parameters. However, the approach can be extended to other flow structures for which fast-tracking results are also available [124, 204, 73]. Cellular flow is an idealized model of homogeneous turbulent flow that contains a periodic array of eddies described by the vortex length scale [126]. As shown in Fig. 2.2, vortices located in adjacent cells swirl in opposite directions. The cellular flow field is chosen here because it captures essential features of fast-tracking phenomena observed in fully turbulent flows, with some important exceptions described in [124, 73, 18]. Furthermore, cellular flow represents the best-case scenario for a vehicle in turbulence in the sense that there exist paths for which the flow always provides a tailwind and never a headwind. Finally, by demonstrating the novel fast-tracking control approach in cellular flow, the results may be applicable to a broad range of natural flow phenomena, including but not limited to Langmuir cells in water bodies [107, 125] and convective cellular motions in clouds [192].

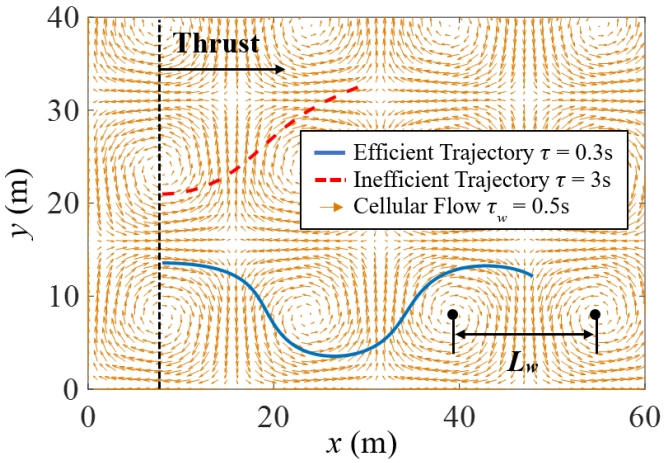


Figure 2.2: The trajectories of two particles with different inertial response times (τ) are compared by allowing them to travel for the same amount of time through a cellular flow field with vortex time scale (τ_w) after they are both released at the dashed black line (see [220] for animation).

Given a characteristic flow velocity U_0, the horizontal and vertical components of the two-dimensional cellular flow velocity (yellow vectors in Fig. 2.2) can be modeled as,

$$w_x = U_0 \sin(\frac{\pi x}{L_w}) \cos(\frac{\pi y}{L_w}) \tag{2.5a}$$

$$w_y = -U_0 \cos(\frac{\pi x}{L_w}) \sin(\frac{\pi y}{L_w}) \tag{2.5b}$$

respectively, where x and y are the coordinates in the plane, and L_w is a characteristic parameter that represents the distance between two adjacent vortices and is known as vortex length scale. Together with the U_0, the vortex length scale, L_w, determines the vortex time scale,

$$\tau_w \equiv \frac{L_w}{U_0}. \tag{2.6}$$

which represents the vortex turnover time.

According to the fast-tracking phenomenon, a particle traversing a cellular flow field makes use of the flow structure to travel faster when its inertial response time, τ, is approximately equal to the vortex time scale defined in (2.6).

As illustrated by the simulated blue trajectory in Fig. 2.2, a fast-tracking particle reaches a higher mean settling velocity. Hence, when compared to particles characterized by very different mass and diameter (e.g., red dashed line in Fig. 2.2), a fast-tracking particle travels a much greater (horizontal) distance over the same period of time.

Inspired by these natural phenomenon, this chapter develops a feedback control approach devised to allow air vehicles to make use of the eddies to traverse the cellular flow efficiently. By using an implicit model following approach, knowledge of the vortex time scale is used to develop a feedback control law that leverages the fast-tracking effect, irrespective of the vehicle's mass and size. In the proposed approach, the gravitational force acting on the particle is replaced by a controllable horizontal thrust force (Fig. 2.2) acting on the vehicle by virtue of an onboard propulsion mechanism, such as a propeller or jet engine.

2.3 Problem Formulation and Assumptions

Although the problems of guidance and control in turbulence have been investigated extensively to date [211, 92, 1, 61, 151, 152, 193], previous approaches have focused on attenuating the influence of external wind forces and moments by methods known as disturbance rejection. Besides being applicable only for small disturbances with known and well-posed statistics, such as zero mean and Gaussian characteristics, previous approaches sought to eliminate wind effects, rather than to exploit them as do natural flyers [143, 132, 4]. As in the extensive literature on trajectory planning for fixed-wing aircraft [68, 32, 79]

and quadcopters [148, 147], let the air vehicle be approximated by a point mass and denote its mass by m_v and its diameter by D_v. The point-mass assumption is effective in practice when the vehicle geometry can be ignored in obstacle avoidance problems, and the vehicle size is much smaller than the vortex length scale, or $D_v << L_w$. Typically, the size of small unmanned aerial vehicles (UAVs) spans from around 15 centimeters to 2 meters, and the length scale of the energetic turbulent eddies in the atmosphere is about 100 meters [50, 150, 146]. Furthermore, as illustrated in Fig. 2.2, the vehicle propelled by a thrust force must traverse a cellular-flow wind field with vortex length scale, L_w, and time scale, τ_w. For simplicity, the lift force is assumed appropriate for maintaining the vehicle aloft or, alternatively, the vehicle may be assumed neutrally buoyant [18]. Also, it is assumed by the same rationale that the effects of the vehicle on the surrounding flow are negligible.

Because the vehicle may encounter different flow fields during its operations and its physical characteristics (mass and diameter) are fixed *a priori*, its inertial response time, τ_v, may not always be approximately equal to τ_w. Therefore, in general, the vehicle may not experience the fast-tracking effect. The problem considered in this chapter is to develop a feedback control law that modifies the vehicle's inertial response time in the closed loop, so as to achieve fast tracking by virtue of the controllable thrust forces. It is assumed that the fluid flow dynamic viscosity, μ, and time scale, τ_w, are either known *a priori* or estimated from wind measurements online, for example, using the sparse identification of nonlinear dynamics (SINDy) [224, 20]. The vehicle physical parameters are lumped into a constant vector, $\boldsymbol{\theta} = [m_v \;\; D_v]^T$, and the onboard propulsion produces a constant horizontal thrust, $T_x = m_v a_x$, as well as acceleration-based control inputs, $\mathbf{u} = [u_x \; u_y]^T$. The vehicle acceleration produced by the constant horizontal

thrust is denoted by $\mathbf{a} = [a_x \ 0]^T$. Then, from transport theory [124], the two-dimensional vehicle dynamics subject to a cellular flow can be modeled by a linear parameter-dependent system,

$$\dot{\mathbf{x}}(t) = \mathbf{A}(\boldsymbol{\theta},\mu)\mathbf{x}(t) + \mathbf{B}(\boldsymbol{\theta},\mu)\mathbf{u}(t) + \mathbf{L}(\boldsymbol{\theta},\mu)\mathbf{w}(t) + \mathbf{a}, \quad \mathbf{x}(t_0) = \mathbf{x}_0, \tag{2.7}$$

where the state vector, $\mathbf{x} = [v_x \ v_y]^T$, consists of the x- and y-components of the vehicle velocity in inertial frame, and the wind flow velocity vector, denoted by $\mathbf{w} = [w_x \ w_y]^T$, is assumed known from onboard measurements. The initial conditions, $\mathbf{x}_0$, are known from the vehicle.

The state-space matrices are given by,

$$\mathbf{A}(\boldsymbol{\theta},\mu) = \begin{bmatrix} -\frac{3\pi D_v \mu}{m_v} & 0 \\ 0 & -\frac{3\pi D_v \mu}{m_v} \end{bmatrix}; \ \mathbf{B}(\boldsymbol{\theta},\mu) = \mathbf{I}_{2\times 2}; \ \mathbf{L}(\boldsymbol{\theta},\mu) = \begin{bmatrix} \frac{3\pi D_v \mu}{m_v} & 0 \\ 0 & \frac{3\pi D_v \mu}{m_v} \end{bmatrix} \tag{2.8}$$

where it is assumed that the vehicle is subject to a linear Stokes drag force [191]. This assumption is justified when the flow is incompressible and $D_v << L_w$, and holds approximately for air vehicles such as fixed-wing aircraft and rotorcraft in high Reynolds number regimes under certain conditions [179, 108, 91, 171]. A state-feedback controller is developed, assuming that the vehicle state is fully observable and estimated with zero error for simplicity [57]. Furthermore, a constant horizontal thrust is provided to obey (2.2), such that

$$T_x = m_v a_x = 3\pi\mu D_v \tau_v a_x \tag{2.9}$$

This chapter seeks to develop a feedback control system inspired by the natural transport phenomena described in Section 2.2.1, such that, in the closed loop, the vehicle behaves like a particle undergoing the fast-tracking effect. The desired automatic feedback control law must provide the vehicle inputs $\mathbf{u}(t)$, in

(2.7), continuously over time, so as to exploit the energy and organized structure of the eddies in the cellular flow field. This novel control approach is demonstrated by solving the following benchmark control problems:

1) *Minimum energy problem*: determine control inputs, $\mathbf{u}(t)$, so as to reach and maintain a desired steady-state velocity through the cellular flow with minimum control effort.

2) *Minimum time problem*: determine control inputs, $\mathbf{u}(t)$, so as to travel a desired distance in the horizontal direction through the cellular flow in minimum time.

The new FTC control approach is derived using implicit model following in the next section. Subsequently, its performance is demonstrated in Section 2.5 and 2.6, and compared to two classic optimal solutions obtained via linear quadratic regulation and bang-bang control, respectively. Although the approach is demonstrated on the simplified air vehicle model in (2.7), the methods proposed in this chapter can be easily extended to more detailed vehicle dynamic models, provided they too may be approximated by linear parameter dependent systems.

2.4 FTC Control Design via Implicit Model Following (IMF)

The FTC feedback control design is developed by specifying an implicit model based on the ideal response of an efficient fast-tracking particle in the loop. While most of the existing control methods seek to compensate for, or reject, wind effects, the FTC control approach seeks to make use of organized cellu-

lar flow structures in order to benefit from them in terms of speed and energy consumption. Because the real geometry and location of the eddies is unknown to the vehicle, it may not be utilized for trajectory optimization. Rather, in an effort to mimic natural transport phenomena, the dynamic model of an ideal fast-tracking particle is first obtained from the known parameters of the cellular flow (L_w and τ_w), as described in Section 2.2.2. Subsequently, a state-feedback control law is obtained using implicit model following (IMF) [188], such that the closed-loop vehicle dynamics may follow the ideal fast-tracking particle model as closely as possible.

The IMF approach, originally proposed in [188], leverages an implicit dynamic model to obtain a control system that conforms to an ideal behavior. Once an ideal dynamic model with state, $\mathbf{x}_m \in \mathbb{R}^n$, is formulated, the IMF control law is obtained by minimizing the error between the time derivatives of the vehicle state, $\mathbf{x} \in \mathbb{R}^n$, and those of the model, i.e.:

$$J = \frac{1}{2} \int_{t_0}^{t_f} [\dot{\mathbf{x}}(t) - \dot{\mathbf{x}}_m(t)]^T \mathbf{Q}_m [\dot{\mathbf{x}}(t) - \dot{\mathbf{x}}_m(t)] \, dt, \tag{2.10}$$

where the subscript m refers to "model". The positive definite weighting matrix $\mathbf{Q}_m \in \mathbb{R}^{n \times n}$ can be utilized to specify a desired trade-off between state variables, for example, in order to account for states' range and units. The IMF equations and control law are derived in Section 2.4.2, based on the fast-tracking particle model presented in the next section.

2.4.1 Ideal Fast-Tracking Particle Model

As a first step, let us reinterpret the gravitational force acting on an inertial particle as a constant thrust in the horizontal direction, T_{x_m}. With a simple coor-

dinate transformation, assuming the air flow is incompressible and the particle diameter, D_m, is significantly smaller than the vortex length scale, L_w, the two-dimensional governing equation for a spherical inertial particle of mass m subject to a drag force $\mathbf{F}_d$ and traveling in a cellular flow field is

$$m\dot{\mathbf{x}}_m = \mathbf{F}_d + \mathbf{T}_m \tag{2.11}$$

where $\mathbf{x}_m = [v_{x_m} \; v_{y_m}]^T$ contains the particle velocity in inertial frame, and $\mathbf{T}_m = [T_{x_m} \; 0]^T$ denotes constant external thrust. In this ideal model, the inertial particle is subject to a linear Stokes drag force,

$$\mathbf{F}_d = -3\pi D_m \mu(\mathbf{x}_m - \mathbf{w}_m) \tag{2.12}$$

where $\mathbf{w}_m = [w_{x_m} \; w_{y_m}]^T$ is the flow velocity in inertial frame [191, 16]. As explained in Section 2.3, the above assumption holds, once again, because it can be assumed that the flow is incompressible and $D_m << L_w$ [179, 108, 91, 171]. In spite of all these assumptions, the ideal particle model presented in this section can well explain many natural particle transport phenomena, such as water droplets settling in air turbulence [149] and soot formation in turbulent flames [209].

When the inertial response time of the ideal particle, τ_m, is approximately equal to τ_w, the particle exhibits the fast-tracking effect and naturally follows the most efficient trajectories inside the cellular flow. From the equations of the particle's inertial response time (2.2) and the particle model (2.11), the particle dynamics can be expressed as a linear parameter-dependent system,

$$\dot{\mathbf{x}}_m(t) = \mathbf{A}_m(\tau_m)\mathbf{x}_m(t) + \mathbf{L}_m(\tau_m)\mathbf{w}_m(t) + \mathbf{a}_m, \quad \mathbf{x}_m(t_0) = \mathbf{x}_{0_m}, \tag{2.13}$$

where $\mathbf{a}_m = \frac{1}{m}\mathbf{T}_m = [a_{x_m} \; 0]^T$ is the particle acceleration produced by the constant horizontal thrust, $\mathbf{x}_{0_m}$ are the particle's initial conditions, and model state-space

matrices, $\mathbf{A}_m$ and $\mathbf{L}_m$, depend only on the ideal particle's inertial response time, τ_m, which is chosen to match the vortex time scale. The state-space matrices are given by

$$\mathbf{A}_m(\tau_m) = \begin{bmatrix} -\frac{1}{\tau_m} & 0 \\ 0 & -\frac{1}{\tau_m} \end{bmatrix}; \ \mathbf{L}_m(\tau_m) = \begin{bmatrix} \frac{1}{\tau_m} & 0 \\ 0 & \frac{1}{\tau_m} \end{bmatrix} \tag{2.14}$$

2.4.2 Fast-Tracking Controller (FTC) Design

Unlike the ideal particle described in the previous section, in general, the vehicle has an inertial response time that is *not* approximately equal to the vortex time scale. Therefore, a feedback control law can be derived to change the vehicle response in the closed loop and make it follow the behavior of the ideal particle model, that is implicit in the law itself. Choose $\mathbf{w} = \mathbf{w}_m$, and construct a quadratic cost function in the form (2.10),

$$\begin{aligned} J &= \frac{1}{2}\int_{t_0}^{t_f} [\dot{\mathbf{x}}(t) - \dot{\mathbf{x}}_m(t)]^T \mathbf{Q}_m [\dot{\mathbf{x}}(t) - \dot{\mathbf{x}}_m(t)]\, dt \\ &= \frac{1}{2}\int_{t_0}^{t_f} [\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + 2\mathbf{x}^T(t)\mathbf{M}\tilde{\mathbf{u}}(t) + \tilde{\mathbf{u}}^T(t)\mathbf{R}\tilde{\mathbf{u}}(t)]\, dt, \end{aligned} \tag{2.15}$$

where $\tilde{\mathbf{u}} = \mathbf{u} + (\frac{1}{\tau_v} - \frac{1}{\tau_m})\mathbf{w} + \mathbf{a} - \mathbf{a}_m$, $\mathbf{Q}_m = \mathbf{I}_{2\times 2}$, and the weighing matrices are designed as follows,

$$\begin{aligned} \mathbf{Q} &= (\mathbf{A} - \mathbf{A}_m)^T \mathbf{Q}_m (\mathbf{A} - \mathbf{A}_m) \\ \mathbf{M} &= (\mathbf{A} - \mathbf{A}_m)^T \mathbf{Q}_m \mathbf{B} \\ \mathbf{R} &= \mathbf{B}^T \mathbf{Q}_m \mathbf{B} \end{aligned} \tag{2.16}$$

in order to minimize (2.15).

For the vehicle model shown in Section 2.3, perfect model following can be achieved (with zero state error) because the following perfect-model-following

criterion is satisfied,

$$(\mathbf{B}\mathbf{B}^L - \mathbf{I}_n)(\mathbf{A} - \mathbf{A}_m) = \mathbf{0}, \tag{2.17}$$

where $\mathbf{B}^L = (\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T$ is the left pseudo-inverse [188]. When the FTC approach is extended to other vehicle dynamics, the above criterion may not be always satisfied and, hence, the optimal IMF control law described in [53] can be adopted to minimize the model-following error. In particular, letting t_f approach infinity in (2.15), the optimal control law can be obtained in terms of a steady-state gain matrix $\mathbf{C}(0)$ such that,

$$\tilde{\mathbf{u}}(t) = -\mathbf{C}(0)\mathbf{x}(t) = -\mathbf{R}^{-1}[\mathbf{B}^T\mathbf{S}(0) + \mathbf{M}^T]\mathbf{x}(t) \tag{2.18}$$

where $\mathbf{S}(0)$ is the solution of the algebraic Riccati equation (ARE),

$$[\mathbf{S}(0)\mathbf{B} + \mathbf{M}]\mathbf{R}^{-1}[\mathbf{B}^T\mathbf{S}(0) + \mathbf{M}^T] - \mathbf{A}^T\mathbf{S}(0) - \mathbf{S}(0)\mathbf{A} - \mathbf{Q} = \mathbf{0} \tag{2.19}$$

In this case, the Riccati matrix solution that guarantees closed-loop asymptotic stability is $\mathbf{S}(0) = \mathbf{0}_{2\times 2}$, and the corresponding steady-state gain matrix is

$$\mathbf{C}(0) = \mathbf{R}^{-1}[\mathbf{B}^T\mathbf{S}(0) + \mathbf{M}^T] = \left(\frac{1}{\tau_m} - \frac{1}{\tau_v}\right)\mathbf{I}_{2\times 2} \tag{2.20}$$

Hence, under the aforementioned assumptions, the FTC IMF feedback control law is

$$\tilde{\mathbf{u}}(t) = -\mathbf{C}(0)\mathbf{x}(t) = -\left(\frac{1}{\tau_m} - \frac{1}{\tau_v}\right)\mathbf{x}(t) \tag{2.21}$$

which is implemented as the following acceleration-based control inputs onboard the air vehicle,

$$\mathbf{u}(t) = \tilde{\mathbf{u}}(t) - \left(\frac{1}{\tau_v} - \frac{1}{\tau_m}\right)\mathbf{w}(t) - \mathbf{a} + \mathbf{a}_m = \left(\frac{1}{\tau_v} - \frac{1}{\tau_m}\right)[\mathbf{x}(t) - \mathbf{w}(t)] - \mathbf{a} + \mathbf{a}_m \tag{2.22}$$

It can be seen that the FTC control law requires only online measurements of the vehicle state, $\mathbf{x}$, and of the local wind flow field, $\mathbf{w}$, both of which may

be estimated with excellent accuracy onboard many vehicles. As a result, without knowledge of the global wind profile (or of the precise eddies' positions and geometries), the FTC-controlled vehicle finds the most efficient trajectories achievable by the vehicle based on its dynamic constraints (2.7).

2.5 Minimum-Energy Solutions and Results

The utilization of aerial vehicles, especially small ones, is often limited by the allowed on-board battery capacity and duration of flight, particularly during rapid and aggressive maneuvers in extreme windy conditions [18, 215, 64]. Therefore, energy consumption has become an essential performance metric for small UAV control design. In this benchmark minimum-energy control problem, a feedback control law is desired to make the vehicle reach and maintain a desired steady-state velocity, $\mathbf{x}_r = [v_{x_r} \;\; v_{y_r}]^T$, through the cellular flow with minimum control effort. For comparison, the desired steady-state velocity $\mathbf{x}_r$ is chosen to be the mean settling velocity of the ideal fast-tracking particle. The classic optimal solution to this problem obtained by linear quadratic regulation will be proposed in Section 2.5.1. Then, the performance of FTC will be demonstrated by comparing with the classic LQR in two different case studies. In practice, the vehicle's inertial response time, τ_v, is *not* approximately equal to that of the ideal fast-tracking particle, τ_m. Therefore, FTC and LQR are implemented and compared for the control of an air vehicle with τ_v greater than τ_m in Section 2.5.2, and an air vehicle with τ_v smaller than τ_m in Section 2.5.3, respectively. In the end, based on multiple numerical simulations, the dependence of the minimum-energy simulation results on the choice of τ_v with respect to τ_m is discussed.

2.5.1 Comparison with Linear Quadratic Regulator (LQR)

LQR provides an optimal solution to the minimum-energy problem, in which the control objective is to reach and maintain a desired steady state velocity. The flow disturbances perpendicular to the desired steady state velocity are compensated by state feedback control. We first rewrite the two-dimensional vehicle dynamics (2.7) in the standard state-space form,

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\bar{\mathbf{u}}(t), \tag{2.23}$$

where $\bar{\mathbf{u}} = \mathbf{u} + \mathbf{a} + \frac{1}{\tau_v}\mathbf{w}$. Whereas, in the previous section, the cost function (2.15) was an integral of the difference between the state derivatives of the air vehicle and the fast-tracking particle, we construct the cost function as a different integral here that penalizes both state excursions and control effort for this particular problem,

$$J = \frac{1}{2}\int_{t_0}^{t_f} [\mathbf{x}^T(t)\mathbf{Q}\mathbf{x}(t) + \bar{\mathbf{u}}^T(t)\mathbf{R}\bar{\mathbf{u}}(t)]\, dt \tag{2.24}$$

where $\mathbf{Q} = \mathbf{R} = \mathbf{I}_{2\times 2}$. With perfect knowledge of the vehicle state, the desired feedback control law, $\bar{\mathbf{u}}$, can be expressed in terms of the steady-state gain matrix $\mathbf{C}(0)$ as t_f approaches infinity such that,

$$\begin{aligned} \bar{\mathbf{u}}(t) &= -\mathbf{C}(0)\mathbf{x}(t) + \mathbf{K}_r\mathbf{x}_r \\ &= -\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(0)\mathbf{x}(t) + \mathbf{K}_r\mathbf{x}_r \end{aligned} \tag{2.25}$$

where $\mathbf{S}(0)$ is a solution to the algebraic Riccati equation (ARE),

$$\mathbf{A}^T\mathbf{S}(0) + \mathbf{S}(0)\mathbf{A} - \mathbf{S}(0)\mathbf{B}\mathbf{R}^{-1}\mathbf{B}^T\mathbf{S}(0) + \mathbf{Q} = \mathbf{0} \tag{2.26}$$

and $\mathbf{K}_r$ can be chosen to track the reference $\mathbf{x}_r$ with zero steady-state error [130],

$$\mathbf{K}_r = -\{[\mathbf{A} - \mathbf{B}\mathbf{C}(0)]^{-1}\mathbf{B}\}^{-1} \tag{2.27}$$

In this case, the Riccati matrix solution that guarantees closed-loop asymptotic stability is

$$\mathbf{S}(0) = -\left(-\frac{1}{\tau_v} + \sqrt{\frac{1}{\tau_v^2} + 1}\right) \mathbf{I}_{2\times 2} \tag{2.28}$$

Combining (2.25), (2.27) and (2.28), we can obtain the following acceleration-based LQR feedback control law,

$$\mathbf{u}(t) = \bar{\mathbf{u}}(t) - \mathbf{a} - \frac{1}{\tau_v}\mathbf{w}(t) = -\left(-\frac{1}{\tau_v} + \sqrt{\frac{1}{\tau_v^2} + 1}\right) \mathbf{x}(t) + \sqrt{\frac{1}{\tau_v^2} + 1}\, \mathbf{x}_r - \mathbf{a} - \frac{1}{\tau_v}\mathbf{w}(t) \tag{2.29}$$

2.5.2 FTC Minimum-Energy Case Study 1

In this case study, FTC and LQR are implemented and compared for the control of a thrust-driven vehicle with $\tau_v > \tau_m$. In simulation, the controlled vehicle with $\tau_v = 0.21$ s traverses the cellular flow with $\tau_w = 0.15$ s for 20 seconds. Additionally, another purely forward-thrust-driven vehicle with the same τ_v as the controlled vehicles and an ideal fast-tracking particle with $\tau_m = \tau_w$ are simulated for comparison. As previously explained in Section 2.3, to keep the control cost comparable among different control designs and case studies, the horizontal thrust in (2.9) remains unchanged when the vehicle's inertial response time changes in different simulations. Therefore, $\tau_v a_x$ stays the same at 1.5 m/s here. In this FTC minimum-energy simulation, the fast-tracking particle, the vehicle purely driven by forward thrust, and FTC-controlled and LQR-controlled vehicles all start from the origin, and their initial velocities are assumed to be zero. FTC aims to make the air vehicle follow the ideal response of the fast-tracking particle. The control objective is to make the air vehicle reach and maintain a desired steady-state velocity, $\mathbf{x}_r = [v_{x_r}\ v_{y_r}]^T$, where $v_{x_r} = 15.41$ m/s and $v_{y_r} = 0$

m/s, through the cellular flow with minimum control effort.

Trajectories of the fast-tracking particle with $\tau_m = 0.15$ s, and purely thrust-driven, FTC-controlled and LQR-controlled vehicles with $\tau_v = 0.21$ s are compared in Fig. 2.3. They all start from the origin at the same time, go through a transitional period in the cellular flow, and travel horizontally in the end. Nevertheless, the LQR-controlled vehicle travels horizontally without any oscillations, because its vertical velocity maintains zero all the time to meet the control objective. The trajectory of the FTC-controlled air vehicle identically overlaps that of the fast-tracking particle, which demonstrates that the FTC-controlled air vehicle can perfectly follow the fast-tracking particle with zero tracking error. They both go through a short transitional period at first, and are then swept and, thus, readily accelerated into the downward sweeping sides of eddies. However, the air vehicle purely driven by constant horizontal forward thrust deviates a little from the desired horizontal direction at first, and takes a longer transitional time to adapt to the cellular flow conditions.

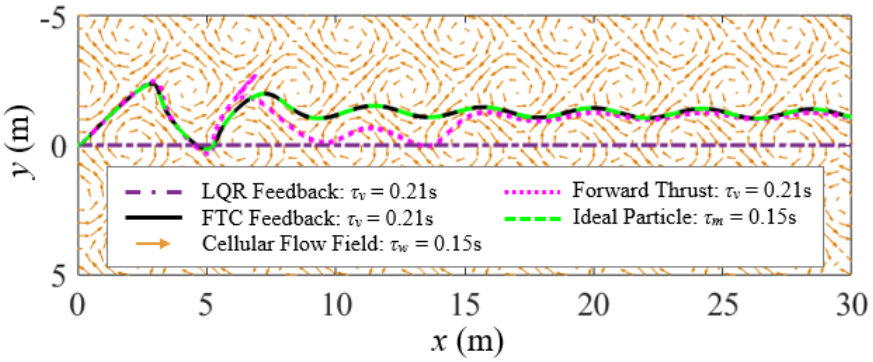


Figure 2.3: Case study 1: the trajectory comparison of the ideal fast-tracking particle with $\tau_m = 0.15$ s, and purely thrust-driven, FTC-controlled and LQR-controlled vehicles with $\tau_v = 0.21$ s traversing a cellular flow with $\tau_w = 0.15$ s demonstrates that the FTC-controlled vehicle can perfectly follow the fast-tracking particle with zero tracking error.

In Fig. 2.4, the velocity time histories of the fast-tracking particle, and

the purely thrust-driven, FTC-controlled and LQR-controlled vehicles are compared. The FTC-controlled and LQR-controlled air vehicles both meet the control objective of reaching and maintaining a desired steady-state velocity, $\mathbf{x}_r = [v_{x_r} \; v_{y_r}]^T$, where $v_{x_r} = 15.41$ m/s and $v_{y_r} = 0$ m/s. With the flow disturbance compensated, the LQR-controlled air vehicle achieves and maintains the desired steady-state velocity perfectly with zero tracking error, while the horizontal velocity component, v_x, of the FTC-controlled air vehicle oscillates a little around the desired horizontal steady-state velocity, v_{x_r}, due to the periodically changing flow conditions. At the expense of sacrificing the vehicle's riding comfort, the FTC-controlled vehicle exploits beneficial flow structures and harvests energy from cellular flows. Additionally, the LQR-controlled air vehicle reaches v_{x_r} faster than the FTC-controlled one. However, the vehicle purely driven by forward thrust takes a longer transitional time to adapt to the fluctuating flow conditions. According to Fig. 2.4, the average horizontal velocities of the LQR-controlled and FTC-controlled air vehicles over the entire period of simulation are both significantly greater than the vehicle purely driven by forward thrust. Consequently, both LQR-controlled and FTC-controlled vehicles travel a longer distance horizontally than the purely thrust-driven vehicle within the same period of time.

The objective of this benchmark minimum-energy problem is to use minimum control effort. Therefore, the quadratic control usage, given by $\mathbf{u}^T\mathbf{u} = u_x^2 + u_y^2$, of FTC is compared to that of LQR in Fig. 2.5. The total control effort of a controller is commonly quantified by the integral quadratic control usage C, which takes the form,

$$C = \int_{t_0}^{t_f} \mathbf{u}^T(t)\mathbf{u}(t)\, dt. \tag{2.30}$$

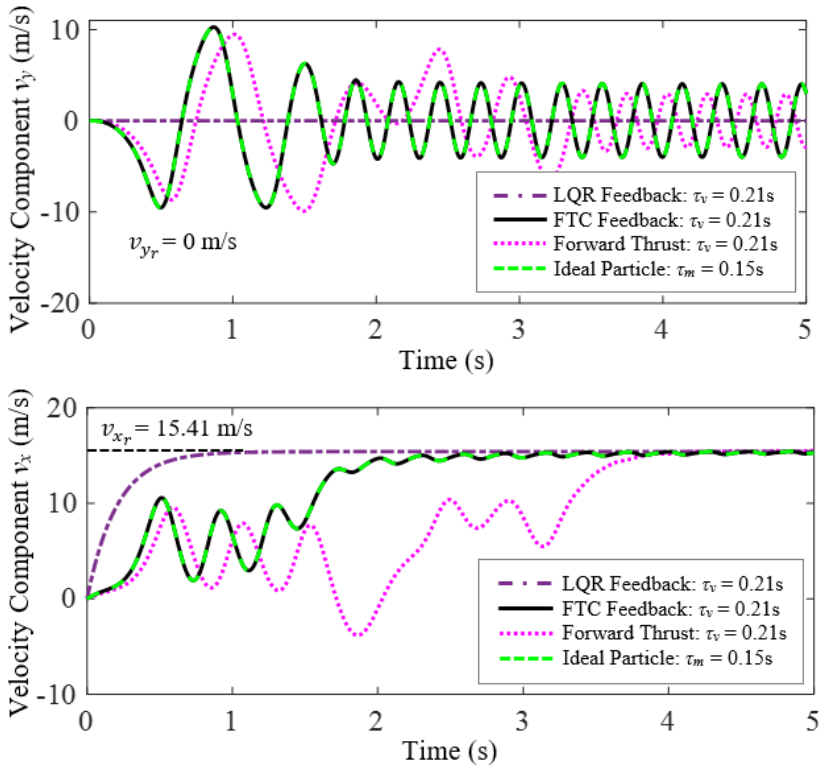


Figure 2.4: Case study 1: the comparison of the velocity time histories of FTC-controlled and LQR-controlled vehicles shows that they both achieve and maintain the desired steady-state velocities (v_{x_r} and v_{y_r}).

In this case study, over the same period from time $t_0 = 0$ s to time $t_f = 20$ s, the integral quadratic control usage of FTC is $C_f = 6.71 \times 10^3$ m^2/s^3, while that of LQR is $C_l = 1.33 \times 10^5$ m^2/s^3. LQR has much larger control cost than FTC.

To represent the difference of inertial response time between the controlled air vehicle and the ideal fast-tracking particle, the ratio of inertial response time is defined as

$$\eta_\tau = \frac{\tau_v}{\tau_m} \tag{2.31}$$

Similarly, to quantify the relative control savings of FTC compared to LQR, the ratio of total control cost is defined as

$$\eta_c = \frac{C_l}{C_f} \tag{2.32}$$

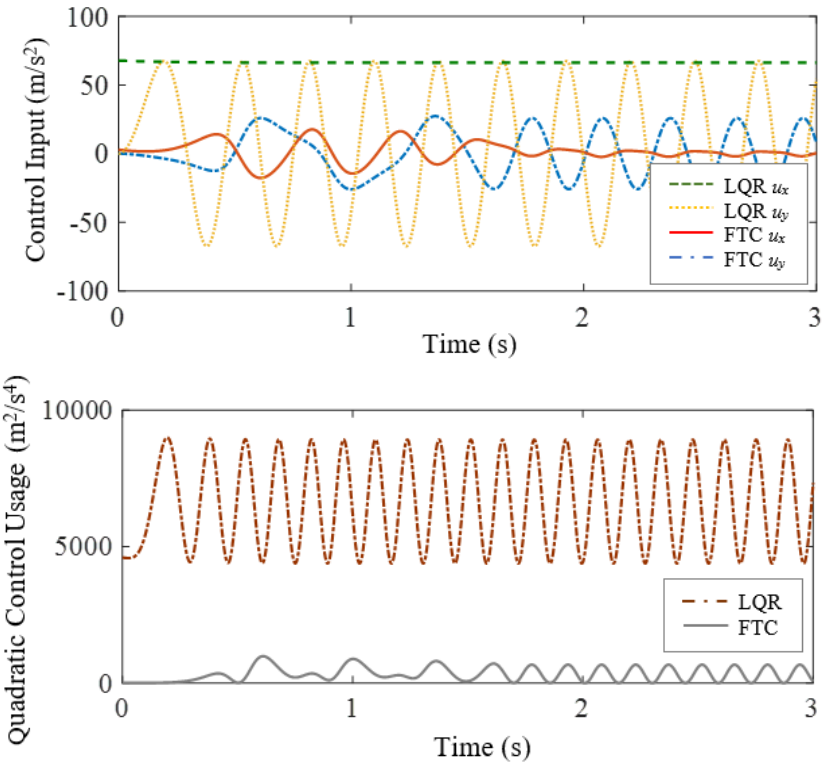


Figure 2.5: Case study 1: the comparison of the FTC and LQR control cost shows that the FTC-controlled vehicle meets the control objective with much less control effort than the LQR-controlled one.

Therefore, we can obtain that, in case study 1, the ratio of inertial response time is $\eta_\tau = 1.40$, and the ratio of total control cost is $\eta_c = 19.83$.

2.5.3 FTC Minimum-Energy Case Study 2

In this case study, FTC and LQR are implemented for the control of a thrust-driven vehicle with $\tau_v = 0.075$ s $< \tau_m$. The cellular flow parameters, control objective and simulation conditions are all the same as in case study 1. Another purely forward-thrust-driven vehicle with $\tau_v = 0.075$ s and an ideal particle are also simulated for comparison. To keep the control cost comparable among different cases, the horizontal thrust T_x, or $\tau_v a_x$ equivalently, remains the same

as in case study 1. Trajectories of the ideal particle and purely thrust-driven, FTC-controlled and LQR-controlled vehicles are compared in Fig. 2.6. Similar to case study 1, the FTC-controlled vehicle can perfectly follow the ideal particle with zero tracking error.

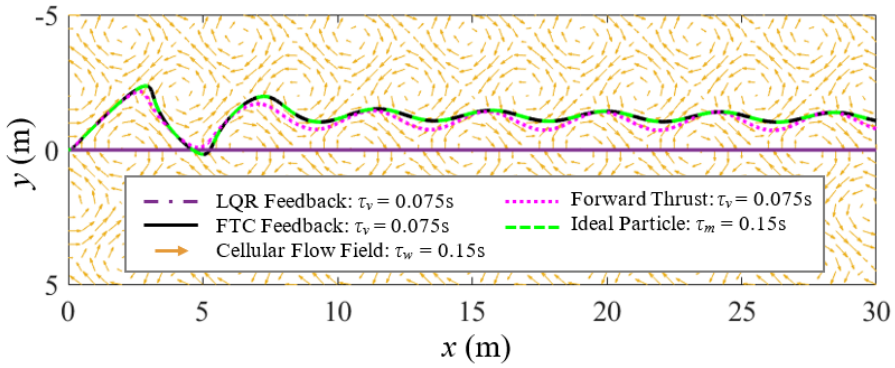


Figure 2.6: Case study 2: the trajectory comparison of the ideal fast-tracking particle with $\tau_m = 0.15$ s, and purely thrust-driven, FTC-controlled and LQR-controlled vehicles with $\tau_v = 0.075$ s traversing a cellular flow with $\tau_w = 0.15$ s.

In Fig. 2.7, the horizontal velocity and quadratic control usage of the two controlled vehicles are compared, which shows that both vehicles meet the velocity tracking control objective. Additionally, the LQR-controlled vehicle reaches v_{x_r} faster than the FTC-controlled one, and its settling time is less than half of the settling time in case study 1. The purely forward-thrust-driven vehicle behaves similarly in each case study with a much longer settling time and smaller average settling velocity compared to the controlled vehicles. Over the same period of simulation, the integral quadratic control usage of FTC is $C_f = 8.22 \times 10^4$ m^2/s^3, while that of LQR is $C_l = 1.04 \times 10^6$ m^2/s^3. Therefore, LQR has larger control cost than FTC. Accordingly, the ratio of inertial response time is $\eta_\tau = 0.5$, and the ratio of total control cost is $\eta_c = 12.69$ in this case.

Through multiple numerical simulations with the ratio of inertial response time η_τ ranging from 0.01 to 500, we find that there is a trade-off between the

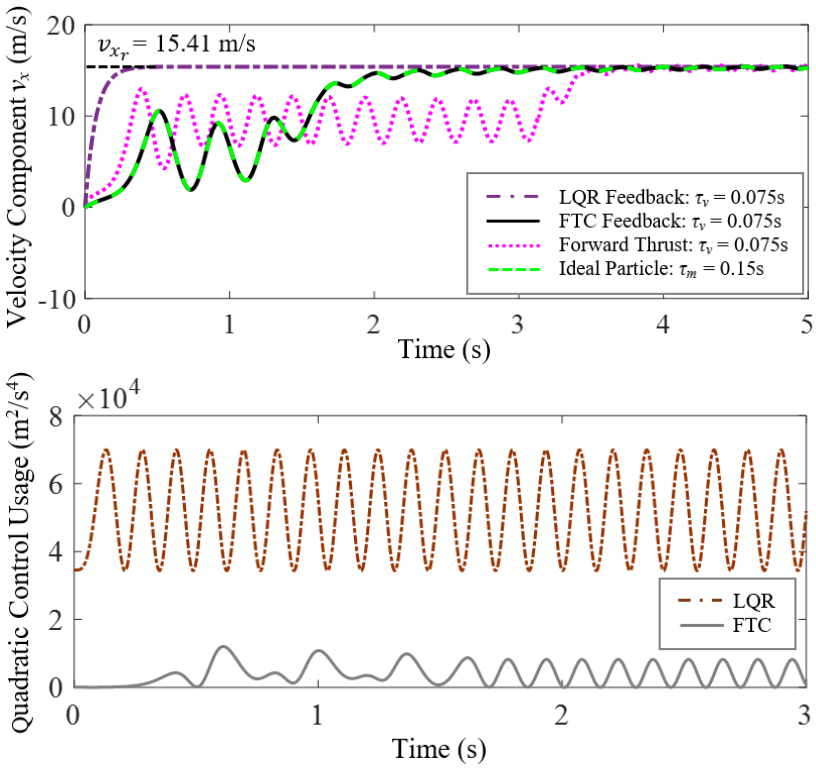


Figure 2.7: Case study 2: the comparison of the horizontal velocity of the ideal particle and purely thrust-driven, FTC-controlled and LQR-controlled vehicles and the comparison of the FTC and LQR control cost show that the two controlled vehicles both achieve and maintain the desired horizontal steady-state velocity, but the FTC-controlled vehicle costs much less control effort.

control savings of FTC and the difference of inertial response time between the controlled vehicle and the fast-tracking particle. The greater the difference in the inertial response time, the more control effort will be made by the FTC controller to follow the ideal response of the fast-tracking particle. In Fig. 2.8, the polynomial fits to the circular and cross data points illustrate how the relative total control cost, $\eta_c = C_l/C_f$, changes as a function of the normalized inertial response time of the vehicle, $\eta_\tau = \tau_v/\tau_m$. The relative total control cost, η_c, approaches infinity when $\eta_\tau = 1$, because the total control cost of FTC is zero when the inertial response time of the vehicle is equal to that of the fast-tracking particle. On the left side of the vertical asymptote at $\eta_\tau = 1$, all the red cross markers

correspond with simulations where $\tau_v < \tau_m$, and the yellow one represents case study 2. In this area, the relative total control cost, η_c, is enhanced as the normalized inertial response time, η_τ, grows. On the right side of the vertical asymptote at $\eta_\tau = 1$, all the blue circular markers correspond with simulations where $\tau_v > \tau_m$, and the yellow one represents case study 1. Conversely, in this area, η_c decreases as η_τ grows. Moreover, there exists two horizontal asymptotes for the polynomial fits: η_c approaches 1.50×10^{-3} as η_τ increases, and approaches 3.19 as η_τ decreases. In addition, the logarithms of the two ratios, η_c and η_τ, are approximately first-order linearly dependent with each other when η_τ is roughly within the range of 2 to 10. All the data points located in the grey shaded area above the horizontal dashed line at $\eta_c = 1$ correspond with simulations in which the FTC-controlled air vehicle achieves the control objective with less control effort than the LQR-controlled one, while those below the dashed line at $\eta_c = 1$ correspond with simulations where FTC costs more control effort than LQR. The critical point where $\eta_c = 1$ locates approximately at $\eta_\tau^* \approx 2.80$. Therefore, below this critical inertial response time ratio, the FTC-controlled vehicle uses less energy to traverse the cellular flow compared with the LQR-controlled one.

2.6 Minimum-Time Solutions and Results

As UAVs have been widely used, there is an increasing need to extend the range and endurance of UAV flight in many applications with extremely strict time limitations, including autonomous medical delivery and emergency response [23, 173]. In addition to the energy consumption, the cost of time has become another essential factor to consider for UAV path planning and control. In this benchmark minimum-time control problem, we aim to find an optimal con-

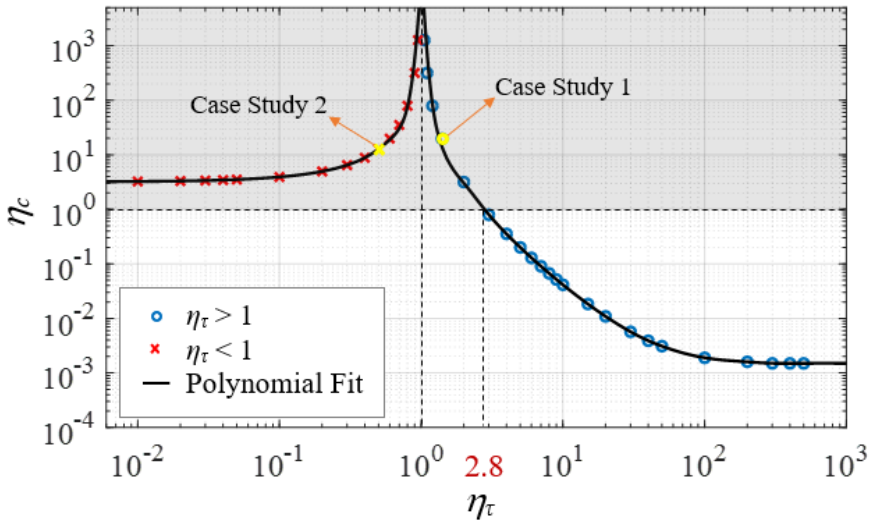


Figure 2.8: Log-log plot showing the relative total control cost (η_c) as a function of the normalized inertial response time of the vehicle (η_τ) for $\tau_m = \tau_w$. Circular and cross markers represent simulations corresponding with different η_τ, and the black lines are polynomial fits to these data points. Yellow markers represent the two simulations chosen as case studies in this section. Data points located in the grey shaded area correspond with simulations in which FTC costs less control effort than LQR.

trol law for the air vehicle with τ_v, such that the controlled vehicle can travel a desired distance in the horizontal direction, x_d, through the cellular flow with $\tau_w = \tau_m$ within a minimum time. The optimal solution to this problem, a BBC control law, will be proposed in Section 2.6.1. Subsequently, the performance of FTC will be demonstrated by comparing with the optimal BBC solution in Section 2.6.2. In the end, based on multiple numerical simulations, the dependence of the minimum-time simulation results on the choice of τ_v with respect to τ_m is discussed.

2.6.1 Comparison with Bang-Bang Controller (BBC)

In still fluid, bang-bang control, which requires full use of available control effort, is the optimal solution to minimum-time problem when the control is bounded [188]. In this chapter, since the horizontal and vertical motions of the vehicle are decoupled, we first consider the horizontal vehicle dynamics and ignore the flow disturbance for simplicity. Additionally, the vertical control input, u_y, is assumed to be zero. Assuming that the vehicle state is represented by $\mathbf{x} = [x\ v_x]^T$, the horizontal vehicle dynamics can be expressed as

$$\begin{bmatrix} \dot{x} \\ \dot{v}_x \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -1/\tau \end{bmatrix} \begin{bmatrix} x \\ v_x \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \hat{u}_x \tag{2.33}$$

where $\hat{u}_x = u_x + a_x$, and the horizontal control input is assumed to be bounded,

$$u_x = u_x(t),\ -u_0 \leqslant u_x \leqslant u_0. \tag{2.34}$$

To minimize the time to reach the desired horizontal position, x_d, the cost function can be constructed as,

$$J = \int_{t_0}^{t_f} 1\, dt \tag{2.35}$$

subject to the horizontal vehicle dynamics in (2.33) and boundary conditions,

$$x(t_0) = v_x(t_0) = 0,\ x(t_f) = x_d,\ v_x(t_f) = v_{x_r}, \tag{2.36}$$

where v_{x_r} is the horizontal steady-state velocity achieved by the ideal fast-tracking particle. In general, the optimal control history for this type of minimum-time problem takes the form [214],

$$u_x(t) = \begin{cases} u_0 & t_0 \leqslant t \leqslant t_w \\ -u_0 & t_w < t \leqslant t_f \end{cases} \tag{2.37}$$

where t_w is the switching time, and t_f is the terminal time. The control switches at time t_w and only takes the boundary value. Then, the switching time t_w and the terminal time t_f can be obtained by solving the constrained optimization problem formulated above.

2.6.2 FTC Minimum-Time Case Study 3

In this case study, FTC and BBC are implemented and compared for the control of a thrust-driven vehicle with $\tau_v = 0.225$ s traversing a cellular flow with $\tau_w = 0.15$ s. In the simulation, the FTC-controlled and BBC-controlled vehicles both start from the origin with zero initial velocity, and the control is assumed to be bounded: $-u_0 \leqslant u_x \leqslant u_0$, where $u_0 = 8$ m/s^2. The control objective is to make the air vehicle travel a desired distance in the horizontal direction, $x_d = 15m$, through the cellular flow using minimum time. FTC-controlled air vehicle achieves this objective by following the ideal response of a fast-tracking particle with $\tau_m = \tau_w$.

By solving the constrained optimization problem with the flow disturbance ignored in Section 2.6.1, we find that, if the control switches at time $t_w = 6.54$ s, the BBC-controlled air vehicle can reach the desired horizontal position at time $t_f = 6.55$ s in still fluid. Although we neglect the flow disturbance for simplicity when deriving the optimal bang-bang control law, the flow effects on the air vehicle are still considered in the simulation. As shown in Fig. 2.9, both the FTC-controlled and BBC-controlled air vehicles start at the same time from the origin, go through a transitional period in the cellular flow, and travel horizontally in the end. However, compared to the FTC-controlled one, the BBC-controlled air

vehicle significantly deviates from the desired horizontal direction at first, and takes a longer transitional time to reach an approximate equilibrium state.

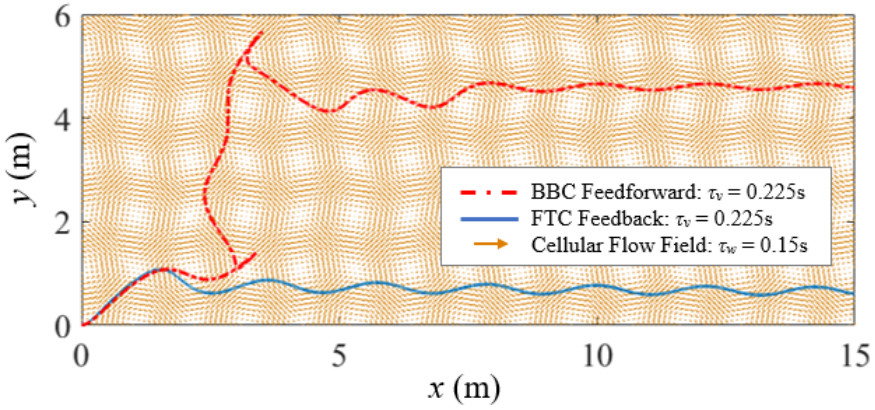


Figure 2.9: Case study 3: the trajectory comparison of the FTC-controlled and BBC-controlled vehicles with $\tau_v = 0.225$ s traversing a cellular flow with $\tau_w = 0.15$ s demonstrates that both vehicles achieve the control objective of reaching a desired horizontal position, $x_d = 15$ m.

Fig. 2.10 shows the horizontal velocity and position time histories of the FTC-controlled and BBC-controlled air vehicles. The time-average horizontal velocity of the BBC-controlled air vehicle is smaller than that of the FTC-controlled one. Consequently, in the simulation, the FTC-controlled air vehicle reaches the desired position, $x_d = 15$ m, at time $t_a = 2.47$ s, while the BBC-controlled one reaches the desired position at time $t_b = 5.09$ s. Even though bang-bang control law is the optimal solution to the benchmark minimum-time problem for bounded control inputs, the FTC-controlled air vehicle achieves the control objective of reaching a desired horizontal position through the cellular flow using less time compared to the BBC-controlled one.

The comparison of the quadratic control usage, $\mathbf{u}^T\mathbf{u} = u_x^2 + u_y^2$, of FTC and BBC is shown in Fig. 2.11. Given that the FTC-controlled and BBC-controlled vehicles travel the same desired horizontal distance, the integral quadratic control usage of FTC over the entire period from time $t_0 = 0$ s to time $t_a = 2.47$ s

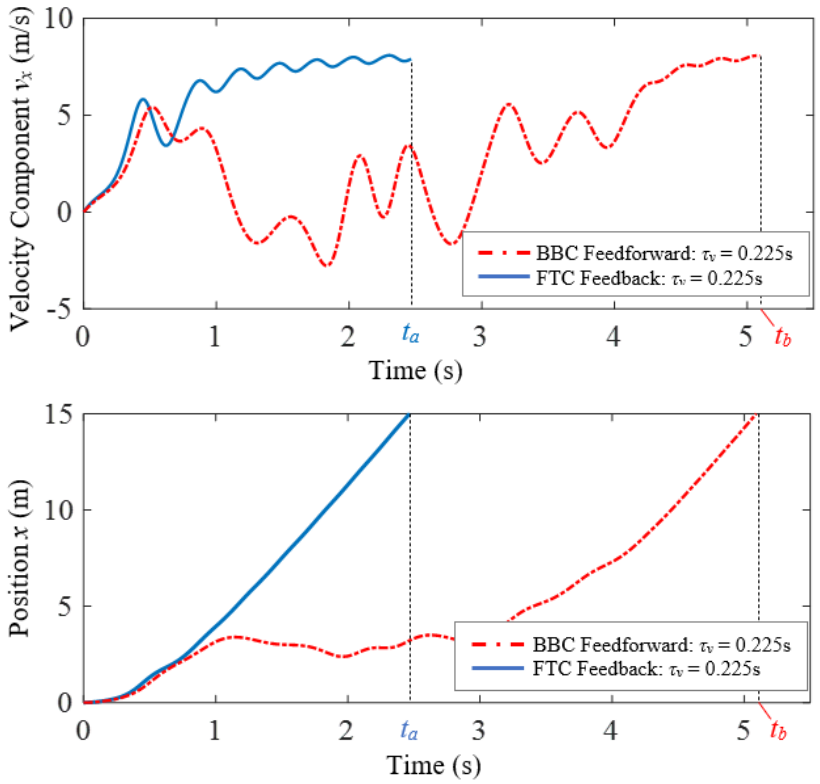


Figure 2.10: Case study 3: the comparison of horizontal velocity and position time histories of the FTC-controlled and BBC-controlled vehicles shows that the FTC-controlled vehicle reaches the desired horizontal position much faster than the BBC-controlled one.

is $C_f = 266.73$ m^2/s^3, while the integral quadratic control usage of BBC over the entire period from time $t_0 = 0$ s to time $t_b = 5.09$ s is $C_b = 325.89$ m^2/s^3. The comparison of FTC and BBC performance is shown in Table 2.1. Compared to the BBC-controlled one, the FTC-controlled air vehicle achieves the control objective of reaching a desired horizontal position through the cellular flow not only within a smaller period of time, but also with less control effort.

Table 2.1: Case study 3: the comparison of FTC and BBC performance.

Controller	t_f **(s)**	$x(t_f)$ **(m)**	C **(m^2/s^3)**
FTC	2.47	15.00	266.73
BBC	5.09	15.00	325.89

In this case study, the ratio of inertial response time is $\eta_\tau = 1.5$. Similarly,

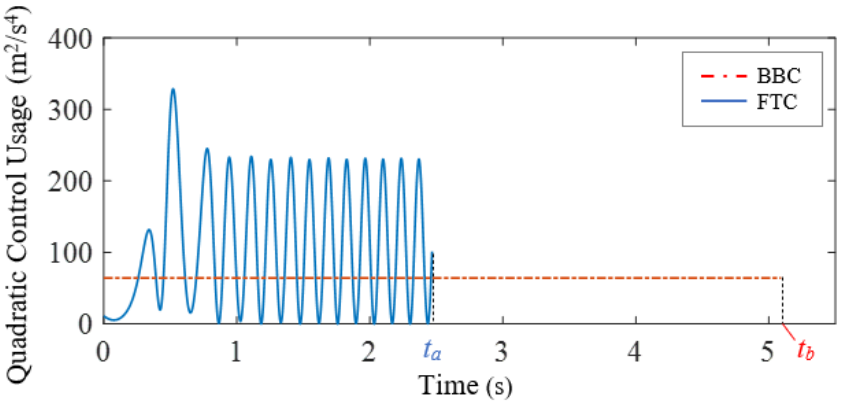


Figure 2.11: Case study 3: the comparison of the FTC and BBC quadratic control usage.

to quantify the relative time savings of FTC compared to BBC, the ratio of total time is defined as

$$\eta_t = \frac{t_b}{t_a} = 2.06 \tag{2.38}$$

where t_a denotes the total time spent by the FTC-controlled vehicle to reach the desired horizontal position, and t_b denotes that of the BBC-controlled vehicle. To analyze how the ratio of total time, η_t, changes as a function of the normalized inertial response time of the vehicle, η_τ, we perform multiple numerical simulations with η_τ ranging from 0.1 to 10. As shown in Fig. 2.12, all the data points located in the grey shaded area above the horizontal dashed line at $\eta_t = 1$ correspond with simulations in which the FTC-controlled vehicle spends less time traversing the flow compared with the BBC-controlled one. The critical point where $\eta_t = 1$ locates approximately at $\eta_\tau^* \approx 6$. The yellow circular marker represents the simulation chosen as the case study in this section. As η_τ increases, η_t grows at first, reaches a maximum, and then starts to decrease when $\eta_\tau \approx 3.5$. Moreover, there exists a horizontal asymptote for the polynomial fit: η_t approaches 0.8 as η_τ keeps increasing.

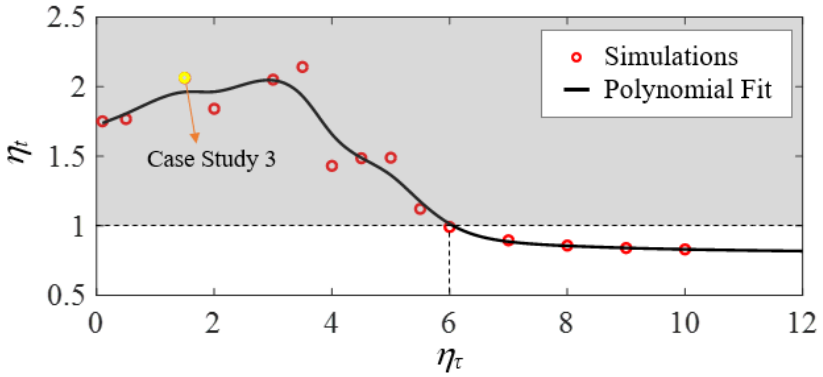


Figure 2.12: The ratio of total time (η_t) as a function of the normalized inertial response time of the vehicle (η_τ) for $\tau_m = \tau_w$ is shown by performing many simulations (circular markers) corresponding to different values of η_τ, and by performing a polynomial fit (black line) demarking case studies in which FTC uses less time than BBC (grey shaded area).

CHAPTER 3

FLOW PARAMETER ESTIMATION

3.1 Introduction

Inspired by particle transport theory [5] and related experiments with water droplets settling in air turbulence [73], we have recently developed a new fast-tracking-particle-inspired flow-aided control approach for air vehicles traversing turbulent flows [223, 18]. As presented in Chapter 2, this FTC control design allows the vehicle to exploit beneficial flow structures, thereby reducing the cost of time and energy to traverse turbulent flows. However, the method requires prior knowledge of the vortex timescale of the turbulent flow. In this chapter, techniques to estimate the turbulent flow parameters, including the mean velocity and vortex length scale and timescale, using noisy on-board measurements obtained from air vehicles traversing turbulent flows will be presented.

Estimating the flow structures of wind and ocean currents helps to guarantee the safety, robustness and efficiency of vehicle navigation in highly turbulent environments [217, 66]. Researchers have already successfully estimated the velocity profiles of wind and water fields using the vehicle's corresponding dynamic responses [106, 156]. Based on the measured flow velocity data, a flow-aided path planning algorithm is developed for unmanned underwater vehicle (UUV) navigation in turbulent ocean currents [182]. To minimize the time or energy consumption of a path planner, beneficial flow structures are often intentionally taken advantage of, while disadvantageous flow structures are usually evaded by autonomous vehicles [121]. In addition, a wind disturbance estimation algorithm is developed for quadcopters, and the estimated data is

then used to optimize the position control and wind disturbance compensation systems [208]. However, most of these existing flow estimation approaches for wind and ocean currents focus primarily on computing the precise flow velocity components, which have large measurement errors and require high computational cost [80, 110].

Turbulent flow structures can be characterized by some key flow parameters, including the mean velocity, the vortex timescale and the vortex length scale [146]. In this chapter, we focus on developing a method to estimate these turbulent flow parameters, and validating it on a two-dimensional cellular flow model. However, this method can be potentially extended to other complex turbulence models. Without prior information of the exact flow parametric model, the novelty of this work is to only estimate the key flow parameters by determining the weights that represent which candidate functions are active in the nonlinear dynamical equations via SINDy. To approximate an unknown dynamic model, SINDy is a computationally efficient approach that considers the unknown nonlinear function as a sparsely weighted combination of all potential user-defined candidate functions, such as constant, polynomial and trigonometric terms [97]. The weights are then determined by solving separate optimization problems [20]. Until now, SINDy has been demonstrated to perform extremely well in identifying various classical ordinary differential equation (ODE) and partial differential equation (PDE) models in fluid dynamics, such as the Lorenz deterministic nonperiodic flow model [115] and the PDE model for the cylinder wake [134].

This chapter is organized as follows. In Section 3.2, relevant background knowledge on SINDy is introduced. The problem of estimating the flow pa-

rameters based on noisy on-board measurements of an air vehicle traversing cellular flows without prior knowledge of the exact flow parametric model is formulated in Section 3.3. Section 3.4 illustrates an approach to estimate the flow parameters via SINDy. In Section 3.5, the proposed method is validated on different measurement data sets with various initial conditions and flow parameters, and numerical simulation results are correspondingly presented.

3.2 Background on Sparse Identification of Nonlinear Dynamics (SINDy)

In this chapter, the SINDy-based flow parameter estimation algorithms are validated on a two-dimensional cellular flow model as introduced in 2.2.2. SINDy is a method that identifies an unknown nonlinear dynamical equation as a sparse combination of all potential user-defined candidate functions [20]. In general, we first consider a nonlinear dynamical equation of the form,

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t)],\ \mathbf{x}(t_0) = \mathbf{x}_0 \tag{3.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the state of the dynamical system, $\mathbf{f}$ is a nonlinear function, and $\mathbf{x}_0$ is the initial condition. To sparsely identify the function $\mathbf{f}$ based on noisy on-board measurements, the state $\mathbf{x}$ and the state derivative $\dot{\mathbf{x}}$ are sampled at times $t_1, t_2, \cdots, t_m$, and then arranged into matrices, $\mathbf{X} \in \mathbb{R}^{m\times n}$ and $\dot{\mathbf{X}} \in \mathbb{R}^{m\times n}$, respectively,

$$\mathbf{X} = [\mathbf{x}(t_1)\ \ \mathbf{x}(t_2)\ \cdots\ \mathbf{x}(t_m)]^T \tag{3.2a}$$

$$\dot{\mathbf{X}} = [\dot{\mathbf{x}}(t_1)\ \ \dot{\mathbf{x}}(t_2)\ \cdots\ \dot{\mathbf{x}}(t_m)]^T \tag{3.2b}$$

We then construct a set of candidates $\mathbf{\Theta}(\mathbf{X}) \in \mathbb{R}^{m \times k}$ containing k user-defined candidate functions, each column of which represents the sampled values of the corresponding candidate function at times $t_1, t_2, \cdots, t_m$:

$$\mathbf{\Theta}(\mathbf{X}) = \begin{bmatrix} f_1(\mathbf{X}) & f_2(\mathbf{X}) & \cdots & f_k(\mathbf{X}) \end{bmatrix} \tag{3.3}$$

where the potential candidate functions are denoted as $f_i,\ i = 1, 2, \cdots, k$. After that, separate optimization problems can be formulated to identify the relative importance of these candidate functions by determining the sparse coefficients in the weight matrix $\mathbf{W}$:

$$\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{W} \tag{3.4}$$

where $\mathbf{W} \in \mathbb{R}^{k \times n}$ is the weight matrix representing the active level of candidate functions in the unknown nonlinear dynamics [97]. Therefore, SINDy can determine the weight matrix $\mathbf{W}$ by solving optimization problems via linear regression, and approximate the nonlinear function $\mathbf{f}$ that maps the state $\mathbf{x}$ to the state derivative $\dot{\mathbf{x}}$ as a sparsely weighted linear combination of candidate functions.

3.3 Problem Formulation

Given a thrust-driven air vehicle traversing a cellular flow with mean velocity U_0 and vortex length scale L_w, we consider the problem of estimating the flow parameters $\mathbf{p} = [U_0\ L_w]^T$ from noisy flight data, including the vehicle state $\mathbf{x} \in \mathbb{R}^n$ and the vehicle state derivative $\dot{\mathbf{x}} \in \mathbb{R}^n$, collected at times $t_1, t_2, \cdots, t_m$ in the form,

$$\dot{\mathbf{x}}(t_i) = \mathbf{f}[\mathbf{x}(t_i); \mathbf{p}] + \mathbf{n}(t_i),\ i = 1, 2, \cdots, m \tag{3.5}$$

where the nonlinear function $\mathbf{f}$ represents the vehicle dynamics, and the process noise $\mathbf{n} \in \mathbb{R}^n$ is modeled as a vector of independent and identically distributed

zero-mean Gaussian noise. In this chapter, the vehicle state $\mathbf{x}$ is assumed to be fully measurable, and the state derivative $\dot{\mathbf{x}}$ is then numerically estimated using finite difference approximation. If this assumption is not met and derivatives cannot be estimated using finite differences, an optimal on-board state estimator can be used instead [56]. Sampled vehicle states $\mathbf{x}$ and state derivatives $\dot{\mathbf{x}}$ at times $t_1, t_2, \cdots, t_m$ are then arranged into matrices, $\mathbf{X} \in \mathbb{R}^{m\times n}$ and $\dot{\mathbf{X}} \in \mathbb{R}^{m\times n}$, respectively.

Based on prior basic knowledge of the cellular flow characteristics, we construct a set of candidates $\boldsymbol{\Theta}(\mathbf{X}) \in \mathbb{R}^{m\times k}$ containing k user-defined candidate functions. The nonlinear vehicle dynamics $\mathbf{f}$ is then approximated as a sparsely weighted linear combination of all potential user-defined candidate functions:

$$\dot{\mathbf{X}} = \boldsymbol{\Theta}(\mathbf{X})\mathbf{W} + \mathbf{N} \tag{3.6}$$

where $\mathbf{W} \in \mathbb{R}^{k\times n}$ is the weight matrix representing the active level of candidate functions, and the noise $\mathbf{N} \in \mathbb{R}^{m\times n}$ is modeled as a matrix of independent and identically distributed zero-mean Gaussian noise. After that, we set up n separate optimization problems to identify the relative importance of these candidate functions by determining the sparse column vectors of coefficients $\mathbf{w}_j \in \mathbb{R}^k$ in the weight matrix $\mathbf{W}$:

$$\mathbf{w}_j = \underset{\mathbf{w}_j}{argmin} \|\dot{\mathbf{x}}_j - \boldsymbol{\Theta}(\mathbf{x}_j)\mathbf{w}_j\|_2^2, \ j = 1, 2, \cdots, n \tag{3.7}$$

where $\mathbf{x}_j$ and $\dot{\mathbf{x}}_j$ are the jth column of the sampled matrices $\mathbf{X}$ and $\dot{\mathbf{X}}$ respectively. The unknown vector of cellular flow parameters $\mathbf{p} = [U_0 \ L_w]^T$ can then be identified by analyzing the weights statistically based on prior knowledge of the cellular flow characteristics and vehicle dynamics. Here, we assume that the cellular flow parameters U_0 and L_w are both unknown bounded constants, so there exist a lower bound and an upper bound for both flow parameters.

3.4 SINDy-Based Flow Parameter Estimation

The flow parameter estimation algorithms presented in this chapter contain four essential steps: collecting the flight data from numerical simulations, constructing a set of potential candidate functions, solving sparse optimization problems to identify the active candidate functions in the nonlinear dynamics, and estimating the flow parameters statistically. In this chapter, the parameter estimation algorithms are developed and validated on a point-mass thrust-driven air vehicle model, as described in Subsection 3.4.1. The flight data is simulated according to this air vehicle model. Then, the procedure for constructing a set of potential candidate functions based on prior knowledge of the target problem is presented in Subsection 3.4.2. Finally, in Subsection 3.4.3, sparse optimization problems are formulated in order to determine the active level of candidate functions, followed by the corresponding statistical analysis and parameter estimation.

3.4.1 Aerial Vehicle Model and Flight Data Simulation

There is a significant precedent for treating autonomous vehicles, such as fixed-wing aircraft [32, 79], quadcopters [148, 147] and unmanned ground robots [10, 176], as point masses. In this chapter, the assumption of treating an air vehicle traversing a cellular flow as a point mass holds when the order of magnitude of the vehicle size L is much smaller than that of the vortex length scale L_w of the cellular flow. Inspired by the dynamics of spherical inertial particles [124], the two-dimensional governing equation for the dynamics of a thrust-driven

point-mass air vehicle of mass m and size L takes the form,

$$m\dot{\mathbf{v}}(t) = -3\pi L\mu[\mathbf{v}(t) - \mathbf{w}(t)] + \mathbf{T} \tag{3.8}$$

where $\mathbf{v} \in \mathbb{R}^2$ is the velocity of the air vehicle, μ is the dynamic viscosity of surrounding fluid, $\mathbf{w} \in \mathbb{R}^2$ is the flow velocity, and $\mathbf{T} \in \mathbb{R}^2$ is a constant thrust. In this model, the air vehicle is subject to a linear Stokes drag force. This assumption holds when the flow is incompressible and the vehicle size is sufficiently small so that the Reynolds number for the vehicle traversing the flow is smaller than one. The inertial response time τ of the vehicle to changes in surrounding flow conditions is given by

$$\tau \equiv \frac{m}{3\pi L\mu} \tag{3.9}$$

Therefore, combining (3.8) and (3.9), the ideal two-dimensional dynamic model of the point-mass air vehicle driven by a constant thrust can be rearranged as

$$\dot{\mathbf{v}}(t) = \frac{1}{\tau}[\mathbf{w}(t) - \mathbf{v}(t)] + \mathbf{a}_t \tag{3.10}$$

where $\mathbf{a}_t = \frac{1}{m}\mathbf{T} = [a_x \; a_y]^T$ is the acceleration produced by the constant thrust $\mathbf{T}$. Based on the cellular flow model in (2.5), the flow velocity can be written as a periodic function $\mathbf{\Phi}$ of the vehicle's position x and y, given the flow parameters $\mathbf{p} = [U_0 \; L_w]^T$:

$$\mathbf{\Phi}(x, y; \mathbf{p}) = \begin{bmatrix} w_x \\ w_y \end{bmatrix} = U_0 \begin{bmatrix} f_1(x, y; L_w) \\ f_2(x, y; L_w) \end{bmatrix} \tag{3.11}$$

where $\mathbf{\Phi}$ is an unknown sparse function in the space of all potential candidate functions, and f_1 and f_2 are unknown periodic functions. Given that the maximum magnitude of the flow velocity cannot exceed U_0, the maximum amplitudes of f_1 and f_2 are both $\sqrt{2}/2$. Combining (3.10) and (3.11), the dynamic

model of a point-mass thrust-driven air vehicle with the inertial response time τ traversing a cellular flow with the mean velocity U_0 and vortex length scale L_w can be written in standard state-space form,

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{1}{\tau} & 0 & 0 & 0 \\ 0 & -\frac{1}{\tau} & 0 & 0 \end{bmatrix} \mathbf{x} + \frac{U_0}{\tau} \begin{bmatrix} 0 \\ 0 \\ f_1(x, y; L_w) \\ f_2(x, y; L_w) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ a_x \\ a_y \end{bmatrix} \tag{3.12}$$

where $\mathbf{x} = [x \ y \ v_x \ v_y]^T \in \mathbb{R}^4$ is the state vector, and a_x and a_y are the horizontal and vertical components of the acceleration produced by constant thrust. In numerical simulations, the vehicle state $\mathbf{x}$ is collected at times $t_1, t_2, \cdots, t_{m+1}$, and the state derivative $\dot{\mathbf{x}}$ is then numerically estimated using finite difference approximation:

$$\dot{\mathbf{x}}(t_i) = \frac{\mathbf{x}(t_{i+1}) - \mathbf{x}(t_i)}{t_{i+1} - t_i}, \ i = 1, 2, \cdots, m \tag{3.13}$$

A more accurate alternative is to use an optimal estimator for the state derivative estimation, but this is also more computationally expensive. The vehicle states $\mathbf{x}$ and state derivatives $\dot{\mathbf{x}}$ collected at times $t_1, t_2, \cdots, t_m$ are then arranged into matrices, $\mathbf{X} \in \mathbb{R}^{m \times 4}$ and $\dot{\mathbf{X}} \in \mathbb{R}^{m \times 4}$, respectively:

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 & \mathbf{X}_2 & \mathbf{X}_3 & \mathbf{X}_4 \end{bmatrix} = \begin{bmatrix} x_1(t_1) & x_2(t_1) & x_3(t_1) & x_4(t_1) \\ x_1(t_2) & x_2(t_2) & x_3(t_2) & x_4(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ x_1(t_m) & x_2(t_m) & x_3(t_m) & x_4(t_m) \end{bmatrix} \tag{3.14a}$$

$$\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{X}}_1 & \dot{\mathbf{X}}_2 & \dot{\mathbf{X}}_3 & \dot{\mathbf{X}}_4 \end{bmatrix} = \begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) & \dot{x}_3(t_1) & \dot{x}_4(t_1) \\ \dot{x}_1(t_2) & \dot{x}_2(t_2) & \dot{x}_3(t_2) & \dot{x}_4(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ \dot{x}_1(t_m) & \dot{x}_2(t_m) & \dot{x}_3(t_m) & \dot{x}_4(t_m) \end{bmatrix} \tag{3.14b}$$

where $\mathbf{X}_i \in \mathbb{R}^m$ and $\dot{\mathbf{X}}_i \in \mathbb{R}^m$, $i = 1, 2, 3, 4$, are the sampled time histories of the four states and state derivatives respectively.

3.4.2 Candidate Functions

SINDy approximates the nonlinear equation as a sparsely weighted linear combination of user-defined candidate functions. As a fist step, we construct a group of candidate functions $\mathbf{\Theta}(\mathbf{X})$ based on prior basic knowledge of the target problem. As introduced in Section III, the vortex length scale L_w is defined as the vortex's turnover time in the cellular flow. Since the unknown equations f_1 and f_2 in (3.11) are both periodic functions of position x and y in the world frame, the ultimate candidate function f_{cand} should be in the set of functions containing trigonometric terms with a period of $2L_w$ or the product of them:

$$\begin{aligned} f_{cand} \in \{&\sin\frac{\pi x}{L_w}, \sin\frac{\pi y}{L_w}, \cos\frac{\pi x}{L_w}, \cos\frac{\pi y}{L_w}, \sin\frac{\pi x}{L_w}\sin\frac{\pi y}{L_w}, \\ &\sin\frac{\pi x}{L_w}\cos\frac{\pi y}{L_w}, \cos\frac{\pi x}{L_w}\sin\frac{\pi y}{L_w}, \cos\frac{\pi x}{L_w}\cos\frac{\pi y}{L_w}\} \end{aligned} \tag{3.15}$$

Given that the vortex length scale L_w is a constant within the range from a lower bound L_{lb} to an upper bound L_{ub}, the ultimate candidate length scale L_{cand} should be in the following set containing all potential discretized length scales:

$$L_{cand} \in \{L_{lb}, L_{lb} + dL, L_{lb} + 2dL, \cdots, L_{ub}\} \tag{3.16}$$

where dL is the incremental length scale. After choosing candidate trigonometric functions and length scales, we can obtain the complete library of candidate

functions $\boldsymbol{\Theta}(\mathbf{X})$ containing all potential candidates:

$$\boldsymbol{\Theta}(\mathbf{X}) = \begin{bmatrix} \sin\left(\frac{\pi}{L_{cand}}\mathbf{X}_1\right)^T \\ \sin\left(\frac{\pi}{L_{cand}}\mathbf{X}_2\right)^T \\ \cos\left(\frac{\pi}{L_{cand}}\mathbf{X}_1\right)^T \\ \cos\left(\frac{\pi}{L_{cand}}\mathbf{X}_2\right)^T \\ \left[\sin\left(\frac{\pi}{L_{cand}}\mathbf{X}_1\right) \otimes \sin\left(\frac{\pi}{L_{cand}}\mathbf{X}_2\right)\right]^T \\ \left[\sin\left(\frac{\pi}{L_{cand}}\mathbf{X}_1\right) \otimes \cos\left(\frac{\pi}{L_{cand}}\mathbf{X}_2\right)\right]^T \\ \left[\cos\left(\frac{\pi}{L_{cand}}\mathbf{X}_1\right) \otimes \sin\left(\frac{\pi}{L_{cand}}\mathbf{X}_2\right)\right]^T \\ \left[\cos\left(\frac{\pi}{L_{cand}}\mathbf{X}_1\right) \otimes \cos\left(\frac{\pi}{L_{cand}}\mathbf{X}_2\right)\right]^T \end{bmatrix}^T = \begin{bmatrix} \sin\left(\frac{\pi}{L_{lb}}\mathbf{X}_1\right)^T \\ \sin\left(\frac{\pi}{L_{lb}+dL}\mathbf{X}_1\right)^T \\ \vdots \\ \sin\left(\frac{\pi}{L_{ub}}\mathbf{X}_1\right)^T \\ \vdots \\ \left[\cos\left(\frac{\pi}{L_{lb}}\mathbf{X}_1\right) \otimes \cos\left(\frac{\pi}{L_{lb}}\mathbf{X}_2\right)\right]^T \\ \left[\cos\left(\frac{\pi}{L_{lb}+dL}\mathbf{X}_1\right) \otimes \cos\left(\frac{\pi}{L_{lb}+dL}\mathbf{X}_2\right)\right]^T \\ \vdots \\ \left[\cos\left(\frac{\pi}{L_{ub}}\mathbf{X}_1\right) \otimes \cos\left(\frac{\pi}{L_{ub}}\mathbf{X}_2\right)\right]^T \end{bmatrix}^T \tag{3.17}$$

where $\mathbf{X}_1 \in \mathbb{R}^m$ and $\mathbf{X}_2 \in \mathbb{R}^m$ are the sampled time histories of the vehicle position x and y at times $t_1, t_2, \cdots, t_m$ respectively, and the symbol $\otimes$ represents element-wise multiplication. Each column of $\boldsymbol{\Theta}(\mathbf{X})$ represents the sampled values of the corresponding candidate function at times $t_1, t_2, \cdots, t_m$. In (3.17), each type of trigonometric candidate functions corresponds with the same number of elements as the set of all potential discretized length scales. For convenience, in the next subsection, the ith trigonometric candidate function in (3.15) is denoted as f_{cand_i}, and the jth candidate length scale in (3.16) is denoted as L_{cand_j}. It should be noted that choosing the range of L_{cand} and the incremental length scale dL is essential, because the size of candidate function set $\boldsymbol{\Theta}(\mathbf{X})$ cannot exceed the length of measurement data set in order to make the formulated optimization problems solvable.

3.4.3 Optimization and Parameter Estimation

Given the user-defined candidate function set $\mathbf{\Theta}(\mathbf{X})$, we set up n separate optimization problems to identify the relative importance of these candidate functions by determining the sparse column vectors of coefficients $\mathbf{w}_j$ in the weight matrix $\mathbf{W}$ via least squares [20]:

$$\mathbf{w}_j = \underset{\mathbf{w}_j}{argmin} \|\dot{\mathbf{x}}_j - \mathbf{\Theta}(\mathbf{x}_j)\mathbf{w}_j\|_2^2, \ j = 1, 2, \cdots, n \tag{3.18}$$

where $\mathbf{x}_j$ and $\dot{\mathbf{x}}_j$ are the jth column of the sampled matrices $\mathbf{X}$ and $\dot{\mathbf{X}}$ respectively. Each element of $\mathbf{W}$, denoted as $w(f_{cand_i}, L_{cand_j})$, corresponds with the ith candidate function and the jth candidate length scale. Then, all weights corresponding to a certain L_{cand_j} are summed up, and the estimated length scale $\hat{L}_w$ is the candidate length scale with the highest weight sum:

$$\hat{L}_w = \underset{L_{cand_j}}{argmax} \sum_i w(f_{cand_i}, L_{cand_j}) \tag{3.19}$$

Given the ultimate goal to estimate the flow parameters U_0 and L_w, it is not necessary to identify the exact form of periodic functions f_1 and f_2. After obtaining the estimated length scale $\hat{L}_w$, we update the set of candidate functions accordingly. The updated set of candidates, denoted as $\mathbf{\Theta}^*$, contains all periodic candidate functions with $\hat{L}_w$ only. Therefore, the size of the updated candidate set $\mathbf{\Theta}^*$ is significantly smaller than that of the original $\mathbf{\Theta}$. Using SINDy, we can obtain the expected periodic functions, denoted as $\bar{f}_1$ and $\bar{f}_2$, by calculating the weighted sum of all periodic candidate functions with non-zero weights. Given that the maximum magnitude of the flow velocity cannot exceed U_0, the maximum amplitudes of identified f_1 and f_2 should both be $\sqrt{2}/2$. Hence, both expected periodic functions $\bar{f}_1$ and $\bar{f}_2$ have to be normalized to meet this re-

quirement:

$$f_i^* = \sqrt{2} \cdot \frac{\bar{f}_i - \frac{1}{2}[max(\bar{f}_i) + min(\bar{f}_i)]}{max(\bar{f}_i) - min(\bar{f}_i)},\ i = 1, 2 \tag{3.20}$$

where f^* is the normalized periodic function, and $\bar{f}$ is the expected periodic function. Plugging the normalized periodic function $\mathbf{f}^* = [f_1^*\ f_2^*]^T$ back in (3.11), we can obtain the flow velocity components in the form,

$$\mathbf{\Phi}(\mathbf{X}) = U_0 \mathbf{f}^*(\mathbf{X}) \tag{3.21}$$

where $\mathbf{X}$ contains sampled vehicle states, and $\mathbf{\Phi}(\mathbf{X})$ is obtained from sampled vehicle state derivatives based on prior basic knowledge of the cellular flow characteristics and the vehicle dynamics in (3.12). Then, U_0 can be estimated by averaging the coefficients in front of the normalized periodic function $\mathbf{f}^*(\mathbf{X})$:

$$\hat{U}_0 = \langle \mathbf{\Phi}(\mathbf{X})[\mathbf{f}^*(\mathbf{X})]^{-1} \rangle \tag{3.22}$$

Given the estimated mean velocity $\hat{U}_0$ and vortex length scale $\hat{L}_w$, the estimated vortex timescale $\hat{\tau}_w$ can then be calculated as

$$\hat{\tau}_w = \frac{\hat{L}_w}{\hat{U}_0} \tag{3.23}$$

3.5 Parameter Estimation Results

In numerical simulations, the capability of the presented method to estimate the mean velocity U_0, the vortex length scale L_w and the vortex timescale τ_w of the cellular flow is demonstrated by testing the algorithms on different simulated data sets with various flow parameters and initial conditions. First, the performance of parameter estimation algorithms is validated on a simulated air vehicle traversing a cellular flow with the mean velocity $U_0^* = 15.43$ m/s, the

vortex length scale $L_w^* = 4.21$ m and the vortex timescale $\tau_w^* = L_w^*/U_0^* = 0.27$ s with a randomized initial position and zero initial velocity. It is assumed that the vortex length scale $\hat{L}_w$ to estimate is a constant within the range from a lower bound $L_{lb} = 1$ m to an upper bound $L_{ub} = 12$ m. The incremental length scale dL is chosen to be 0.2 m. As shown in Fig. 3.1, the estimated vortex length scale with the highest weight is $\hat{L}_w = 4.20$ m. The corresponding estimated mean velocity is $\hat{U}_0 = 15.95$ m/s, and the resultant estimated vortex time scale is $\hat{\tau}_w = \hat{L}_w/\hat{U}_0 = 0.26$ s. All three estimated cellular flow parameters are close to their true values. Table 3.1 shows that, with reasonable restrictions on the range of L_{cand} and careful choice of incremental length scale dL, the cellular flow parameters U_0, L_w and τ_w can be estimated accurately within a range of error around $\pm 4\%$ for this simulation.

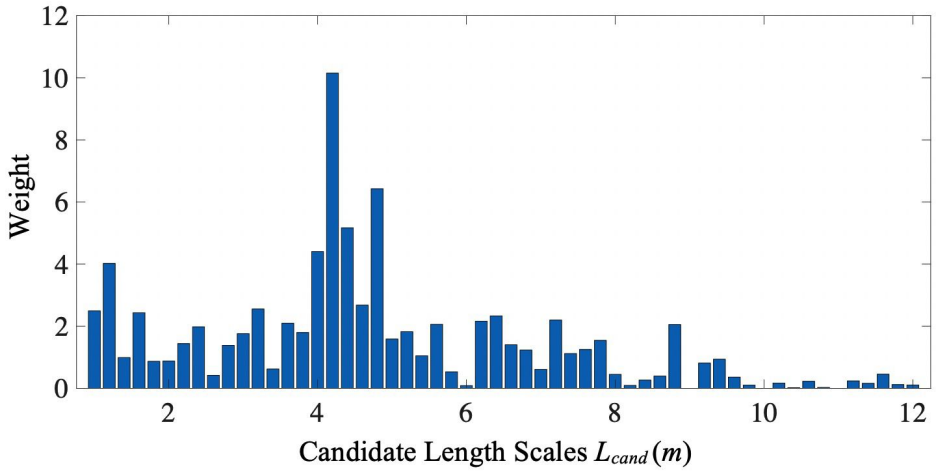


Figure 3.1: Weights of the candidate length scales L_{cand} ranging from a lower bound $L_{lb} = 1$ m to an upper bound $L_{ub} = 12$ m. The estimated vortex length scale with the highest weight is $\hat{L}_w = 4.20$ m.

To evaluate the overall performance, the parameter estimation algorithms are then validated on simulated thrust-driven air vehicles with different initial conditions traversing cellular flows with different parameters. Among 95 measurement data sets, the vortex timescale τ_w^* of the cellular flow is equally spaced

Table 3.1: Comparison of the true and estimated values of the cellular flow parameters.

	U_0 (m/s)	L_w (m)	τ_w (s)
True Value	15.43	4.21	0.27
Estimated Value	15.95	4.20	0.26
Percentage Error (%)	3.37	-0.24	-3.70

within the range from 0.05 s to 1 s, and the vortex length scale L_w^* is a random number. Additionally, the vehicle's initial position is randomized, and its initial velocity is set to zero. The overall performance is shown in Fig. 3.2. At most times, the vortex timescale $\hat{\tau}_w$, the vortex length scale $\hat{L}_w$ and the mean velocity $\hat{U}_0$ can be estimated accurately with a few outliers arising possibly due to measurement noise and inaccuracy caused by estimating the state derivatives via finite difference approximation. Furthermore, when testing the algorithms on simulated data sets with smaller vortex timescales τ_w^*, the deviations of the estimated vortex length scales $\hat{L}_w$ and mean velocities $\hat{U}_0$ from their true values increase. However, some data points in Fig. 3.2 show that deviated estimations of the vortex length scale $\hat{L}_w$ and mean velocity $\hat{U}_0$ are correlated so that their ratio yields relatively good estimations of the vortex timescale $\hat{\tau}_w$. With 95 different simulations and validation tests, the average absolute errors and percentage errors of the three cellular flow parameters are listed in Table 3.2 below. After comparing their average percentage errors, we draw the conclusion that our new parameter estimation algorithms can estimate the vortex length scale $\hat{L}_w$ more accurately than the vortex timescale $\hat{\tau}_w$ and mean velocity $\hat{U}_0$.

Table 3.2: Average absolute and percentage errors of cellular flow parameters.

	U_0 (m/s)	L_w (m)	τ_w (s)
Absolute Error	-4.43	-0.17	0.10
Percentage Error (%)	-18.80	-5.91	33.30

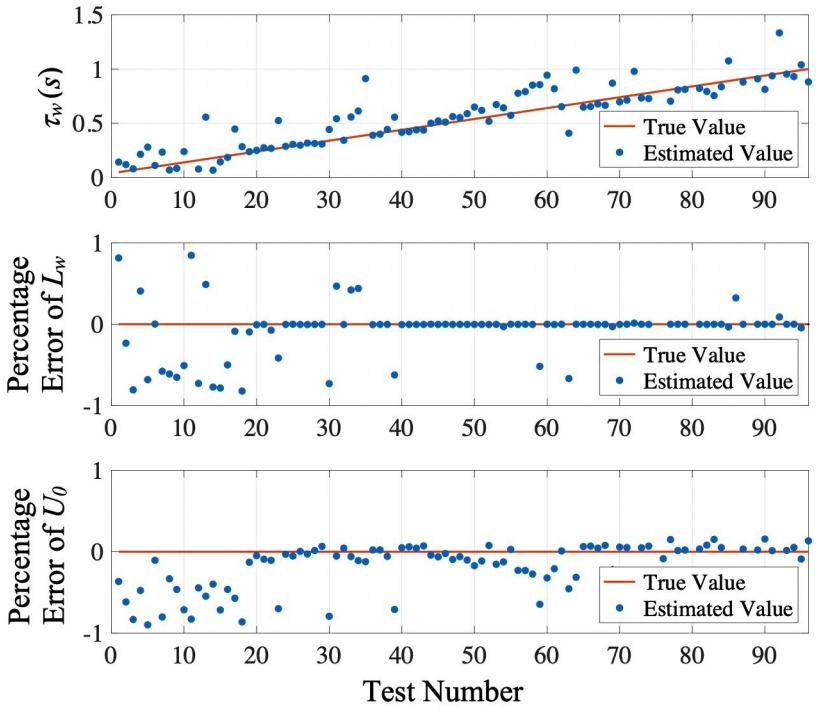


Figure 3.2: Overall performance of the parameter estimation algorithms on different simulated data sets. In general, the vortex timescale $\hat{\tau}_w$, the vortex length scale $\hat{L}_w$ and the mean velocity $\hat{U}_0$ can be accurately estimated with only a few outliers arising.

CHAPTER 4

KERNEL-BASED NEURAL DECODING

4.1 Introduction

In addition to particle transport theory presented in Chapter 2, MAV sensing and control strategies can be extracted from biological systems. Nervous systems of animals can integrate information from multiple sensory modalities, and make rapid and coherent behavioral decisions in complex environments [122, 11]. However, most existing artificial intelligence systems rely on rich but separate modalities of sensory feedback. Typically, they are poorly integrated and predetermined for particular tasks, such as object recognition, action recognition and target tracking [185, 45, 228]. Therefore, there is a massive untapped opportunity for us to reverse-engineer the neural control system that bridges sensory perception and motor control of complex animal behaviors. However, neural decoding has been considered one of the biggest challenges in reverse-engineering the neuromorphic perception and control systems in nature [95, 86], because sensory signals are encoded in low-dimensional neural activities [62], and sparsity and compressive sensing are essential for biological decision-making processes [63]. To extract nonlinear dynamic control strategies from biological neural systems and approximate them via spiking neural network (SNN), we need to decode useful continuous-time signals from spike trains, and use them for downstream control inputs [38, 231].

Spike train decoding is a mathematical problem of inferring external stimuli or biological control signals encoded in sequences of spike timings [19, 103]. It is fundamental and essential for determining the complete biological neural

control system that bridges sparse sensory codes and motor control [203, 26]. However, there is still a debate in the neuroscience community on how sensorimotor information is encoded in spike trains. The traditional *rate coding* scheme, where information is encoded in average firing rate, is well-accepted and has been shown in many different sensory and motor circuits [2, 88]. However, it assumes that most information is encoded in average firing rate, and does not take into account any precise spike timing information [37]. As demonstrated in [153, 155], spike timing encodes more information of a hawk moth's turning behavior than spike count in tethered flight, and is essential for the coordination of muscle pairs. In addition to the *rate coding*, more recent studies have identified and shown evidence for *temporal coding*, which employs temporal features, such as temporal difference, time to the first spike and phase of firing, to uncover the mapping from temporal patterns of spikes to continuous representations [195, 102, 172, 84]. There is growing evidence that relative spike timing information is essential for uncovering the whole biological motor program, including the correlation between neuron pairs. These traditional approaches mentioned above do not actually capture the extra information encoded in the relative spike timings between correlated spike trains [153, 133].

In recent years, kernel tricks have been borrowed from the machine learning community and widely used by neuroscientists to represent spike trains as objects in Hilbert space, and decode the neural signals using well-developed regression methods [137, 140, 111]. In [137], the author proposed a reproducing kernel Hilbert space (RKHS) framework that uses an *instantaneous kernel* to determine similarities between single spike trains directly. This RKHS framework can be formulated by many types of spike train kernel designs, including count kernels [140], linear functional kernels [169], and nonlinear functional kernels

[139]. Gaussian process regression, which assumes a prior Gaussian distribution with its covariance given by the kernel, has also been widely used for spike train decoding [170, 212]. One distinct disadvantage of these kernel-based spike train decoding methods is that they only capture the difference of either spike counts or exact spike timings between spike trains from different motor units, and will not perform well especially when the spike trains correlate with each other.

A hawk moth is an ideal small insect to test the importance of relative spike timing information for neural control, due to unprecedented access through electromyography (EMG) recordings to all the neural signals that control their flight muscles. These insect fliers use only 10 muscles, 5 per wing, to execute robust and agile flight in unsteady environments, which likely put extreme demands on their neural control systems. More importantly, relative spike timing is coordinated across every muscle in the moth's flight control [153]. Hawk moths also integrate multiple sensory modalities to execute this control [165]. In this chapter, we aim to discover the neural control policy for the flight of a tethered hawk moth visually tracking a moving robotic flower as shown in Fig. 4.1, which is an ecologically relevant behavior that moths can execute innately without learning. Unlike the traditional kernel-based approaches summarized above, the new RKHS framework proposed in this chapter is based on the kernel evaluation between every pair of correlated spike trains across the population. The novelty of this new *relative-time kernel* design is that it allows to take into account both single spike train patterns and relative spike timing information among multiple neurons for the first time.

This chapter is organized as follows. Section 4.2 first introduces how we col-

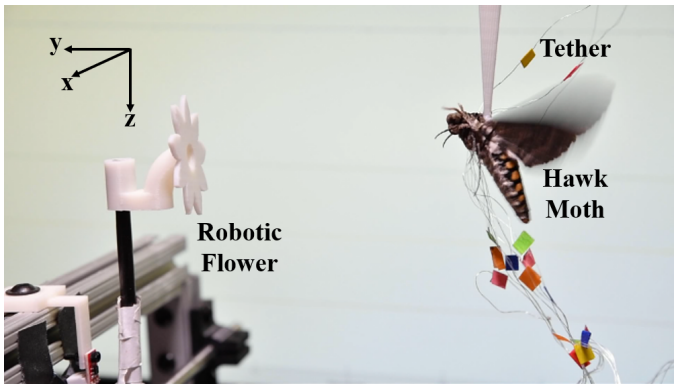


Figure 4.1: Picture of a hawk moth visually tracking a moving robotic flower while tethered to a custom 6-axis F/T transducer.

lect the spike train and control signal data in flower tracking experiments. The spike train decoding problem is then formulated in this section, along with its basic assumptions. The new relative-time kernel design that considers the extra information encoded in relative spike timings among multiple neurons is presented in Section 4.3. In Section 4.4, the performance of the relative-time-kernel-based spike train decoder is demonstrated by comparing to that of benchmark instantaneous-kernel-based and rate-based decoders.

4.2 Problem Formulation

In our experiments, hawk moths ($N = 7$) visually track a robotic flower that oscillates horizontally with a 1-Hz sinusoidal trajectory while tethered to a custom 6-axis F/T transducer. The sampling rate for the experiments is 10^4 Hz, and hawk moths in tethered flight have wing strokes of approximately 50 ms in length. This flower tracking experiment creates a variety of turning forces and torques, because there are about 20 wing strokes per flower oscillation. In this chapter, we aim to uncover the precise mapping from the recorded spike trains

of the 10 primary muscles actuating the moth wings to the resulting forces and torques on the body. More details on the experimental platform and data collection can be found in [153]. The forces and torques, $\mathbf{y} \in \mathbb{R}^6$, are collected at times $t_1, t_2, \cdots, t_n$, and then arranged into a matrix, $\mathbf{Y} \in \mathbb{R}^{n\times 6}$, such that

$$\mathbf{Y} = [\mathbf{y}(t_1) \;\; \mathbf{y}(t_2) \;\; \cdots \;\; \mathbf{y}(t_n)]^T \tag{4.1}$$

To map a spike train containing a sequence of spike times to a continuous variable that can be used for regression, we represent the sequence of spike times as a binned spike train that is changing over time as an user-defined sliding window moves [33, 74, 167]. The larger the bin size, the more information will be stored in the binned spike trains. However, the regression algorithm will also become more computationally expensive. For muscle i, the spike times t_k^i within a certain bin size T before time t are stored in a binned spike train,

$$X^i(t) = \{t_k^i \in (t-T, t]\}, i = 1, 2, \cdots, 10 \text{ and } k \in \mathbb{N}^* \tag{4.2}$$

where k represents spike indices. Similar to the forces and torques in (4.1), the binned spike trains of 10 primary flight muscles are then collected at times $t_1, t_2, \cdots, t_n$, and arranged into a matrix, $\mathbf{X} \in \mathbb{R}^{n\times 10}$, such that

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}^T(t_1) \\ \mathbf{x}^T(t_2) \\ \vdots \\ \mathbf{x}^T(t_n) \end{bmatrix} = \begin{bmatrix} X^1(t_1) & X^2(t_1) & \cdots & X^{10}(t_1) \\ X^1(t_2) & X^2(t_2) & \cdots & X^{10}(t_2) \\ \vdots & \vdots & \ddots & \vdots \\ X^1(t_n) & X^2(t_n) & \cdots & X^{10}(t_n) \end{bmatrix} \tag{4.3}$$

where $\mathbf{x}(t) = [X^1(t)\; X^2(t) \cdots X^{10}(t)]^T$ denotes the output signal vector containing 10 binned spike trains at any given time t. In this chapter, we consider the problem of determining the decoding function, $\mathbf{f}^*$, that minimizes the difference

between the predicted and true resulting forces and torques on the moth body,

$$\mathbf{f}^* = \underset{\mathbf{f}\in\mathcal{H}}{argmin}\{\sum_{i=1}^{n} \left\|\mathbf{y}(t_i) - \mathbf{f}[X^1(t_i), X^2(t_i), \cdots, X^{10}(t_i)]\right\|_2^2 + \lambda\|\mathbf{f}\|_{\mathcal{H}}^2\} \tag{4.4}$$

where $\mathcal{H}$ denotes the Hilbert space, and λ is a tuning parameter for penalized regression.

4.3 Kernel Design

In general, a reproducing kernel Hilbert space (RKHS) can be defined by a symmetric and positive definite Mercer kernel. The input sample, X, is first mapped to the RKHS as a function, $K(X, \cdot)$, obtained by fixing the first coordinate. Then, the inner product of two functions in the RKHS can be computed by a kernel evaluation in the input space,

$$\langle X|X'\rangle_{\mathcal{H}} = K(X, X') \tag{4.5}$$

which brings computational simplicity. In our spike train decoding problem, given a set of binned spike trains, $X^i = \{t_k^i : k = 1, 2, \cdots, m_i\}$, $i = 1, 2, \cdots, 10$, from 10 different primary muscles respectively, every pair of binned spike trains, X^i and X^j, can be represented as a sum of two-dimensional Dirac delta functions,

$$x_{ij}(\sigma, \tau) = \sum_{k_i, k_j} \delta(\sigma - t_{k_i}^i, \tau - t_{k_j}^j) \tag{4.6}$$

which can then be converted to a continuous multivariate function by convolving with a filter h,

$$f_{ij}(\sigma, \tau) = x_{ij} * h = \sum_{k_i, k_j} h(\sigma - t_{k_i}^i, \tau - t_{k_j}^j) \tag{4.7}$$

where i and j denote two different muscles, and k represents spike indices. In this chapter, we choose a two-dimensional Gaussian filter h for the convolution in (4.7),

$$h(\mathbf{v}) = exp(-\frac{1}{2}\mathbf{v}^T\mathbf{\Sigma}^{-1}\mathbf{v}) \tag{4.8}$$

where $\mathbf{v}$ denotes the mean vector, and $\mathbf{\Sigma}$ denotes the covariance matrix. For illustration purposes, Fig. 4.2 shows three binned spike trains, $X^i = \{t_k^i : k = 1, 2, \cdots, m_i\}$, $i = 1, 2, 3$, collected in our flower tracking experiment. If we take two binned spike trains, X^1 and X^3, for example, the continuous multivariate function containing the information of relative spike times between these two spike trains can be represented by a two-dimensional Gaussian distribution as shown in Fig. 4.3.

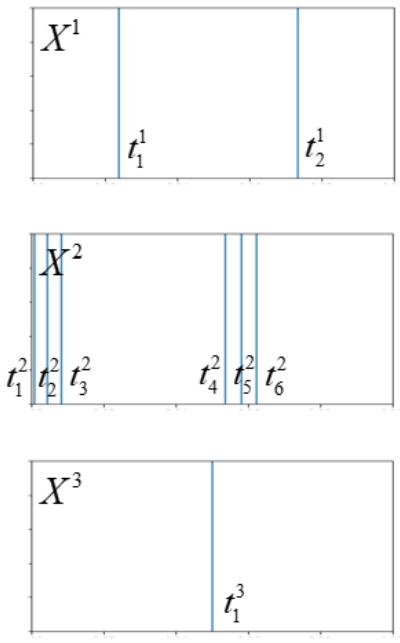


Figure 4.2: An example of three binned spike trains containing the information of exact spike times.

For RKHS regression, the kernel evaluation between two pairs of spike trains

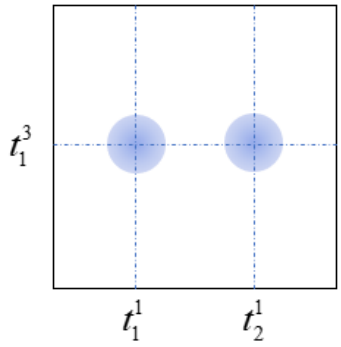


Figure 4.3: An example of the multivariate Gaussian distribution containing the information of relative spike times between spike trains, X^1 and X^3.

can be defined as,

$$K(X^{ij}, X^{ij'}) = \langle f_{ij} | f'_{ij} \rangle = \int_0^T \int_0^T f_{ij}(\sigma, \tau) f'_{ij}(\sigma, \tau) d\sigma d\tau \tag{4.9}$$

where X^{ij} denotes the two-dimensional Gaussian distribution determined by the pair of spike trains, X^i and X^j, the superscript $(\cdot)'$ refers to another pair of spike trains, and T represents the bin size. Then, the final kernel function across the flight muscle population can be given by,

$$K(\mathbf{x}, \mathbf{x}') = \sum_{i,j} K(X^{ij}, X^{ij'}) \tag{4.10}$$

Based on the representer theorem in [168], the evaluation of the decoding function, $\hat{\mathbf{f}}$, at binned spike trains, $\mathbf{Z} \in \mathbb{R}^{l \times 10}$, from 10 muscles in the test data set can be obtained by taking linear combinations of the kernel evaluations,

$$\hat{\mathbf{f}}(\mathbf{Z}) = \mathbb{K}(\mathbf{Z}, \mathbf{X})\boldsymbol{\alpha} \tag{4.11}$$

where $\mathbf{X} \in \mathbb{R}^{n \times 10}$ denotes binned spike trains used for training, $\mathbb{K}_{rs} = K[\mathbf{z}(t_r), \mathbf{x}(t_s)] \in \mathbb{R}^{l \times n}$ is the Gram matrix, and the coefficients, $\boldsymbol{\alpha} \in \mathbb{R}^{n \times 6}$, are given by,

$$\boldsymbol{\alpha} = [\mathbb{K}(\mathbf{X}, \mathbf{X}) + \sigma_n^2 \mathbf{I}]^{-1} \mathbf{Y} \tag{4.12}$$

where $\mathbb{K}_{pq} = K[\mathbf{x}(t_p), \mathbf{x}(t_q)] \in \mathbb{R}^{n\times n}$ is the Gram matrix, σ_n^2 denotes the variance of observation noise, and $\mathbf{Y} \in \mathbb{R}^{n\times 6}$ represents the corresponding forces and torques used for training. Then, we can use this relative-time-kernel-based decoding function to predict the output forces and torques by evaluating the function at arbitrary binned spike trains from 10 flight muscles. This method can be easily applied to much larger neural systems with more than 10 neurons by combining kernel evaluations between more pairs of correlated spike trains in (4.10) accordingly. However, it will become more computationally expensive as the number of neurons increases.

To benchmark our new kernel design, we will compare the performance of our relative-timing-kernel-based regression with that of the traditional rate coding [2, 88] and instantaneous-kernel-based regression [170]. The rate coding method is based on the assumption that average firing rate encodes most information. The instantaneous kernel directly determines similarities between single spike trains, and does not capture relative timing information [137, 140]. For the instantaneous kernel, the binned spike train from muscle i is represented as a combination of Dirac delta functions,

$$x_i(t) = \sum_{k_i} \delta(t - t^i_{k_i}) \tag{4.13}$$

which can then be converted to a continuous function by convolving with a one-dimensional Gaussian filter g,

$$f_i(t) = x_i * g = \sum_{k_i} g(t - t^i_{k_i}) \tag{4.14}$$

The kernel evaluation between two binned spike trains can be defined as,

$$K(X^i, X^{i'}) = \langle f_i | f_i' \rangle = \int_0^T f_i(t) f_i'(t) dt \tag{4.15}$$

Then, the instantaneous kernel function across the flight muscle population can be given by,

$$K(\mathbf{x}, \mathbf{x}') = \sum_i K(X^i, X^{i'}) \tag{4.16}$$

In the following section, we will show the prediction results of relative-time-kernel-based, instantaneous-kernel-based and rate-based regressions.

4.4 Regression Results

For the hawk moth motor program, we choose the size of the sliding window for spike train binning to be 50 ms, which is about the length of each wing stroke. To capture the stroke-to-stroke modulation in one complete flower oscillation cycle, the training data used for RKHS regression should cover at least one second. Therefore, given that the sampling rate of moth experiments is 10^4 Hz, we need to use 10^5 binned spike trains from 10 primary muscles and 6×10^4 output forces and torques collected during the hawk moth's flapping flight for training. Given that the resulting forces and torques do not change dramatically in training and test data sets, we decrease the resolution of training data to reduce the computational complexity by collecting the training data every 20 time steps. Then, we obtain a sequence of binned spike trains from 10 primary muscles at times $t_1, t_2, \cdots, t_{500}$, arrange them into a matrix, $\mathbf{X} \in \mathbb{R}^{500\times 10}$, and correspondingly collect the output forces and torques, $\mathbf{Y} \in \mathbb{R}^{500\times 6}$, within one second as the training data for RKHS regression. Finally, we test our new regression-based decoder on a test data set, $\mathbf{Z} \in \mathbb{R}^{500\times 10}$.

Fig. 4.4 shows the resulting forces and torques predicted by relative-time-kernel-based, instantaneous-kernel-based and rate-based regressions along

with the true values measured in the moth experiment. The resulting forces and torques have been predicted accurately within a permissible range of error by the relative-time-kernel based method. The instantaneous kernel directly determines similarities between single spike trains, and does not capture relative timing information [137, 140]. The rate coding method is based on the assumption that average firing rate encodes most information. Unlike these two traditional methods, our relative-time kernel compares every pair of correlated spike trains across the population, and considers the extra information encoded in relative spike times among different spike trains. As shown in Fig. 4.4, both the relative-time-kernel-based and instantaneous-kernel-based decoders outperform the rate-based decoder significantly. More importantly, the relative-time-kernel-based decoder can capture small changes in forces and moments better than the other two traditional methods, particularly for torque components, T_x and T_y.

In Fig. 4.5, we compare the absolute prediction errors of relative-time-kernel-based, instantaneous-kernel-based and rate-based regressions. The absolute prediction error, e, is defined as,

$$e = |y - \hat{y}| \tag{4.17}$$

where y denotes the true value, and $\hat{y}$ denotes the predicted value. It can be observed that the relative-time-kernel-based decoder can predict the resulting forces and torques more accurately than the other two traditional decoders, especially for the torque prediction. Predicting torques is much harder and more critical than forces as the rotational modes are less stable in moth flapping flight. To determine how well the decoder captures the variance in data, we use the standard deviation of the absolute prediction error, σ_e, and R-squared score, R^2.

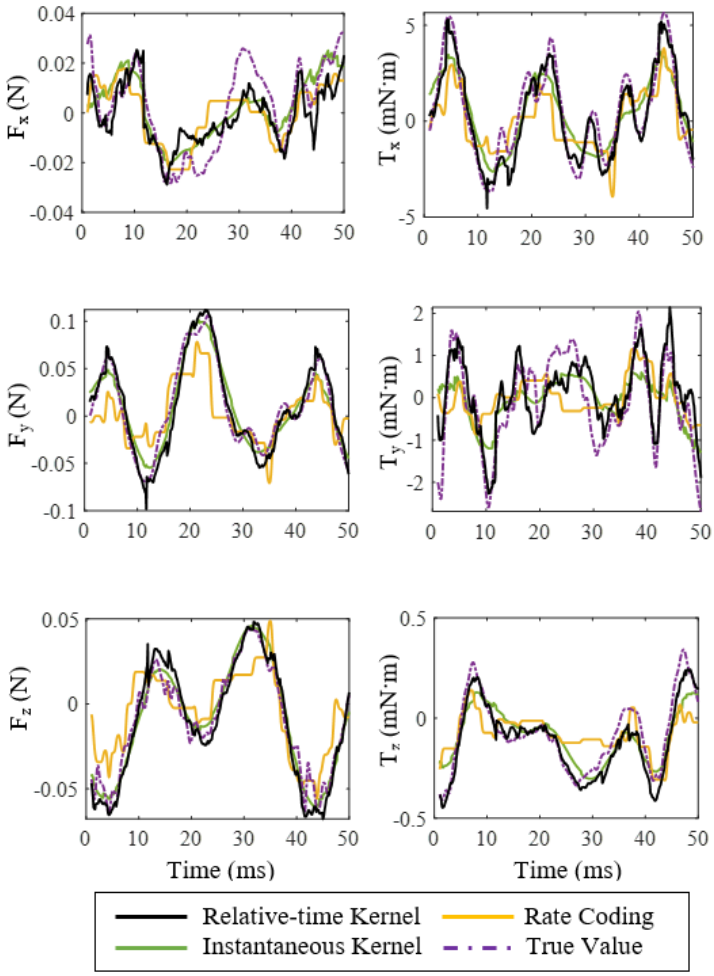


Figure 4.4: Comparison of relative-time-kernel-based, instantaneous-kernel-based and rate-based predictions of resulting forces and torques.

The standard deviation of the absolute error, σ_e, is given by,

$$\sigma_e = \sqrt{\sum_{i=1}^{n}(e_i - \bar{e})^2} \tag{4.18}$$

where $\bar{e} = \frac{1}{n}\sum_{i=1}^{n} e_i$. The lower the standard deviation of the absolute error is, the better the model captures the data variance. The R-squared score, R^2, is given

by,

$$R^2(y,\hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.19}$$

where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. The higher the R-squared score is, the better the model captures the data variance. In Table 4.1, these two performance metrics are used to quantitatively determine the accuracy of relative-time-kernel-based, instantaneous-kernel-based and rate-based regressions. The percentage improvement of relative-time kernel over instantaneous kernel is highlighted in yellow, and the average magnitudes of percentage improvement for performance metrics, σ_e and R^2, are 14.3% and 16.0%, respectively. It can be observed that the standard deviation of the absolute error of relative-time-kernel-based regression is smaller than that of instantaneous-kernel-based regression except for the force component, F_z. Furthermore, the R-squared scores of relative-time-kernel-based regression for the predictions of F_x, F_y, T_x, T_y and T_z are all higher than those of instantaneous-kernel-based regression. The slightly worse performance of the relative-time kernel for the prediction of F_z is possibly due to measurement noise and inaccuracy caused by unstable vertical motions of the flapping insect in the experiment.

Compared to force prediction, our proposed relative-time kernel has a much higher percentage improvement of up to 52.1% over the instantaneous kernel in torque prediction. This significant difference between force and torque predictions is due to the fact that the performance of traditional kernel-based decoders in force prediction is already good enough, but predicting within-wingstroke torque is much more challenging and needs to be improved especially for individual wingstrokes [183]. In our experiments, to visually track a horizontally

moving robotic flower, the moth is responding to a rotating stimulus. The moth behavior we elicit generates large variation in torques, but was not designed to produce large systematic variations in forces. Consequently, the torques especially yaw torque T_z is the most relevant and challenging to predict. Having taken the extra information of relative spike times into account, the relative-time-kernel-based decoder significantly improves the torque prediction compared to the traditional instantaneous-kernel-based and rate-based decoders.

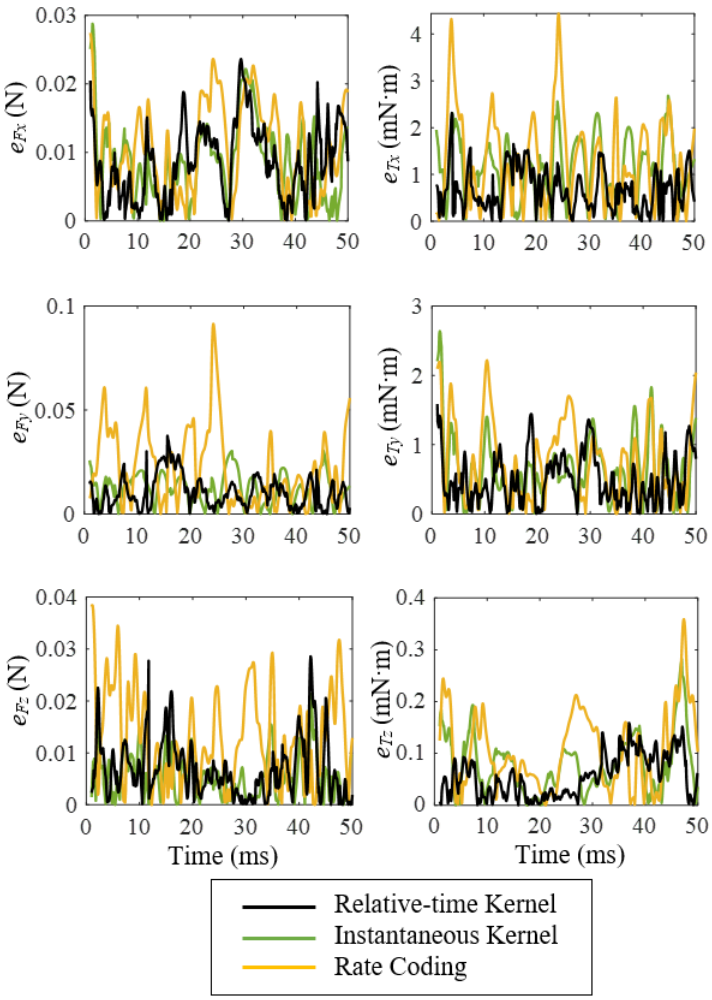


Figure 4.5: Comparison of the absolute prediction errors of the relative-time-kernel-based, instantaneous-kernel-based and rate-based regressions.

Table 4.1: Regression performance comparison.

		F_x	F_y	F_z
$\sigma_e \downarrow$	Relative-time	**0.0054**	**0.0071**	0.0042
	Instantaneous	0.0056	0.0076	**0.0040**
	Rate coding	0.0065	0.0181	0.0087
	% Improvement	3.6%	6.6%	-5.0%
		T_x	T_y	T_z
$\sigma_e \downarrow$	Relative-time	**0.4350**	**0.3542**	**0.0397**
	Instantaneous	0.6193	0.4535	0.0558
	Rate coding	0.9270	0.5506	0.0742
	% Improvement	29.8%	21.9%	28.9%
		F_x	F_y	F_z
$R^2 \uparrow$	Relative-time	**0.6477**	**0.9203**	0.9133
	Instantaneous	0.6369	0.9037	**0.9477**
	Rate coding	0.4690	0.5924	0.7230
	% Improvement	1.7%	1.8%	-3.6%
		T_x	T_y	T_z
$R^2 \uparrow$	Relative-time	**0.8869**	**0.6609**	**0.8555**
	Instantaneous	0.7151	0.4345	0.7133
	Rate coding	0.5481	0.1839	0.4760
	% Improvement	24.0%	52.1%	19.9%

CHAPTER 5

ADAPTIVE SPIKING NEURAL NETWORK CONTROL

5.1 Introduction

Due to their extremely light weight and small size, MAVs especially those with flapping wings have been widely used for assisting humans to access confined spaces inaccessible to vehicles of larger scales and gather information about targets in hazardous environments [109, 166, 82, 136, 24]. Inspired by the intelligent flight strategies of many animal flyers, FWMAVs are much more efficient in generating lift than fixed-wing aircraft and rotorcraft at this micro scale [119, 49, 28]. However, these micro-scale vehicles typically have limited power budgets, and significant variations in physical parameters and dynamic characteristics usually arise from their complex fabrication process. [218, 78]. FWMAVs are extremely sensitive to these manufacturing variations, and therefore may easily become unstable or even damaged [144]. Moreover, the dynamic timescale of flapping flight requires onboard sensors and controllers to operate at high frequency [71, 230]. In spite of all these physical constraints and control design challenges, many flight control algorithms have been developed for FWMAVs in recent years [38, 36, 117, 40, 30, 29, 31, 131, 81]. For instance, a model-predictive high-level controller and a data-driven low-level inverse dynamics controller are combined and implemented for the control of an insect-scale flapping-wing robot to conduct hovering and waypoint tracking maneuvers [40]. An adaptive control scheme consisting of a position feedback controller and a neural-network-based attitude controller is proposed in terms of a hierarchical framework, and allows the controlled flapping-wing vehicle to

accomplish trajectory tracking in the longitudinal plane [81]. However, most of these existing FWMAV control approaches either rely on accurate modeling of the vehicle dynamics or apply to a limited set of maneuvers such as hovering, longitudinal and lateral flight. Therefore, full-envelope flapping flight control design that is robust to unmodeled uncertainties has yet to be developed at this micro scale.

However, for fixed-wing aircraft, people have already developed failure-tolerant flight control systems that can adapt to significant variations in physical parameters, dynamic characteristics and actuator effectiveness, and achieve desired control performance over the full flight envelope in the presence of these uncertainties [186, 142, 116, 6]. The reconfiguration of adaptive control systems can be achieved by either restructuring the control loop or changing the parameters of the control systems [187, 77]. Particularly, artificial neural networks are found to be useful for reconfigurable control design, because they can reflexively change their own connection weights online based on the observed difference between desired and actual system response [187, 186, 52, 56, 58, 113]. For instance, a neural-network-based adaptive controller is trained by the use of a two-phase learning procedure, which contains an offline training phase to approximate a gain-scheduled controller and an online training phase guided by a dual heuristic programming (DHP) adaptive critic, and is demonstrated to be tolerate of significant uncertainties, such as unmodeled dynamics, actuator failures and parameter variations [56]. The light weight, small size and fast dynamics of FWMAVs make them highly sensitive to these uncertainties and therefore easy to go unstable rapidly, so online adaptation and failure tolerance are critical for real-time FWMAV control [38, 34].

Spiking neural networks (SNNs) closely mimic biological brains by communicating via discrete pulses of information only when the neuron's membrane potential reaches the threshold [72, 198, 114]. Therefore, SNN controllers have the potential to be implemented in power-efficient, biologically inspired neuromorphic chips that FWMAVs can be equipped with, and therefore tend to take the place of traditional neural-network-based controllers in modern robot control [36, 38, 89, 90, 60]. Based on offline supervised learning (SL) and online policy gradient reinforcement learning (PGRL) [205, 219, 120, 196, 9, 101, 13], this chapter presents a novel adaptive SNN controller that can learn and adapt to unmodeled uncertainties and actuator failures online. The two-phase adaptive SNN control design contains an offline learning phase, in which SNNs are trained offline by supervised learning to approximate a gain-scheduled proportional-integral-filter (PIF) compensator designed to stabilize the ideal FWMAV dynamic model, and an online learning phase, in which the connection weights of spiking neurons are incrementally updated online in the direction that maximizes the reward function according to the PGRL learning rule. The novelty of this adaptive SNN control design is that the policy gradient online learning rule is biologically plausible and therefore potentially applicable to the next generation of FWMAVs equipped with bio-inspired power-efficient neuromorphic chips [120]. In the online learning phase, only the state error measurement is required by the adaptive control policy, and the spiking neural connection weights will be adjusted accordingly to generate the instantaneous adaptive control signal, which will be then added to the baseline gain-scheduled PIF control signal. Therefore, prior detection and identification of the uncertainty or failure is not needed, which brings computational efficiency and allows the adaptive SNN controller to account for a wide variety of unexpected

circumstances. To benchmark its performance, we compare our novel adaptive SNN control design with a classical non-adaptive SNN controller, which is simply a fixed offline SNN approximation of the gain-scheduled PIF compensator. Both the adaptive and non-adaptive SNN controllers are implemented for the control of a simulated insect-scale flapping-wing robot known as RoboBee [118]. In four different case studies, the RoboBee controlled by the adaptive SNN controller is shown to be able to hover with wing damage, track a moving flower with wing asymmetry, follow a square trajectory with state measurement errors, and conduct a coordinated turn with actuator failure, while the non-adaptive SNN controller fails to accomplish all of these missions.

This chapter is organized as follows. The problem of determining an SNN control law that adapts to unmodeled uncertainties and actuator failures is formulated in Section 5.2. Background knowledge on the PIF compensation and gain scheduling control design is then introduced in Section 5.3. After that, the two-phase adaptive SNN control design is proposed in Section 5.4. In particular, the offline learning phase during which SNNs are trained offline to approximate the gain-scheduled PIF compensator is introduced in Subsection 5.4.1. The online PGRL learning rule to incrementally update the SNN connection weights is described in Subsection 5.4.2. In Section 5.5, the new adaptive SNN control design is implemented for the full-envelope flight control of a simulated flapping-wing robot in the presence of four different uncertainties in four case studies, respectively. The performance of the adaptive SNN controller is demonstrated by comparing with a benchmark non-adaptive SNN controller.

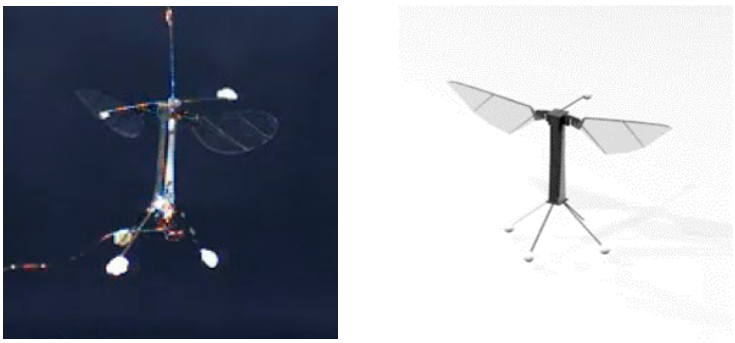
5.2 Problem Formulation

The real-time sensing and control of FWMAVs must operate at a high frequency due to their fast dynamics, and the computation needs to be energy-efficient enough to fit within the tightly limited power budgets [71, 230]. Moreover, their small size and light weight make these vehicles not only highly sensitive to manufacturing variations, but also extremely easy to undergo damage [218, 78, 144]. Therefore, FWMAV control has become a challenging problem nowadays. This chapter focuses on the problem of developing an adaptive full-envelope SNN control law that is robust to significant uncertainties, such as physical parameter variations, unmodeled dynamics and measurement errors, as well as actuator failures. This new adaptive SNN control design is developed for and then validated on an insect-scale flapping-wing robot known as RoboBee (Fig. 5.1) [118], whose high-fidelity dynamic model has been presented and experimentally validated in [35]. In general, FWMAV dynamics can be modeled by a nonlinear parameter-dependent system in standard form,

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t], \quad \mathbf{x}(t_0) = \mathbf{x}_0 \tag{5.1}$$

where $\mathbf{x} \in \mathbb{R}^n$ is the vehicle state vector, $\mathbf{u} \in \mathbb{R}^m$ is the control input, $\mathbf{p} \in \mathbb{R}^l$ is a vector of physical parameters such as the wing shape and drag and lift coefficients, and the initial conditions $\mathbf{x}_0$ are assumed to be known from the vehicle [35, 118, 128, 38, 34]. The control design presented in this chapter can be easily extended to other flapping-wing vehicle dynamic models, provided that they can be approximated by a nonlinear parameter-dependent system in (5.1) as well.

The onboard measurement, $\mathbf{z} \in \mathbb{R}^s$, can be represented as a nonlinear func-



(a) (b)

Figure 5.1: Pictures of (a) the insect-scale flapping-wing robot known as RoboBee and (b) high-fidelity 3D Blender® model of the RoboBee used in simulation.

tion of the vehicle state $\mathbf{x}$ with additive measurement noise,

$$\mathbf{z}(t) = \mathbf{m}[\mathbf{x}(t)] + \mathbf{n}(t) \tag{5.2}$$

where the vector function $\mathbf{m}[\cdot]$ represents the ideal measurement model, and the measurement noise, $\mathbf{n} \in \mathbb{R}^s$, is modeled as a vector of independent and identically distributed zero-mean Gaussian noise. For simplicity, we assume that the vehicle state is fully observable, and all elements of the state can be measured with additive measurement noise, that is, $\mathbf{z} = \hat{\mathbf{x}} = \mathbf{x} + \mathbf{n}$. If this assumption does not stand, an onboard state estimator can be used instead [56, 210, 194]. The vehicle motions are sensed by the system's output vector, $\mathbf{y} \in \mathbb{R}^r$, which can be modeled as a nonlinear function of the vehicle state $\mathbf{x}$ and control input $\mathbf{u}$,

$$\mathbf{y}(t) = \mathbf{h}[\mathbf{x}(t), \mathbf{u}(t)] \tag{5.3}$$

The command input, $\mathbf{y}^* \in \mathbb{R}^r$, is used to specify a desired maneuver such as longitudinal flight and coordinated turn for the vehicle. This desired command value can be provided by external path planning algorithms or human operators [55].

The control law is assumed to be dependent on the onboard state measure-

ment $\hat{\mathbf{x}}$ and desired command value $\mathbf{y}^*$,

$$\mathbf{u}(t) = \mathbf{c}[\hat{\mathbf{x}}(t), \mathbf{y}^*(t)] \tag{5.4}$$

which may contain derivatives or integrals of its arguments, as described in the following section. Given the ideal FWMAV dynamic model in (5.1), onboard state measurement $\hat{\mathbf{x}}$, output vector $\mathbf{y}$ and desired command input $\mathbf{y}^*$, our control design objective is to determine a bounded control history $\mathbf{u}(t)$ that allows the controlled vehicle to accomplish the maneuver specified by the desired command value, assuring that the output tracking error, $\mathbf{e}(t) = \|\mathbf{y}(t) - \mathbf{y}^*(t)\|$, caused by uncertainties or actuator failures is acceptably small and converges to zero as time t approaches infinity. In general, the desired adaptive control law needs to minimize the cost function below consisting of a scalar terminal cost ϕ dependent on the final state $\mathbf{x}(t_f)$ and time t_f, and a scalar integral function of the vehicle state $\mathbf{x}$, control input $\mathbf{u}$, system output $\mathbf{y}$ and time t:

$$J = \phi[\mathbf{x}(t_f), t_f] + \int_{t_0}^{t_f} L[\mathbf{x}(t), \mathbf{u}(t), \mathbf{y}(t), t]dt \tag{5.5}$$

subject to the vehicle dynamics in (5.1). To meet the control objective of following a desired command, we can define an augmented state in the Lagrangian L by including additional error terms that are functions of the state $\mathbf{x}$, control input $\mathbf{u}$ and system output $\mathbf{y}$, as illustrated in the following background section on gain-scheduled PIF compensation.

In this chapter, the performance of this new adaptive SNN control design is demonstrated by considering the following four significant uncertainties that FWMAVs may commonly encounter on the fly:

1) *Parameter variation*: the actual vehicle physical parameters $\mathbf{p}$ are different from those parameters in the ideal vehicle dynamic model due to manufacturing variations or system identification errors.

2) *Unmodeled dynamics*: manufacturing or modeling inaccuracy causes variations in dynamic characteristics, such as wing asymmetry and coupling effects, that have not been considered in the ideal vehicle dynamic model.

3) *Measurement error*: the vehicle state $\mathbf{x}$ is measured onboard with additive measurement noise $\mathbf{n}$.

4) *Actuator failure*: partial loss of effectiveness is experienced by the actuators at time t_a, and the actual control input $\mathbf{u}$ generated by the partially failed actuators is given by,

$$\mathbf{u}(t) = \boldsymbol{\varepsilon}(t) \odot \mathbf{u}^*(t), \ t \geqslant t_a \tag{5.6}$$

where $\boldsymbol{\varepsilon} \in \mathbb{R}^m$ is a vector of effective factors, the symbol $\odot$ denotes component-wise multiplication, $\mathbf{u}^* \in \mathbb{R}^m$ is the desired control input, and t_a is the failure time [87, 199].

This chapter presents a new adaptive full-envelope SNN control design comprised of offline and online learning phases that integrate classical multivariable control knowledge with supervised and reinforcement learning. In the offline learning phase, a gain-scheduled controller is developed to stabilize the ideal FWMAV dynamic model in (5.1) using PIF compensation, as reviewed in the next section. Feed-forward SNNs are then trained offline by supervised learning to approximate this classical gain-scheduled PIF compensator in Subsection 5.4.1. In the online learning phase, spiking neuron populations are trained by reinforcement learning to minimize the system's output error and adapt to unmodeled uncertainties, as introduced in Subsection 5.4.2. This new adaptive SNN controller is then implemented for the full-envelope control of the RoboBee in the presence of the four aforementioned uncertainties, and its performance is compared with a benchmark non-adaptive SNN controller in Sec-

tion 5.5.

5.3 Background on Gain-Scheduled PIF Compensation

Classical feedback control theory has provided many useful insights into the inherent flight control strategies of natural flapping-wing flyers such as fruit flies [43, 112] and moths [39, 184]. For example, the yaw and speed control models of *Drosophila* have been found to conform to a classical proportional-integral (PI) feedback control architecture [43]. Inspired by these experimental studies on animal flyers, we choose to use a classical gain-scheduled PIF compensator to stabilize the ideal FWMAV dynamic model. Gain scheduling is a common design approach to control nonlinear dynamic systems by interpolating a family of locally optimal linear control designs, such as linear quadratic regulator (LQR) and PIF compensator, throughout the entire operating regions of the nonlinear systems [174]. Moreover, PIF compensation not only provides good noise rejection at high frequency, but also tolerates parameter variations to some extent [189]. Therefore, the gain-scheduled PIF compensator is a great initialization point for the subsequent online adaptation phase [52].

As a first step in the gain scheduling design, a family of equilibrium points can be obtained by letting the state derivatives be zero in (5.1),

$$\mathbf{0} = \mathbf{f}[\mathbf{x}^*(\mathbf{a}), \mathbf{u}^*(\mathbf{a}), \mathbf{p}(\mathbf{a})] \tag{5.7}$$

where $\mathbf{a}$ denotes a vector of dynamically significant scheduling parameters, and the arguments of the vector function $\mathbf{f}$ are assumed to be dependent on these scheduling parameters. As shown in the block diagram in Fig. 5.2, the scheduling variables $\mathbf{a}$ are provided by a scheduling variable generator (SVG)

containing algebraic functions that take the desired command input $\mathbf{y}^*$ as input. More details on the computation of quasi-steady set points for FWMAVs can be found in [35]. These equilibrium points obtained in (5.7) specify a set of operating conditions spanning the full FWMAV flight envelope, and are indexed by $\kappa = 1, 2, \cdots, M$. Linearizing the nonlinear dynamics around these operating points and keeping $\mathbf{a}$ fixed for each operating point, we can obtain a set of linearized models,

$$\dot{\tilde{\mathbf{x}}}(t) = \mathbf{A}\tilde{\mathbf{x}}(t) + \mathbf{B}\tilde{\mathbf{u}}(t) \tag{5.8}$$

where $\tilde{\mathbf{x}}(t) = \mathbf{x}(t) - \mathbf{x}^* \in \mathbb{R}^n$ is the state deviation, $\tilde{\mathbf{u}}(t) = \mathbf{u}(t) - \mathbf{u}^* \in \mathbb{R}^m$ is the control deviation, and the state space matrices are given by,

$$\mathbf{A} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}^*,\mathbf{u}=\mathbf{u}^*}, \ \mathbf{B} = \left.\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right|_{\mathbf{x}=\mathbf{x}^*,\mathbf{u}=\mathbf{u}^*} \tag{5.9}$$

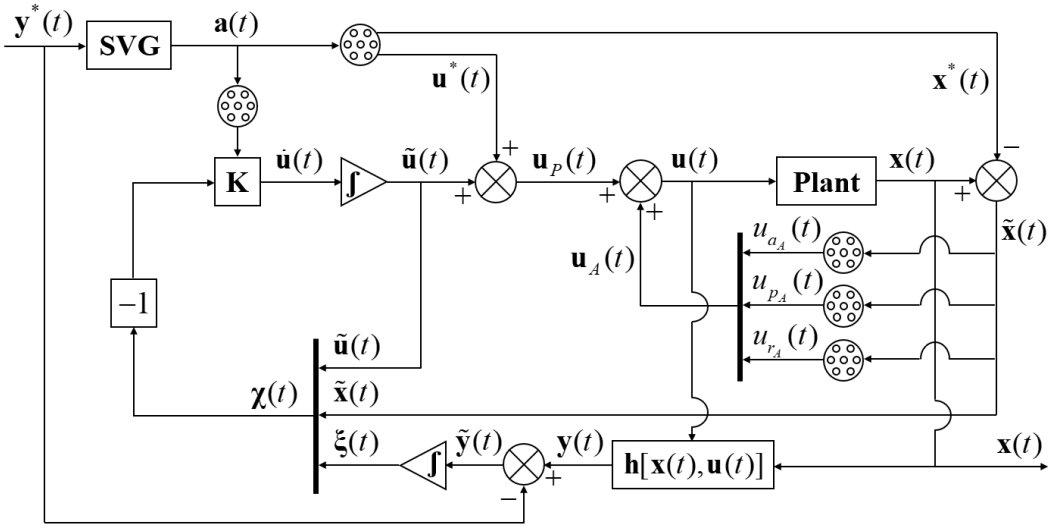


Figure 5.2: Adaptive SNN control architecture containing offline trained SNNs that approximates a gain-scheduled PIF compensator and online adaptive SNNs that learn to account for uncertainties, where five clusters of circles represent separate populations of spiking neurons.

After obtaining a family of linearized models, the PIF compensation approach augments the vehicle state deviation $\tilde{\mathbf{x}}$ with the control deviation $\tilde{\mathbf{u}}$ and

the integral of the output error, $\boldsymbol{\xi}(t) = \boldsymbol{\xi}(0) + \int_0^t [\mathbf{y}(\tau) - \mathbf{y}^*] d\tau \in \mathbb{R}^r$, which is designed to reduce the steady-state error, and therefore uses the augmented state vector, $\boldsymbol{\chi} = [\tilde{\mathbf{x}}^T \ \tilde{\mathbf{u}}^T \ \boldsymbol{\xi}^T]^T \in \mathbb{R}^{n+m+r}$ [189]. The PIF feedback control gains are chosen to minimize the quadratic cost function,

$$J = \lim_{t_f \to \infty} \frac{1}{2t_f} \int_{t_0}^{t_f} [\boldsymbol{\chi}^T(t) \mathbf{Q} \boldsymbol{\chi}(t) + \dot{\mathbf{u}}^T(t) \mathbf{R} \dot{\mathbf{u}}(t)] dt \tag{5.10}$$

where $\mathbf{Q}$ and $\mathbf{R}$ are weighing matrices for the augmented state and control derivative, respectively. This cost function is subject to the augmented state equation, which can be written as,

$$\dot{\boldsymbol{\chi}}(t) = \mathbf{A}'(t) \boldsymbol{\chi}(t) + \mathbf{B}'(t) \dot{\mathbf{u}}(t) \tag{5.11}$$

where the state-space matrices are given by,

$$\mathbf{A}'(t) = \begin{bmatrix} \mathbf{A}(t) & \mathbf{B}(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{H}_x(t) & \mathbf{H}_u(t) & \mathbf{0} \end{bmatrix}, \ \mathbf{B}'(t) = \begin{bmatrix} \mathbf{0} \\ \mathbf{I}_m \\ \mathbf{0} \end{bmatrix} \tag{5.12}$$

and the matrices, $\mathbf{H}_x$ and $\mathbf{H}_u$, relate the state $\mathbf{x}$ and the control $\mathbf{u}$ to the system's output. It is computationally expensive and difficult to obtain the solution to the periodic Riccati equation based on the time-varying dynamic model in (5.11). Therefore, we choose to average the time-varying linear dynamics over a flapping period T and ignore the wing motions based on the so-called stroke-averaging technique [34, 161, 206]. Based on the time-varying linearized model in (5.11), the stroke-averaged dynamic model can be obtained as,

$$\dot{\boldsymbol{\chi}}(t) = \bar{\mathbf{A}}' \boldsymbol{\chi}(t) + \bar{\mathbf{B}}' \dot{\mathbf{u}}(t) \tag{5.13}$$

where the stroke-averaged state-space matrices are given by,

$$\bar{\mathbf{A}}' = \frac{1}{T} \int_0^T \mathbf{A}'(t) dt, \ \bar{\mathbf{B}}' = \frac{1}{T} \int_0^T \mathbf{B}'(t) dt \tag{5.14}$$

Our numerical simulation results show that the settling time of the RoboBee's wing stroke angle for a step reference input is more than five times greater than an entire flapping period, so the stroke-averaged dynamic model in (5.13) will obtain good enough performance compared to the actual periodically time-varying dynamic model in (5.11).

The optimal linear control law minimizing the quadratic cost function in (5.10) with a steady-state feedback gain matrix, $\mathbf{K} \in \mathbb{R}^{m\times(n+m+r)}$, as t_f approaches infinity is

$$\dot{\mathbf{u}}(t) = -\mathbf{K}\boldsymbol{\chi}(t) = -\mathbf{R}^{-1}\bar{\mathbf{B}}'^{T}\mathbf{S}(0)\boldsymbol{\chi}(t) \tag{5.15}$$

where $\mathbf{S}(0)$ is a solution to the algebraic Riccati equation of the form,

$$\mathbf{S}(0)\bar{\mathbf{B}}'\mathbf{R}^{-1}\bar{\mathbf{B}}'^{T}\mathbf{S}(0) - \bar{\mathbf{A}}'^{T}\mathbf{S}(0) - \mathbf{S}(0)\bar{\mathbf{A}}' - \mathbf{Q} = \mathbf{0} \tag{5.16}$$

We will then integrate the derivative of control input $\dot{\mathbf{u}}$ obtained in (5.15) before transmitting it to the plant. Assuming that the command vector is nonsingular, the optimal PIF control law above can be rearranged as,

$$\begin{aligned}\dot{\mathbf{u}}(t) &= -\mathbf{K}_1[\mathbf{x}(t) - \mathbf{x}^*] - \mathbf{K}_2[\mathbf{u}(t) - \mathbf{u}^*] - \mathbf{K}_3\left\{\boldsymbol{\xi}(0) + \int_0^t [\mathbf{y}(\tau) - \mathbf{y}^*]d\tau\right\} \\ &= -\mathbf{K}_1\mathbf{x}(t) - \mathbf{K}_2\mathbf{u}(t) - \mathbf{K}_3\boldsymbol{\xi}(t) - \mathbf{K}_4\mathbf{y}^*\end{aligned} \tag{5.17}$$

where $\mathbf{K}_1 \in \mathbb{R}^{m\times n}$, $\mathbf{K}_2 \in \mathbb{R}^{m\times m}$ and $\mathbf{K}_3 \in \mathbb{R}^{m\times r}$ are all partitioned matrices obtained from the full feedback gain matrix, and $\mathbf{K}_4 \in \mathbb{R}^{m\times r}$ can be obtained from the linearized output equation [189]. Performing Laplace transform on both sides of the equation above and assuming zero initial conditions, we can obtain the transfer function between the command vector and the control input,

$$\frac{\mathbf{U}(s)}{\mathbf{Y}^*(s)} = \frac{1}{s}(s\mathbf{I}_m + \mathbf{K}_2)^{-1}(s\mathbf{K}_4 + \mathbf{K}_3) \tag{5.18}$$

which consists of an integrator, a low-pass filter and a lead compensator. This correspondingly provides the closed-loop system with excellent command

tracking at low frequency, noise rejection at high frequency, and loop shaping around the cross-over frequency.

Using the design approach described above, linear PIF control laws can be developed for a family of stroke-averaged FWMAV dynamic models, $\{\bar{\mathbf{A}}', \bar{\mathbf{B}}'\}_{\kappa=1,2,\cdots,M}$, linearized at a set of operating conditions spanning the full flight envelope. Locally optimal feedback gain matrices and Riccati matrices, $\{\mathbf{K}, \mathbf{S}\}_{\kappa=1,2,\cdots,M}$, can be correspondingly obtained for all operating conditions indexed by κ. In the following section, feedforward SNNs will be trained offline by supervised learning to interpolate this family of locally optimal linear control designs, and obtain a global baseline controller that can be used to stabilize the ideal nonlinear FWMAV dynamic model for full-envelope flight, and serve as a good initialization point for the subsequent online adaptation phase.

5.4 Adaptive SNN Control Design

The onboard sensing and control loops of FWMAVs must operate not only at high frequency due to their fast dynamics, but also efficiently enough to fit within their tightly limited power budgets [71, 230, 36, 38]. Inspired by biological brains, SNNs communicate using discrete pulses of information and can be potentially implemented in energy-efficient neuromorphic processors and sensors, which presents a promising solution to the challenging FWMAV control problem [34, 72, 198, 114]. In this chapter, an adaptive SNN control framework is developed using a two-phase design approach. Ultimately, the control input $\mathbf{u}$ transmitted to the vehicle is a combination of an offline SNN approximation of the gain-scheduled PIF control signal $\hat{\mathbf{u}}_P$ as described in previous section and

an additional online adaptive term $\mathbf{u}_A$,

$$\mathbf{u}(t) = \hat{\mathbf{u}}_P(t) + \mathbf{u}_A(t) \tag{5.19}$$

During the offline learning phase, the SNN is trained by supervised learning to interpolate a family of locally optimal PIF compensators designed to correspondingly stabilize the ideal dynamic model linearized at a number of set points over the full flight envelope. During the online learning phase, the additional online adaptive term is learned online by policy gradient reinforcement learning framework to minimize the output error caused by uncertainties that have not been considered in the ideal dynamic modeling and offline learning phase. The offline and online learning phases will be described in the following two subsections, respectively.

5.4.1 Offline Learning Phase

As a locally optimal control law guaranteed to stabilize the linearized FWMAV dynamic model at all the achievable set points over the full flight envelope, the gain-scheduled PIF compensator introduced in Section 5.3 can provide a good initialization for the subsequent SNN online adaptation. As the first step in the offline learning phase, feedforward SNNs are trained offline by supervised learning to interpolate the family of locally optimal PIF control solutions, $\{\mathbf{K}, \mathbf{S}\}_{\kappa=1,2,\cdots,M}$, and serve as a global controller to stabilize the ideal FWMAV dynamic model over the full flight envelope. As shown in the control architecture in Fig. 5.2, one population of spiking neurons is trained to approximate the steady-state PIF gain matrix as a function of scheduling variables, and a separate population of spiking neurons is trained to interpolate the feedforward set points, $\{\mathbf{x}^*, \mathbf{u}^*\}_{\kappa=1,2,\cdots,M}$, as a function of scheduling variables.

In this chapter, the spiking neuron model conforms to the Neural Engineering Framework (NEF), in which time-varying input signals are encoded in populations of spiking neurons and continuous-time output signals are then decoded from the output current of these spiking neurons [48, 190]. The input current J_i into the spiking neuron i is dependent on the scheduling variables $\mathbf{a}$,

$$J_i(\mathbf{a}) = \eta_i \mathbf{v}_i^T \mathbf{a}(t) + J_{b_i} \tag{5.20}$$

where η_i is a constant gain factor, $\mathbf{v}_i$ is the encoding weight vector, and J_{b_i} is a fixed bias current of neuron i. Based on the leaky integrate-and-fire (LIF) neuron model [70], the voltage V_i across the membrane of neuron i is modeled as,

$$\dot{V}_i(t) = -\frac{1}{\tau_d}[V_i(t) - R_i J_i(\mathbf{a})] \tag{5.21}$$

where τ_d is the decaying time constant, and R_i is the passive membrane resistance of neuron i. If the voltage across the membrane exceeds a threshold, a spike will be emitted by the neuron. After that, the membrane voltage will be reset to the resting value for a while, and then follow the LIF model in (5.21) again. Therefore, the spike response of neuron i to the input current J_i can be modeled as a nonlinear function G_i, and the resulting spike train r_i can be represented as a sum of Dirac delta functions,

$$r_i(t) = G_i[J_i(\mathbf{a})] = \sum_{k_i} \delta_i(t - t_{k_i}^i) \tag{5.22}$$

where k_i denotes the spike index. The sum of Dirac delta functions can then be converted to a continuous function via convolution with a filter h_i,

$$s_i(\mathbf{a}) = G_i[J_i(\mathbf{a})] * h_i(t) = \sum_{k_i} h_i(t - t_{k_i}^i) \tag{5.23}$$

where the filter $h_i(t) = \frac{1}{\tau_p} e^{-\frac{t}{\tau_p}}$ is an exponential decaying function with a post-synaptic time constant τ_p. The post-synaptic current from a population of N

spiking neurons can then be decoded into a continuous-time signal $g(t)$ by,

$$g(t) = \sum_{i=1}^{N} s_i[\mathbf{a}(t)]w_i = \mathbf{w}^T\mathbf{s}[\mathbf{a}(t)] \tag{5.24}$$

where $\mathbf{s} = [s_1 \ s_2 \ \cdots \ s_N]^T$ is the post-synaptic current, and $\mathbf{w} = [w_1 \ w_2 \ \cdots \ w_N]^T$ is the output decoding weights that can be trimmed by supervised learning to approximate time-varying signals, such as the steady state PIF matrices and feedforward set points, as a function of scheduling variables.

As reviewed in Section 5.3, the locally optimal PIF control law is found to be proportional to the augmented state vector $\boldsymbol{\chi}$, and can be written as,

$$\dot{\mathbf{u}}(t) = -\mathbf{K}(\mathbf{a})\boldsymbol{\chi}(t) = -\mathbf{K}_1(\mathbf{a})\tilde{\mathbf{x}}(t) - \mathbf{K}_2(\mathbf{a})\tilde{\mathbf{u}}(t) - \mathbf{K}_3(\mathbf{a})\boldsymbol{\xi}(t) \tag{5.25}$$

where the gain matrices, $\mathbf{K}$, $\mathbf{K}_1$, $\mathbf{K}_2$ and $\mathbf{K}_3$, are all calculated by feedforward SNNs to vary as a function of scheduling parameters $\mathbf{a}$. One of the main reasons for introducing the integral term, $\mathbf{K}_3\boldsymbol{\xi}$, in the PIF control law above is to reduce the steady-state error caused by modeling uncertainties. Since an additional PGRL-based adaptive term will later be included in the SNN control design to learn and compensate for the same type of uncertainties online, we neglect this integral term while training the SNN offline to approximate the steady-state PIF gain matrix. Performing the Laplace transformation on (5.25), the transfer function for the PIF compensator can be expressed as a low-pass filtered gain matrix,

$$\mathbf{G}(s) \triangleq \frac{\mathbf{U}(s)}{\mathbf{X}(s)} = -[s\mathbf{I}_m + \mathbf{K}_2(\mathbf{a})]^{-1}\mathbf{K}_1(\mathbf{a}) \tag{5.26}$$

Based on the final value theorem, the approximate steady-state PIF gain matrix can be given by,

$$\mathbf{G}(0) = -\mathbf{K}_2^{-1}(\mathbf{a})\mathbf{K}_1(\mathbf{a}) \triangleq -\hat{\mathbf{K}}(\mathbf{a}) \tag{5.27}$$

Considering the scheduling variables $\mathbf{a}$ provided by the SVG as the input

to the network and the steady-state PIF gain matrix, $\hat{\mathbf{K}} \in \mathbb{R}^{m \times n}$, as the output, we use a population of N_1 spiking neurons to interpolate the family of PIF gain matrices. The network output weighing vector, $\mathbf{w}_{ij}^* \in \mathbb{R}^{N_1}$, that gives the optimal approximation of each element of the steady-state gain matrix, $\hat{K}_{ij} \in \mathbb{R}$, can be computed by standard least squares optimization over M set points sampled throughout the entire flight envelope,

$$\mathbf{w}_{ij}^* = \underset{\mathbf{w}_{ij}}{\operatorname{argmin}} \sum_{\kappa=1}^{M} [\hat{K}_{ij}(\mathbf{a}_\kappa) - \mathbf{w}_{ij}^T \mathbf{s}_1(\mathbf{a}_\kappa)]^2, \; i = 1, 2, \cdots, m, \; j = 1, 2, \cdots, n \tag{5.28}$$

where $\mathbf{a}_\kappa$ is the scheduling variables corresponding with the κ-th sampled set point, and $\mathbf{s}_1 \in \mathbb{R}^{N_1}$ is the post-synaptic current from this population of N_1 spiking neurons. Then, the SNN approximation of each element of the steady-state PIF gain matrix is a linear transformation of the post-synaptic current $\mathbf{s}_1$,

$$\hat{\mathbf{K}}(\mathbf{a}) = \begin{bmatrix} \mathbf{w}_{11}^{*T}\mathbf{s}_1(\mathbf{a}) & \mathbf{w}_{12}^{*T}\mathbf{s}_1(\mathbf{a}) & \cdots & \mathbf{w}_{1n}^{*T}\mathbf{s}_1(\mathbf{a}) \\ \mathbf{w}_{21}^{*T}\mathbf{s}_1(\mathbf{a}) & \mathbf{w}_{22}^{*T}\mathbf{s}_1(\mathbf{a}) & \cdots & \mathbf{w}_{2n}^{*T}\mathbf{s}_1(\mathbf{a}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{w}_{m1}^{*T}\mathbf{s}_1(\mathbf{a}) & \mathbf{w}_{m2}^{*T}\mathbf{s}_1(\mathbf{a}) & \cdots & \mathbf{w}_{mn}^{*T}\mathbf{s}_1(\mathbf{a}) \end{bmatrix} \tag{5.29}$$

In a separate population of N_2 spiking neurons, we consider the scheduling variables as the input to the network and the feedforward set-point vector, $\hat{\mathbf{v}} = [\mathbf{x}^{*T} \; \mathbf{u}^{*T}]^T \in \mathbb{R}^{m+n}$, as the output. Similarly, the network output weighing matrix, $\mathbf{W}_P^* \in \mathbb{R}^{(m+n) \times N_2}$, that gives the optimal interpolation of the family of set points, $\{\mathbf{x}^*, \mathbf{u}^*\}_{\kappa=1,2,\cdots,M}$, can be computed by standard least squares optimization over the entire flight envelope,

$$\mathbf{W}_P^* = \underset{\mathbf{W}}{\operatorname{argmin}} \sum_{\kappa=1}^{M} \|\hat{\mathbf{v}}(\mathbf{a}_\kappa) - \mathbf{W}\mathbf{s}_2(\mathbf{a}_\kappa)\|^2 \tag{5.30}$$

where $\mathbf{s}_2 \in \mathbb{R}^{N_2}$ is the post-synaptic current from this population of N_2 spiking neurons. Then, the SNN approximation of the feedforward set points is a linear

transformation of the post-synaptic current $\mathbf{s}_2$,

$$\begin{bmatrix} \hat{\mathbf{x}}^*(\mathbf{a}) \\ \hat{\mathbf{u}}^*(\mathbf{a}) \end{bmatrix} = \mathbf{W}_P^* \mathbf{s}_2(\mathbf{a}) \tag{5.31}$$

The steady-state PIF gains (5.29) and set points (5.31) approximated by single-layer feedforward SNNs can then be used to compute the PIF control input $\hat{\mathbf{u}}_P$ as a baseline for the subsequent online learning phase,

$$\hat{\mathbf{u}}_P(t) = -\hat{\mathbf{K}}[\mathbf{a}(t)]\hat{\mathbf{x}}^*[\mathbf{a}(t)] + \hat{\mathbf{u}}^*[\mathbf{a}(t)] \tag{5.32}$$

5.4.2 Online Learning Phase

The classical gain-scheduled PIF compensator design introduced in Section 5.3 is developed using a family of ideal linearized vehicle dynamic models, which does not take into account significant uncertainties such as physical parameter variations, unmodeled dynamics, measurement errors and actuator failures. These unmodeled uncertainties that FWMAVs commonly encounter on the fly will immediately destabilize the vehicle in roll, pitch and yaw [38, 34, 40]. A good adaptive controller for full-envelope flapping flight should be capable of learning to compensate for these unmodeled uncertainties in real time. Therefore, apart from the SL-based SNN approximation of the baseline gain-scheduled PIF control law described in the previous subsection, an additional adaptive control input will be learned online by policy gradient reinforcement learning (PGRL) method to stabilize the altitude and orientation of the vehicle in the presence of unmodeled uncertainties in the online learning phase.

Given that the SNN connection weights will be adjusted incrementally by reinforcement learning at every time step, we first discretize the plant dynamics

in (5.1), and consider a discrete nonlinear parameter-dependent system with time increments equally spaced in the time interval $t_0 \leqslant t_k \leqslant t_f$, such that

$$\mathbf{x}(t_{k+1}) = \mathbf{g}[\mathbf{x}(t_k), \mathbf{u}(t_k), \mathbf{p}] \tag{5.33}$$

with the initial conditions $\mathbf{x}_0$ assumed to be known. For brevity, the abbreviated form of the state and control input, $\mathbf{x}_k \equiv \mathbf{x}(t_k)$ and $\mathbf{u}_k \equiv \mathbf{u}(t_k)$, will be used from here on. The discrete dynamic system can be modeled as a Markov decision process (MDP), in which the state, action and immediate reward of the learning agent at time t_k are given by $\mathbf{x}_k$, $\mathbf{u}_k$ and r_k, respectively [96]. The control policy $\boldsymbol{\pi}$ is approximated by a population of spiking neurons with output connection weights $\mathbf{w}_k$ at time t_k,

$$\mathbf{u}_k = \boldsymbol{\pi}(\mathbf{x}_k, \mathbf{w}_k) \tag{5.34}$$

which is assumed to be differentiable with respect to the connection weights. The performance of the control policy $\boldsymbol{\pi}$ is denoted by the long-term average reward per step $\rho(\boldsymbol{\pi})$ that the learning agent aims to maximize,

$$\rho(\boldsymbol{\pi}) = \sum_{\mathbf{x}} \lim_{k\to\infty} P(\mathbf{x}_k = \mathbf{x}|\mathbf{x}_0, \boldsymbol{\pi}) \sum_{\mathbf{u}} P(\mathbf{u}_k = \mathbf{u}|\mathbf{x}_k = \mathbf{x}, \mathbf{w}_k) E(r_{k+1}|\mathbf{x}_k = \mathbf{x}, \mathbf{u}_k = \mathbf{u}) \tag{5.35}$$

where $P(\cdot)$ and $E(\cdot)$ denote the probability and expectation, respectively [196].

As shown in the adaptive SNN control architecture in Fig. 5.2, the state error, $\tilde{\mathbf{x}}_k = \mathbf{x}_k - \mathbf{x}_k^*$, at time t_k is influenced by the closed-loop system dynamics and therefore dependent on the choice of SNN connection weights $\mathbf{w}_k$. To conduct a maneuver specified by the desired command input, the online learning rule of the adaptive SNN controller needs to change the connection weights in a direction that eliminates the error function e_k representing the difference between the actual and desired vehicle states,

$$e_k(\mathbf{w}_k) = \boldsymbol{\lambda}^T[\tilde{\mathbf{x}}_k(\mathbf{w}_k) + \alpha\dot{\tilde{\mathbf{x}}}_k(\mathbf{w}_k)] \tag{5.36}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^n$ is a user-defined weighing vector that specifies a desired trade-off between state variables, and $\alpha \in \mathbb{R}$ is a constant scalar that weighs the state error against its derivative. According to the characteristics of flapping flight dynamics, it is reasonable to assume that $x-$, $y-$ and $z-$components of the vehicle's body velocity and their derivatives weigh more than other state variables in the error function above for adaptive pitch, roll and amplitude control, respectively. In addition, since FWMAV dynamics are neutrally stable in yaw, we do not consider adaptive yaw control in this chapter for simplicity.

Since the SNN connection weights are typically adjusted online at a learning rate much higher than the controller's processing rate, the error function e_k can serve as a measure for the performance of the adaptive control policy at time t_k with larger errors mapping to smaller long-term rewards,

$$\hat{\rho}[\boldsymbol{\pi}(\mathbf{x}_k, \mathbf{w}_k)] = -\frac{1}{2} e_k^2(\mathbf{w}_k) = -\frac{1}{2} \left\{ \boldsymbol{\lambda}^T [\tilde{\mathbf{x}}_k(\mathbf{w}_k) + \alpha \dot{\tilde{\mathbf{x}}}_k(\mathbf{w}_k)] \right\}^2 \tag{5.37}$$

Then, we consider the typical reinforcement learning problem of finding the optimal connection weights $\mathbf{w}^*$ that maximize the reward function, such that

$$\mathbf{w}^* = \underset{\mathbf{w}_k}{\operatorname{argmax}}\ \hat{\rho}[\boldsymbol{\pi}(\mathbf{x}_k, \mathbf{w}_k)] = \underset{\mathbf{w}_k}{\operatorname{argmin}}\ \left\{ \boldsymbol{\lambda}^T [\tilde{\mathbf{x}}_k(\mathbf{w}_k) + \alpha \dot{\tilde{\mathbf{x}}}_k(\mathbf{w}_k)] \right\}^2 \tag{5.38}$$

Based on the policy gradient reinforcement learning (PGRL) method, the adaptive learning rule to update the SNN output connection weights at time t_k while moving towards the maximum of the reward function takes the form,

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \gamma \left(\frac{\partial \hat{\rho}}{\partial \mathbf{w}_k} \right)^T \tag{5.39}$$

where $\gamma \in \mathbb{R}$ is a user-defined scalar learning rate [196, 9, 101, 13]. By differentiating the quadratic error function in (5.37) with respect to the connection weights and substituting in the spiking neuron model, the gradient of the reward func-

tion with respect to the connection weights can be obtained as follows,

$$\frac{\partial \hat{\rho}}{\partial \mathbf{w}_k} = -e_k(\mathbf{w}_k)\mathbf{s}_k^T \tag{5.40}$$

where $\mathbf{s}_k$ is the post-synaptic current from the spiking neuron population at time t_k. This leads to the PGRL-based adaptive SNN learning rule to adjust the connection weights over time,

$$\mathbf{w}_{k+1} = \mathbf{w}_k - \gamma e_k(\mathbf{w}_k)\mathbf{s}_k \tag{5.41}$$

which is guaranteed to converge to the optimal connection weights $\mathbf{w}^*$ [175, 196, 213, 101, 14]. Every element of the adaptive term, $\mathbf{u}_A \in \mathbb{R}^m$, is separately computed by a population of N_3 spiking neurons, so a set of m optimal weighing vectors can be obtained and arranged into a weighing matrix, $\mathbf{W}_A^* \in \mathbb{R}^{m \times N_3}$. Then, the adaptive term is a linear transformation of the post-synaptic current $\mathbf{s}$,

$$\mathbf{u}_A(t) = \mathbf{W}_A^* \mathbf{s}(t) \tag{5.42}$$

When this PGRL-based adaptive SNN learning rule is applied in real time, the learning rate at which the output connection weights are adjusted should be at least ten times as high as the processing rate at which the controller computes and transmits a new control signal in simulation. To demonstrate the online learning performance of the adaptive SNN controller, we plot the mean-squared error signal versus the number of epochs when the flapping-wing vehicle is commanded to hover with an initial disturbance in Fig. 5.3 below as an example. Here, the learning rate of the adaptive SNN is 10 kHz, and the processing rate of the controller is 1 kHz. Both of them are much higher than the 120-Hz flapping frequency of the vehicle, which is another solid support for our choice of using stroke-averaged dynamic model to obtain the optimal offline PIF control law as described in Section 5.3. We can observe from the SNN update over 500

epochs below that the mean-squared error increases slightly at first, and is then significantly reduced by half within only 300 epochs. This error history shows that the adaptive SNN controller quickly adapts to the initial disturbance with its connection weights converging to optimal values.

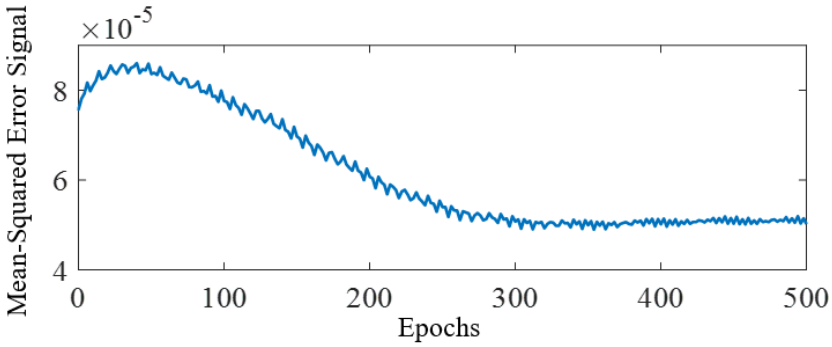


Figure 5.3: The mean-squared error signal versus the number of epochs when the vehicle is commanded to hover with an initial disturbance.

5.5 Flight Control Simulation and Results

The adaptive two-phase SNN controller is implemented for the full-envelope control of a simulated flapping-wing vehicle with eight degrees of freedom known as RoboBee (Fig. 5.1) [117, 35]. However, it can be easily extended to other FWMAV dynamic models, provided that they can be approximated by the nonlinear parameter-dependent system described in (5.1) as well. The full state of the RoboBee can be expressed as a combination of its wing state and body state:

$$\mathbf{x} = [\phi_r \;\; \psi_r \;\; \phi_l \;\; \psi_l \;\; \dot{\phi}_r \;\; \dot{\psi}_r \;\; \dot{\phi}_l \;\; \dot{\psi}_l \;\; x \;\; y \;\; z \;\; \phi \;\; \theta \;\; \psi \;\; u \;\; v \;\; w \;\; p \;\; q \;\; r]^T \tag{5.43}$$

where ϕ_r and ϕ_l are the stroke angles of the right wing and left wing, ψ_r and ψ_l are the passively controlled pitch angles of the right wing and left wing, x, y and

z represent the vehicle position, Euler angles, namely yaw ϕ, roll θ and pitch ψ, represent the body orientation, u, v and w represent the vehicle's body velocity, and p, q and r represent the vehicle's angular velocity in the body frame. The right-wing and left-wing stroke angles, ϕ_r and ϕ_l, are entirely determined by the control input $\mathbf{u}$,

$$\mathbf{u} = [u_a \;\; u_r \;\; u_p \;\; u_y]^T \tag{5.44}$$

where u_a, u_r, u_p and u_y represent amplitude, roll, pitch and yaw control input, respectively. The command input $\mathbf{y}^*$ is used to specify a desired maneuver for the RoboBee, and is given by

$$\mathbf{y}^* = [V \;\; \gamma \;\; \dot{\xi} \;\; \beta]^T \tag{5.45}$$

where V denotes the vehicle velocity, γ denotes the climb angle, $\dot{\xi}$ denotes the turn rate, and β denotes the sideslip angle. More details on the derivation and validation of the vehicle's flight and actuator dynamics can be found in [35].

In numerical simulations, the RoboBee is commanded to conduct four different maneuvers over the full flight envelope in the presence of different uncertainties. The four classes of uncertainties described in Section 5.2 are simulated as follows:

1) *Parameter variation*: To simulate the damage of the left wing, we decrease the aerodynamic forces and moments generated by the damaged wing by 20%, and the wing mass by 10% with respect to the ideal model in case study 1.

2) *Unmodeled dynamics*: Static random bias of the stroke amplitude and mean stroke angle of both wings is introduced to reflect the asymmetry between the right-wing and left-wing aerodynamic characteristics in case study 2.

3) *Measurement error*: Zero-mean random noise is directly added to the on-

board measurement of the vehicle's roll, pitch and yaw angles and their rates in case study 3.

4) *Actuator failure*: Partial loss of effectiveness of the actuator is introduced by reducing the left-wing stroke angle amplitude by 10% with respect to the ideal model in case study 4.

The capability of the adaptive SNN controller to account for these significant uncertainties is demonstrated by comparing to a benchmark non-adaptive SNN controller that is trained offline to simply approximate a gain-scheduled PIF compensator. The flight control simulation results of the adaptive and non-adaptive SNN controllers will be compared in the following four different case studies.

5.5.1 Case Study 1: Hovering with Wing Damage

Due to their small size and fragile material, the flapping wings of FWMAVs can be easily damaged by accidental collision with obstacles or external disturbances such as strong wind gusts [25, 34]. In this case study, the ability of the adaptive SNN to learn and adapt to wing damage is demonstrated. To replicate the unforeseen situation in which the RoboBee's wing tip is damaged, we decrease the aerodynamic forces and moments generated by the damaged left wing by 20%, and the mass of the left wing by 10% with respect to the ideal dynamic model used for offline SNN training in simulation. Removing 10% of the mass from the tip of the wing will correspondingly reduce the aerodynamic forces and moments by approximately 20%. The percentage decrease in the aerodynamic forces and moments is greater than that in the mass of the

wing, because the wing tip is the fastest moving part while the wing flaps, thus contributing most to the aerodynamic force and moment generation [51, 99]. Then, we demonstrate the adaptive SNN controller's ability to learn and adapt to these unexpected variations caused by wing damage when the vehicle is commanded to hover with an initial disturbance.

The vehicle trajectories of the adaptive and non-adaptive SNN controllers are compared in Fig. 5.4 below. We can observe that the adaptive SNN successfully adapts to the unexpected left wing damage, and stays hovering around the starting point in the presence of an initial disturbance. However, the non-adaptive SNN fails to stabilize the hovering flight. The vehicle controlled by the non-adaptive SNN drifts dramatically in both y- and z-directions as shown in the snapshots in Fig. 5.5. More information on the vehicle's response can be found in the velocity comparison of both controllers in Fig. 5.6. In spite of the unexpected wing damage, the vehicle controlled by the adaptive SNN controller settles down quickly within only 1 second, and its velocity keeps maintained around zero until the end of the simulation. The high-frequency oscillations of the velocity time histories especially in x-direction are due to the flapping motions. The damage of the left wing results in an unexpected asymmetry of aerodynamic forces and moments between the left and the right wings. Therefore, the vehicle controlled by the non-adaptive SNN keeps drifting horizontally in y-direction. Moreover, removing the wing tip will reduce the overall thrust correspondingly, so the vehicle controlled by the non-adaptive SNN fails to maintain a zero velocity in z-direction as well.

The adaptive SNN control history in Fig. 5.7a shows that the controlled vehicle reaches and maintains the hovering equilibrium state in less than 1 second.

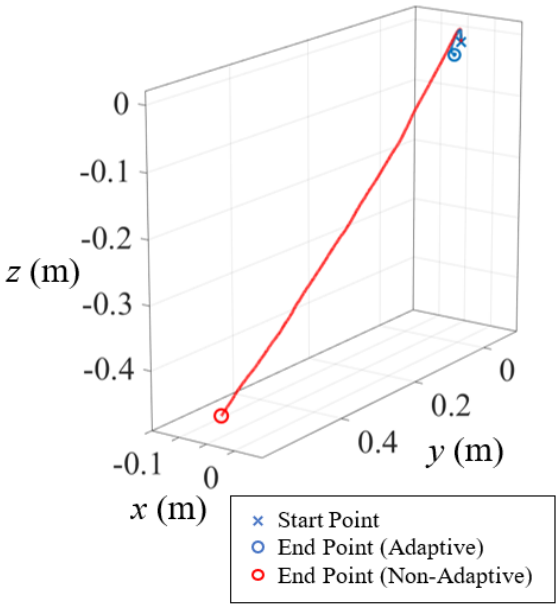


Figure 5.4: Case study 1: vehicle trajectory comparison between the adaptive and non-adaptive SNN controllers shows that the adaptive SNN successfully stabilizes the hovering flight, while the non-adaptive SNN drifts dramatically from the start point.

The amplitude control cost is much larger than the yaw, pitch and roll control cost, because these other three control inputs are affecting the vehicle's orientation only by biasing the stroke amplitude and mean stroke angle [35]. Furthermore, the online adaptive terms of amplitude, pitch and roll control input can indicate the differences between the adaptive and non-adaptive SNN control histories, as plotted in Fig. 5.7b. Adaptive pitch, roll and amplitude control histories represent how the adaptive terms are learned online by reinforcement learning to eliminate the body velocity deviations in x-, y- and z-directions, respectively. To stabilize the vertical motion of the flapping-wing vehicle, the adaptive amplitude control input is learned online to make up for the overall thrust and mass deduction caused by the left wing damage. Moreover, to keep the horizontal velocity component maintained around zero, the adaptive roll control input is learned online to compensate for the asymmetry of aerody-

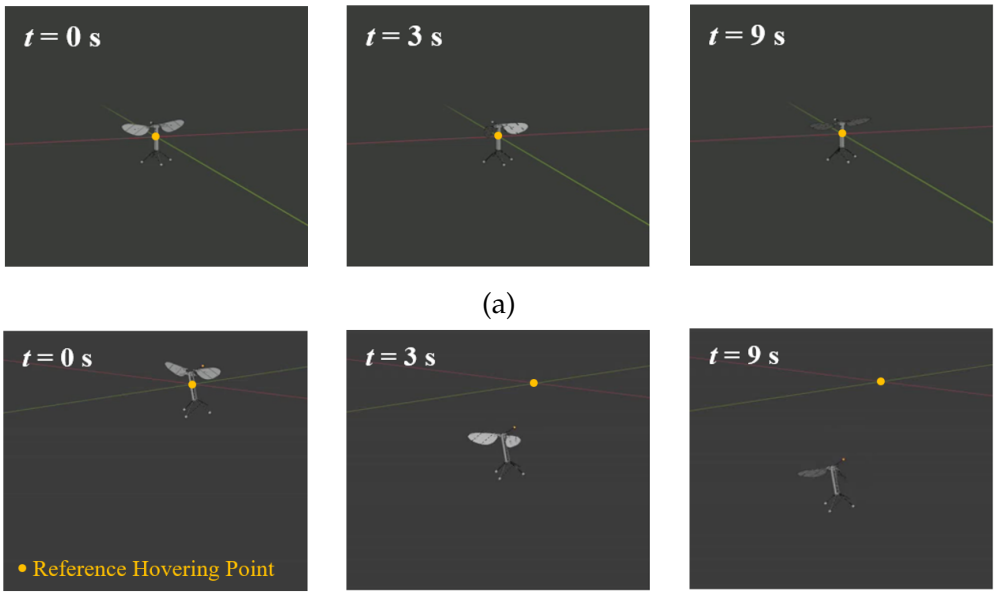


Figure 5.5: Case study 1: visualization of controlled hovering flight in Blender®. (a) The adaptive-SNN-controlled vehicle successfully adapts to the wing damage and stays around the orange reference hovering point. (b) The non-adaptive-SNN-controlled vehicle fails to adapt to the wing damage and drifts dramatically from the orange reference hovering point (see [221] for animation).

namic forces and moments between right and left wings.

5.5.2 Case Study 2: Flower Tracking with Wing Asymmetry

Wing asymmetry is one of the most common and formidable manufacturing uncertainties that FWMAVs may encounter, and can cause unmodeled variations in aerodynamic characteristics that promptly destabilizes the vehicles [38]. To demonstrate the ability of our new adaptive SNN controller to learn to compensate for wing asymmetry online, we introduce static random bias of the stroke amplitude and mean stroke angle of both wings in numerical simulations, and compare the vehicle responses of the adaptive and non-adaptive SNN controllers when the flapping-wing vehicle is commanded to track a robotic flower

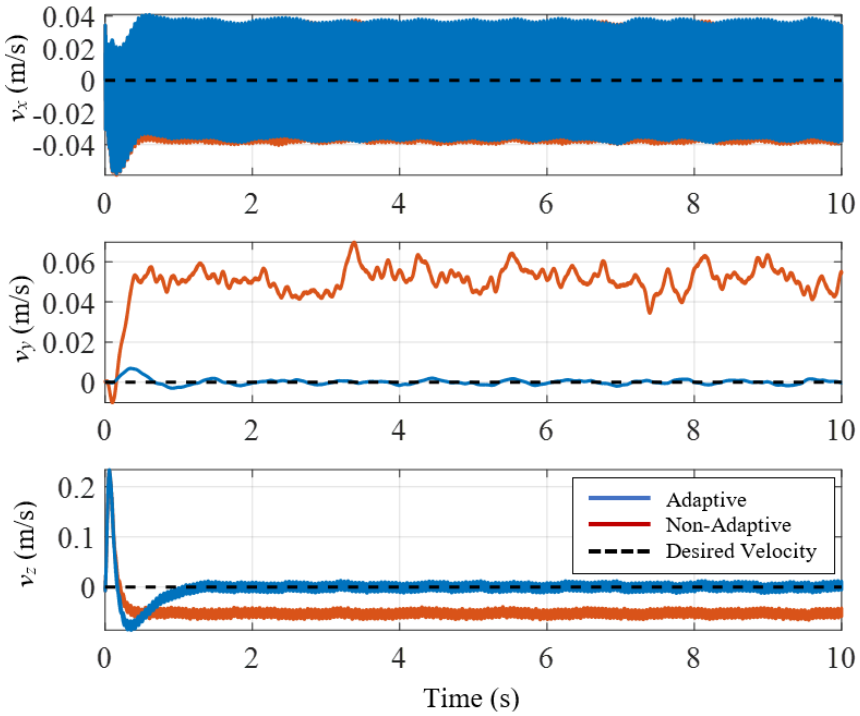


Figure 5.6: Case study 1: comparison of velocity time histories of the adaptive and non-adaptive SNN controllers shows that the adaptive SNN rapidly reaches and maintains zero velocity, while the non-adaptive SNN fails to keep v_y and v_z maintained around zero.

that moves sideways sinusoidally. In this case study, the control objective of tracking the sinusoidal velocity history of a moving robotic flower is inspired by experimental studies on insect behaviors [154]. This flower tracking scenario can be commonly seen when insects try to feed from flowers blown by wind. In this chapter, the estimation of the flower's velocity is accomplished using the event-based object detection algorithm developed in [69]. As shown in Fig. 5.8, the green bounding box is generated to include all the event-based optical flows that exceed a user-defined threshold. The position of the moving robotic flower is then assumed to be represented by the center of the identified bounding box. Given the estimated position, the flower's velocity is calculated using finite difference approximation. In numerical simulations, the flapping-wing vehicle is first commanded to hover with initial disturbance from time $t = 0$ to $t = 2$ s,

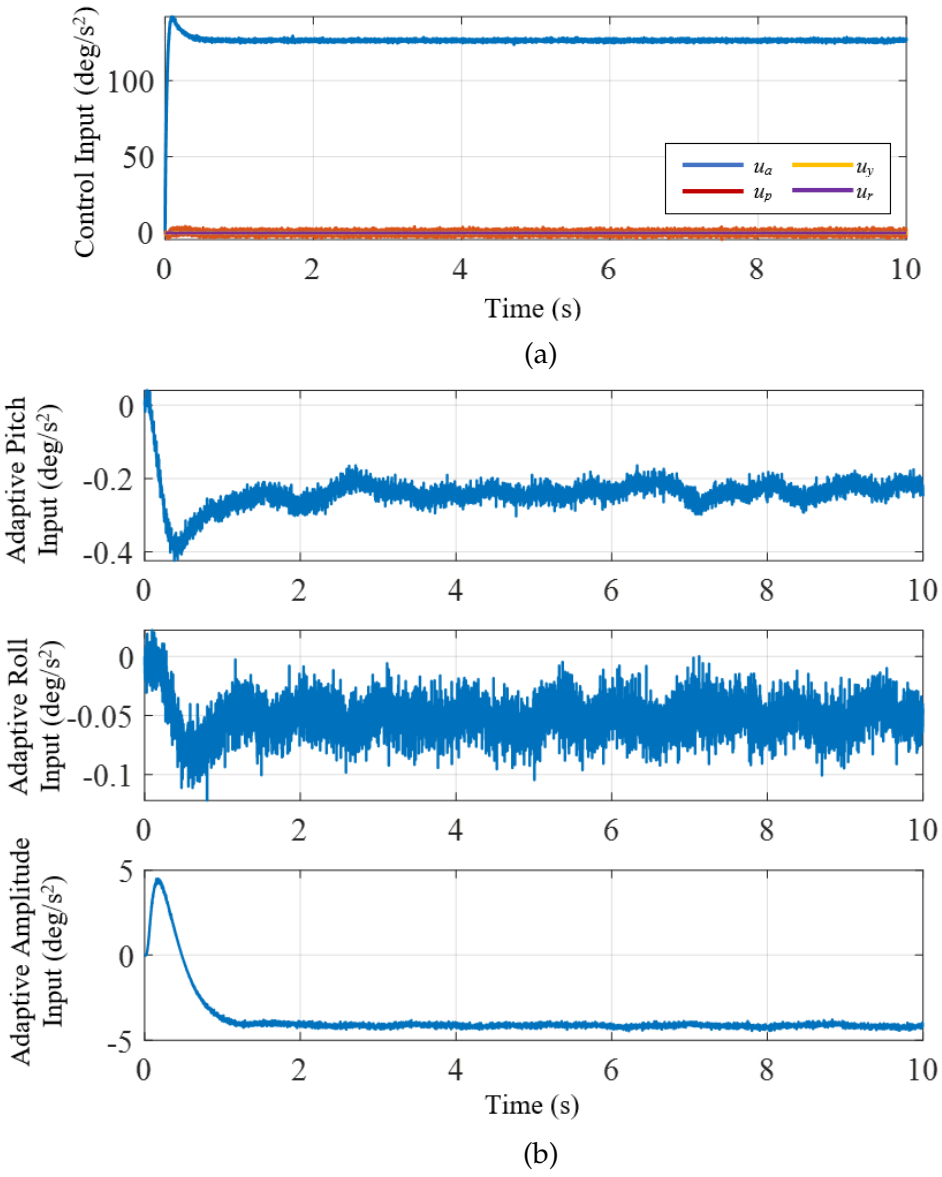


Figure 5.7: Case study 1: (a) control history of the adaptive SNN controller; (b) adaptive pitch, roll and amplitude control histories.

and then track the sinusoidal motion of the robotic flower after time $t = 2$ s. Fig. 5.8b shows that the estimated velocity of the moving robotic flower is very close to the true value. Then, we directly use the estimated flower velocity as a command velocity for the flapping-wing vehicle to follow. If the estimated velocity is smaller than zero, the sideslip angle β in the desired command vector will be set to be -90 deg. Otherwise, the sideslip angle β is maintained at 90 deg in

simulation.

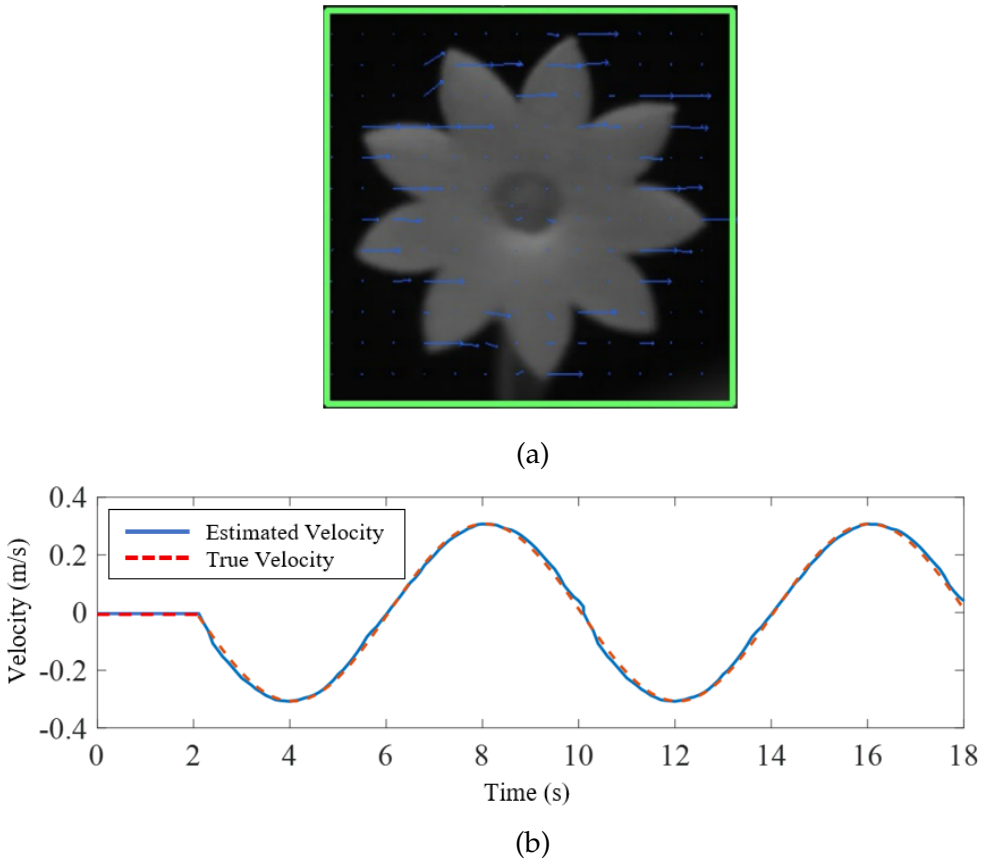


Figure 5.8: Case study 2: (a) the green bounding box is generated by an object detection algorithm using event-based optical flow (blue arrows); (b) the estimated velocity history of the moving robotic flower.

It can be observed from the velocity time histories in Fig. 5.9 that the flapping-wing vehicle controlled by the adaptive SNN controller quickly adapts to the initial disturbance and then remains hovering. The settling time of the adaptive SNN controller is less than 1 second. As soon as the flower-tracking command starts at time $t = 2$ s, the vehicle controlled by the adaptive SNN follows the sinusoidal motion in y-direction with extremely small tracking error while approximately remaining stationary in both x- and z-directions. Every time when the sideslip command changes its direction, the velocity deviates slightly from its desired set point, but the adaptive SNN learns to alleviate these

overshoots very quickly. However, the non-adaptive SNN fails to achieve the control objective of tracking the sinusoidally moving robotic flower. We can observe from the vehicle's velocity response that v_x and v_z are not maintained around zero as specified by the desired command. Due to the static roll torque bias caused by the wing asymmetry, there is a large undesired velocity tracking error for the horizontal velocity component v_y when the sideslip command is in the negative y-direction.

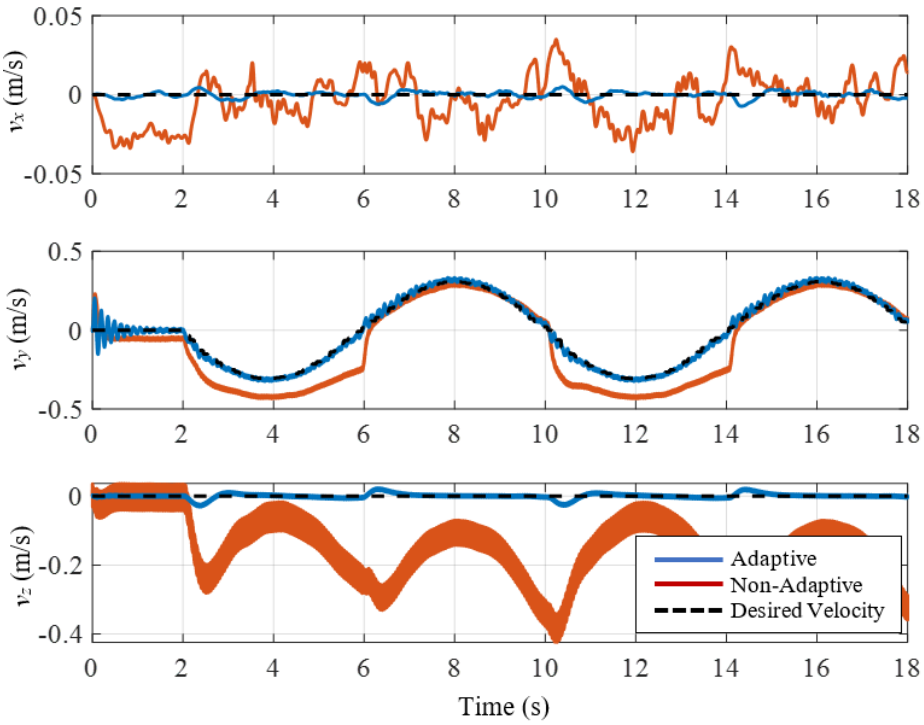
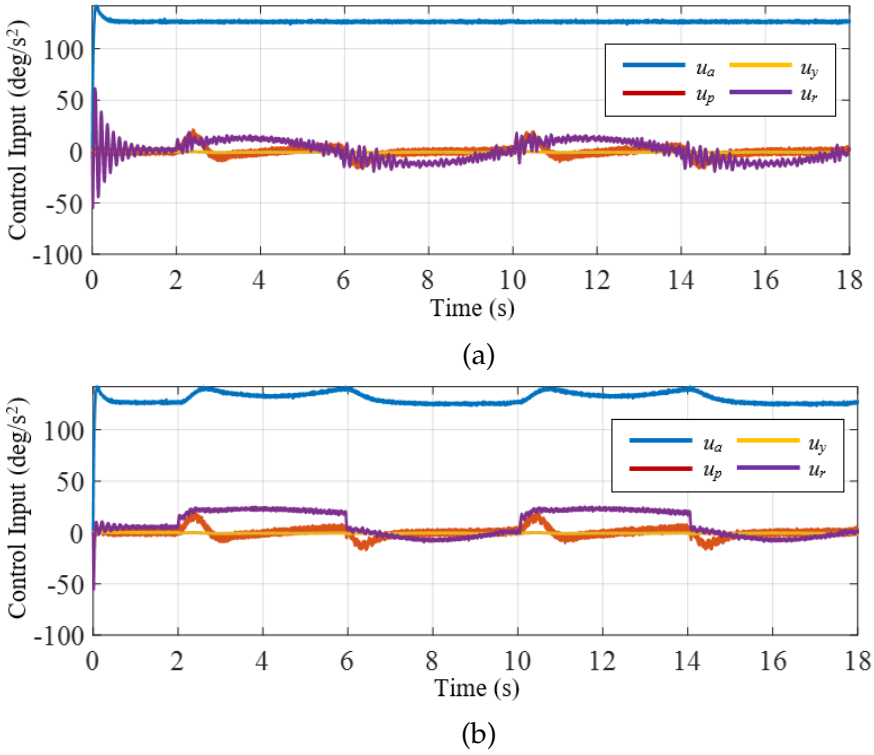


Figure 5.9: Case study 2: comparison of velocity time histories of the adaptive and non-adaptive SNN controllers shows that the adaptive SNN tracks the reference body velocity better than the non-adaptive one.

The control histories of the adaptive and non-adaptive SNN controllers are compared in Fig. 5.10. Both controlled flapping-wing vehicles reach a static equilibrium state within only 1 second when they are first commanded to hover with an initial disturbance. The roll control history of the adaptive SNN becomes approximately sinusoidal when the controlled vehicle starts to track the

moving robotic flower. Small oscillations occur in its roll control history, because the desired roll torque is changing very frequently as the vehicle tries to track the sinusoidal horizontal velocity of the flower. However, the roll control history of the non-adaptive SNN is not sinusoidal, which results in large tracking error in v_y as shown in Fig. 5.9. Furthermore, due to the static wing stoke angle bias, the amplitude control input of the non-adaptive SNN deviates dramatically from the desired set point.



(a)

(b)

Figure 5.10: Case study 2: (a) control history of the adaptive SNN controller; (b) control history of the non-adaptive SNN controller.

5.5.3 Case Study 3: Square Trajectory Following with State Measurement Error

In addition to modeling uncertainties such as wing damage and wing asymmetry introduced in previous case studies, a successful adaptive SNN controller should be able to account for state measurement errors that are inevitably caused by onboard sensor noise [47, 227, 85]. In this case study, we demonstrate the ability of our adaptive SNN controller to learn and adapt to onboard state measurement errors by comparing with a benchmark non-adaptive SNN controller. In numerical simulations, the flapping-wing vehicle is first commanded to hover with an initial disturbance from time $t = 0$ to $t = 0.5$ s, which is followed by a command to fly forward with its body velocity $V = 0.5$ m/s for 5 seconds and then a coordinated-turn command with the turning rate $\dot{\xi} = 90$ deg/s for 1 second at $V = 0.5$ m/s. This sequence of forward and turning commands is repeated three more times until a square trajectory is accomplished by the vehicle. Every corner of the square trajectory is achieved by conducting a 90-degree coordinated turn. The desired trajectory specified by this sequence of commands is plotted in a dashed line in Fig. 5.11 below. In the simulation, zero-mean random noise is added to the onboard measurement of the vehicle's roll, pitch and yaw angles and their rates, which will then be used as an essential information to compute the control signal.

The trajectories of the flapping-wing vehicle controlled by the adaptive and non-adaptive SNNs are compared in Fig. 5.11. The adaptive-SNN-controlled vehicle successfully follows the reference square trajectory with extremely small tracking error, and returns to the start point eventually. However, the non-adaptive-SNN-controlled vehicle dramatically deviates from the desired trajec-

tory, and does not go back to where it begins in the end as shown in the snapshots in Fig. 5.12. More details can be found in the comparison of the position time histories in Fig. 5.13. We can observe that the adaptive SNN controller successfully tracks the desired position while maintaining a constant altitude in spite of the initial disturbance and state measurement error. Nevertheless, the non-adaptive SNN controller not only fails to track the desired position in both x- and y-directions, but also cannot maintain the vehicle's altitude in z-direction.

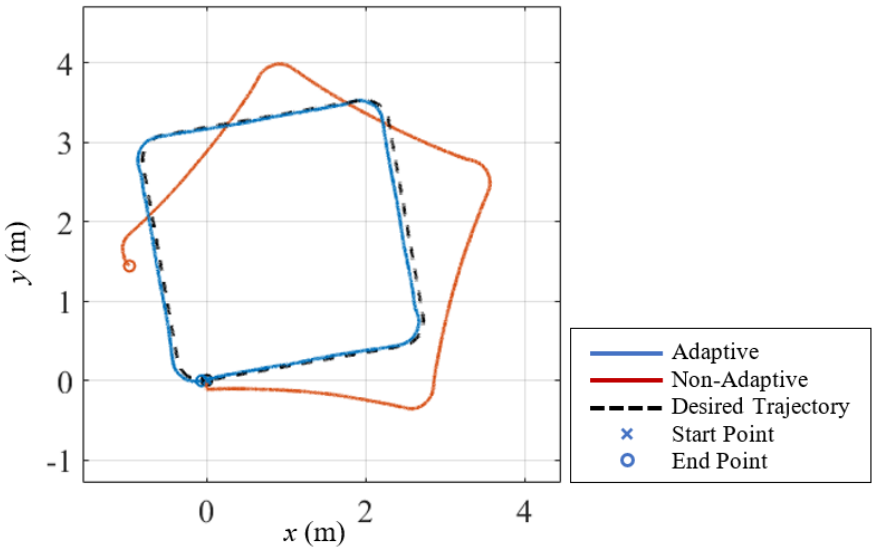
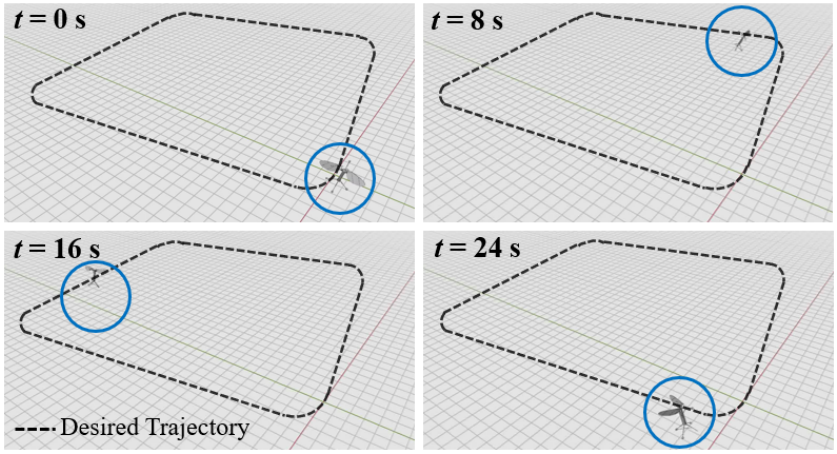
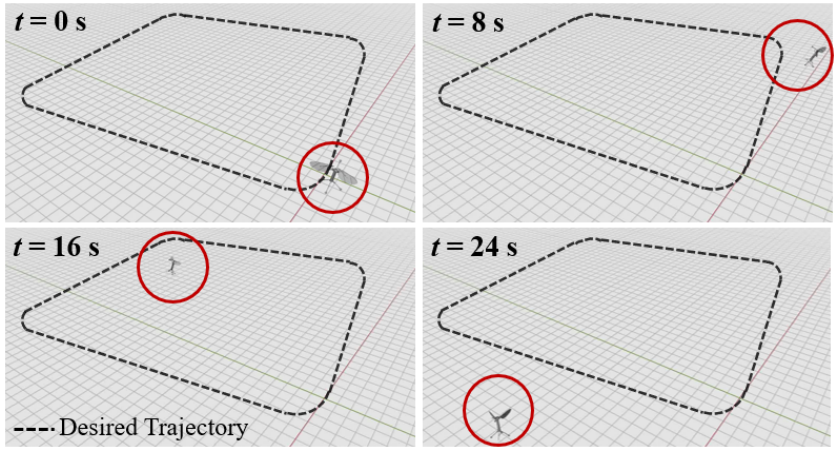


Figure 5.11: Case study 3: vehicle trajectory comparison between the adaptive and non-adaptive SNN controllers shows that the adaptive SNN follows the desired square trajectory with much smaller tracking error than the non-adaptive one.

In Fig. 5.14a, the control history of the adaptive SNN controller shows that the amplitude and pitch control cost is relatively larger than the yaw and roll control cost for the flapping-wing vehicle to fly in a square trajectory. The control histories take less than 4 seconds to approximately settle down to the hovering set-point values subject to the initial disturbance after the vehicle takes off. The four step changes of the yaw control input are due to the command of four 90-degree coordinated turns at the corner of the square trajectory. The online



(a)



(b)

Figure 5.12: Case study 3: visualization of controlled square trajectory following flight in Blender®. (a) The adaptive-SNN-controlled vehicle (inside blue circle) successfully adapts to the state measurement error and follows the desired square trajectory (black dashed line). (b) The non-adaptive-SNN-controlled vehicle (inside red circle) fails to adapt to the state measurement error and deviates dramatically from the desired square trajectory (see [222] for animation).

adaptive pitch, roll and amplitude control histories are plotted in Fig. 5.14b to further indicate the difference of control usage between the adaptive and non-adaptive SNN controllers. The results show that all three adaptive control inputs are relatively large at the beginning to compensate for the initial disturbance. When the vehicle starts to fly in a square trajectory, the adaptive SNN controller continuously adjusts the offline learned control law to account for the

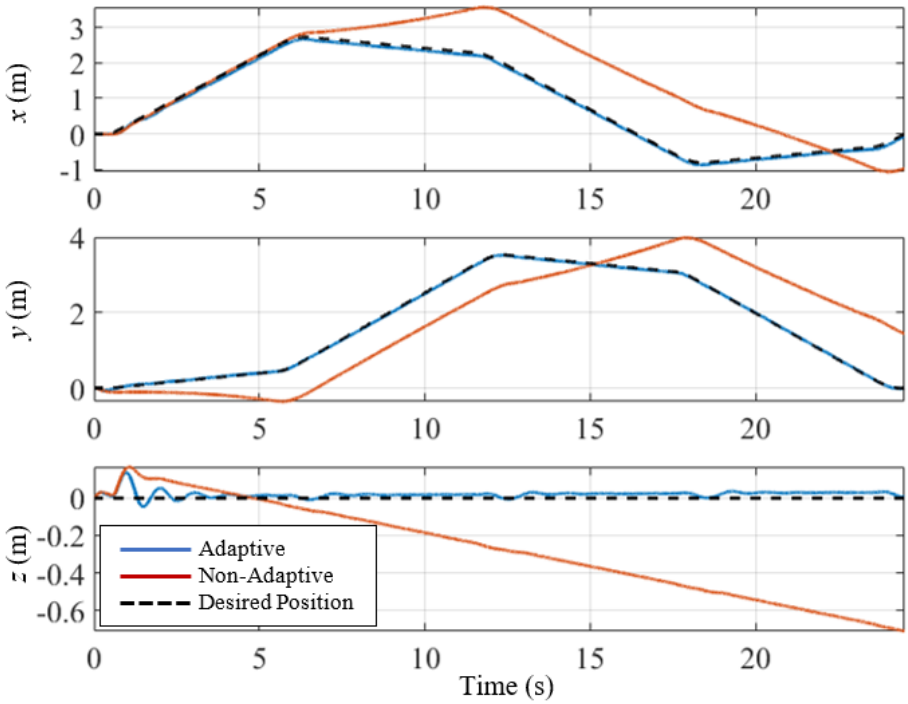
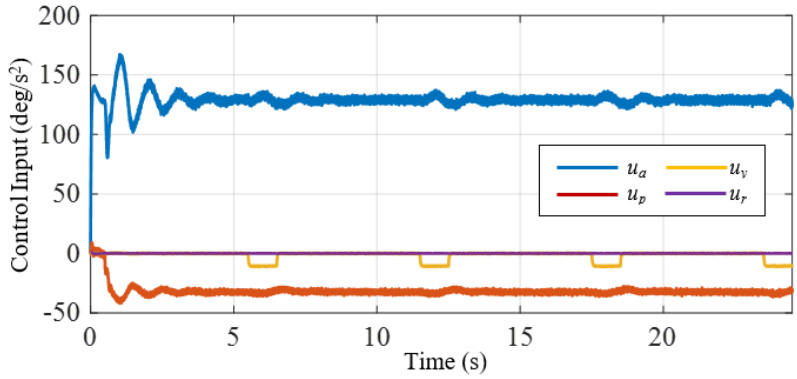


Figure 5.13: Case study 3: comparison of position time histories of the adaptive and non-adaptive SNN controllers shows that the adaptive SNN successfully tracks the desired position while maintaining a constant altitude, but the non-adaptive SNN fails to accomplish this control objective.

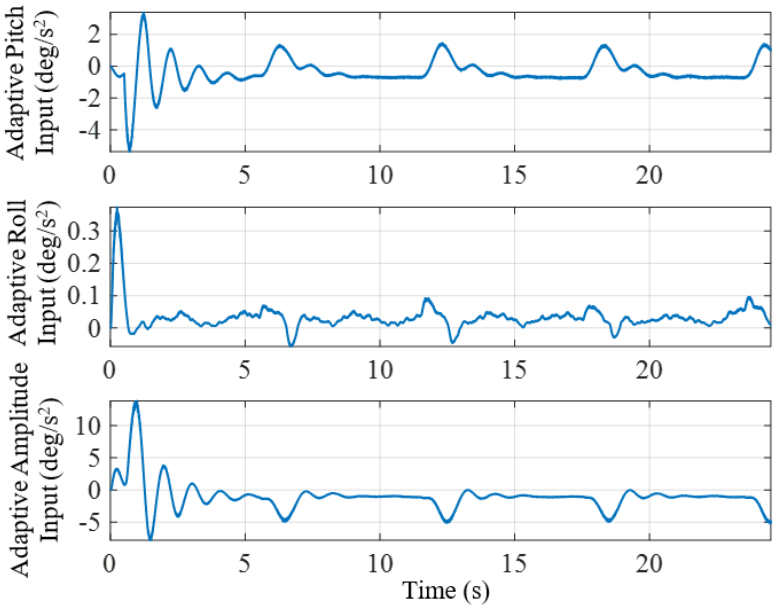
onboard state measurement error of velocity, Euler angles and angular rates that we introduce in simulation. In particular, the adjustments to the offline learned control law are increased every time when the vehicle is commanded to turn at the corner of the square trajectory.

5.5.4 Case Study 4: Coordinated Turn with Actuator Failure

As one of the most detrimental mechanical limitations that actual FWMAVs may be faced with, actuator failure will restrict the operational range of flapping wings and therefore cause extremely unstable motions on the fly [65, 25, 42]. In this case study, the capability of our new adaptive SNN controller to learn



(a)



(b)

Figure 5.14: Case study 3: (a) control history of the adaptive SNN controller; (b) adaptive pitch, roll and amplitude control histories.

to account for sudden partial actuator failure is demonstrated by comparing with a benchmark non-adaptive SNN controller. In numerical simulations, the flapping-wing vehicle is first commanded to hover with an initial disturbance from time $t = 0$ s to $t = 0.5$ s, which is followed by a command to ascend with its climb angle $\gamma = 90$ deg and body velocity $V = 0.5$ m/s for another 2 seconds. During the ascending flight, partial loss of effectiveness of the actuator is introduced by reducing the left-wing stroke angle amplitude by 10% since

time $t = 1.5$ s. The vehicle is then commanded to perform a coordinated turn with its turn rate $\dot{\xi} = 90$ deg/s from time $t = 2.5$ s to $t = 18.5$ s, which is followed by a command to descend with its climb angle $\gamma = -90$ deg for 2 seconds. The reference trajectory specified by this sequence of commands contains four full circles of coordinated turn, and is plotted in black dashed lines in Fig. 5.15.

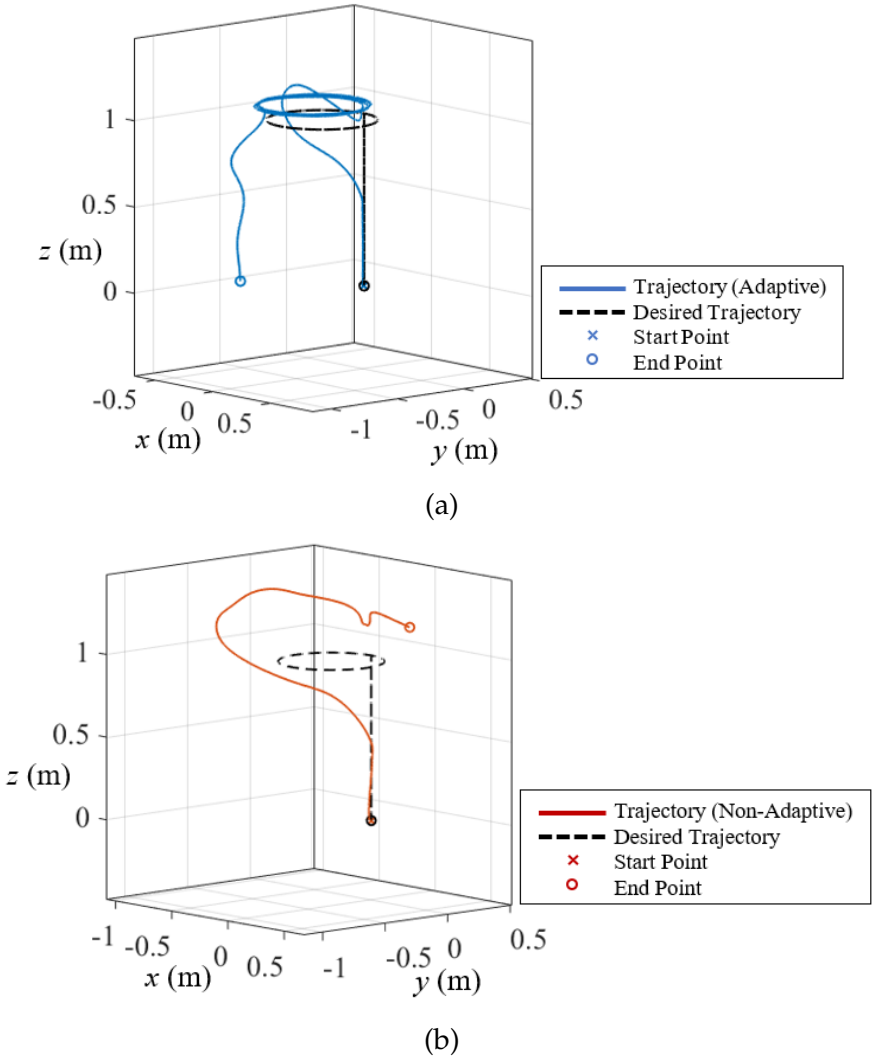


Figure 5.15: Case study 4: (a) vehicle trajectory of the adaptive SNN controller; (b) vehicle trajectory of the non-adaptive SNN controller. The vehicle controlled by the adaptive SNN successfully accomplishes the coordinated turn with small displacement from the desired trajectory, while the non-adaptive-SNN-controlled vehicle goes completely out of control.

The trajectories of the vehicles controlled by the adaptive and non-adaptive

SNNs are compared in Fig. 5.15. It can be observed that both SNN controllers are able to track the reference trajectory until the moment when the actuator partially fails. Even though its trajectory slightly deviates from the reference trajectory during the ascending flight, the adaptive SNN controller quickly adapts to the actuator failure on the fly. As shown in Fig. 5.15a, the adaptive-SNN-controlled vehicle accomplishes the coordinated turn with small displacement from the desired trajectory, and safely lands in the end. However, Fig. 5.15b shows that the vehicle controlled by the non-adaptive SNN goes completely out of control after the actuator partially fails, and does not accomplish the coordinated turn and descending flight as specified by the desired command input.

As shown in the velocity comparison in Fig. 5.16, velocity responses of the adaptive and non-adaptive SNNs both change dramatically at the moment of actuator failure, but only the adaptive SNN successfully takes the body velocity back to the desired reference value. In particular, a large spike in the body velocity v_y can be observed in the velocity time history of the adaptive SNN controller at time $t = 1.5$ s when the actuator partially fails, which results in a small displacement from the desired vertical ascending trajectory in Fig. 5.15a. However, the adaptive SNN controller is able to settle down the velocity in less than 3 seconds, which indicates that it can quickly learn to account for the unexpected partial actuator failure. Nevertheless, the velocity response of the non-adaptive SNN goes out of bound immediately after the actuator partially fails. The body velocity of the vehicle even exceeds the velocity limit of our current FWMAV dynamic model, thereby resulting in a terrible crash eventually. Therefore, as shown in Fig. 5.17b, the control cost of the non-adaptive SNN controller required to stabilize the wild vehicle increases drastically approximately at time $t = 4.5$ s. On the contrary, the control history of the adaptive SNN in Fig. 5.17a

shows that the amplitude and pitch control inputs overshoot every time when the command input changes, but they both reach and maintain the desired set-point values rapidly in less than 2 seconds.

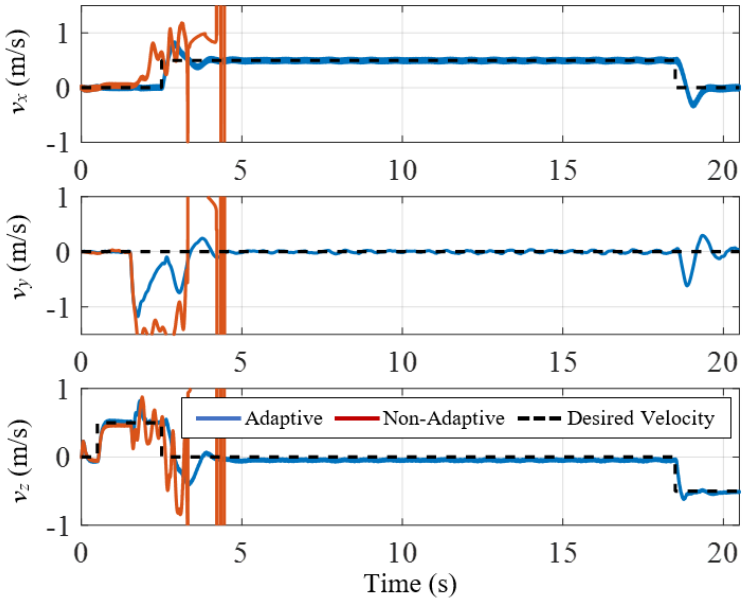
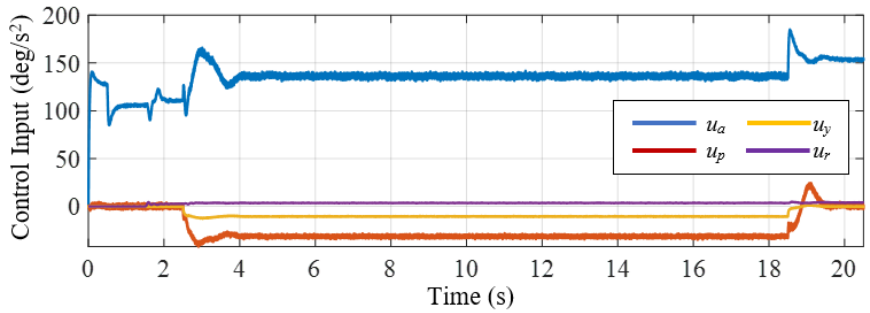
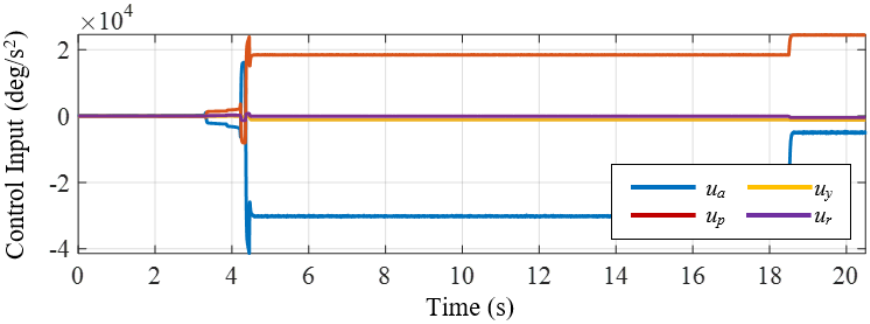


Figure 5.16: Case study 4: comparison of body velocity time histories of the adaptive and non-adaptive SNN controllers shows that the adaptive SNN brings the vehicle velocity back to the desired value quickly after the actuator fails at time $t = 1.5$ s, while the velocity response of the non-adaptive SNN goes out of bound immediately.



(a)



(b)

Figure 5.17: Case study 4: (a) control history of the adaptive SNN controller; (b) control history of the non-adaptive SNN controller.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

This dissertation covers biologically inspired sensing and control approaches for micro aerial vehicles. We first present a novel and general control design approach, referred to as fast-tracking controller or FTC in short, applicable to air vehicles flying through cellular flows. The new design philosophy only requires approximate knowledge of a few key flow parameters, such as the mean velocity, the typical vortex length scale, and the typical vortex timescale, and allows the vehicle, with a feedback controller in the loop, to behave like an ideal fast-tracking particle by means of implicit model following. The simulation results show that, indeed, FTC control allows the vehicle to take advantage of beneficial tailwinds by means of onboard propulsion and local wind measurements, ultimately reducing the travel time and energy consumption required to traverse the cellular flow. The energy-harvesting potential of FTC is demonstrated by considering two benchmark control problems: minimum-energy problem and minimum-time problem. The comparison with classic optimal solutions obtained via LQR and BBC control theory shows that, by following the ideal response of the fast-tracking particle, the FTC-controlled air vehicle achieves larger average horizontal velocity than the purely thrust-driven one. Furthermore, the FTC-controlled vehicle can reach and maintain a desired steady-state velocity through the cellular flow with less control effort than the LQR-controlled vehicle, and reach a desired horizontal position faster than the BBC-controlled vehicle.

Furthermore, with reasonable restrictions on the range of vortex length scale and an appropriate choice of incremental length scale, the vortex timescale, of

which the aforementioned FTC control design requires prior knowledge, can be estimated onboard accurately within a permissible range of error by a SINDy-based flow parameter estimation algorithm. However, when testing the estimation algorithm on different simulated data sets, we observe that small errors in the estimated vortex length scale make the algorithm unable to distinguish one ultimate candidate function from others, thus resulting in dispersed weights and a large deviation when estimating the mean velocity and the vortex timescale. Errors in the estimated vortex length scales may be caused by the low resolution and inappropriate choice of the incremental length scale when constructing the library of all potential candidate functions. Finally, the novel FTC control design and corresponding flow parameter estimation algorithm presented in this dissertation are also potentially applicable to other types of vehicles, such as underwater vehicles in Langmuir-type water cells.

In addition to particle transport theory, inspirations on MAV sensing and control can be drawn from natural flyers, such as hawk moth. As an essential step of reverse-engineering moth's neural control systems, this dissertation presents a novel regression-based spike train decoding method that uncovers the precise mapping from the spike trains of ten primary flight muscles to the resulting forces and torques on the moth body for the flight of a hawk moth visually tracking a robotic flower. The new relative-time kernel design considers the extra relative spike timing information among multiple spike trains by comparing every pair of correlated spike trains across the flight muscle population. The relative-time-kernel-based decoder captures the data variance better and predicts the resulting forces and torques more accurately than the benchmark instantaneous-kernel-based and rate-based decoders. Furthermore, compared to force prediction, the proposed relative-time kernel has a much higher per-

centage improvement over the instantaneous kernel in torque prediction. Regarding the future work beyond the relative-time kernel design approach described in this dissertation, this new regression-based spike train decoder can be used to train an SNN model of hawk moth sensorimotor control.

In the end, inspired by insect's flapping flight control strategies, this dissertation presents a novel two-phase adaptive SNN control design approach that allows the FWMAVs to conduct a full range of maneuvers, such as hovering, flower tracking, square trajectory following and coordinated turn, in the presence of significant unmodeled uncertainties. The offline supervised learning phase provides a fixed SNN approximation of a classical gain-scheduled PIF compensator designed to stabilize the ideal FWMAV dynamic model. During the online learning phase, SNNs are trained by policy gradient reinforcement learning rule to minimize the state deviation and adapt to unmodeled uncertainties. In numerical simulations, the adaptive SNN controller is shown to outperform a benchmark non-adaptive SNN controller in controlling the full-envelope flight of a flapping-wing robot known as RoboBee when unforeseen situations, such as parameter variations, unmodeled dynamics, measurement errors and actuator failures, are encountered on the fly. The bio-inspired sensing and control strategies presented in this dissertation can be potentially applied to develop the next generation of smart, agile and adaptive MAVs that are highly efficient and robust in the presence of wind disturbances and modeling uncertainties.

BIBLIOGRAPHY

[1] Mujahid Abdulrahim, Abdulghani Mohamed, and Simon Watkins. Control strategies for flight in extreme turbulence. In *AIAA Guidance, Navigation, and Control Conference*, page 1909, 2017.

[2] Edgar D Adrian and Yngve Zotterman. The impulses produced by sensory nerve-endings: Part ii. the response of a single end-organ. *Journal of Physiology*, 61(2):151–171, 1926.

[3] Hubert Airy. The soaring of birds. *Nature*, 28(709):103–103, 1883.

[4] Zsuzsa Ákos, Máté Nagy, Severin Leven, and Tamás Vicsek. Thermal soaring flight of birds and unmanned aerial vehicles. *Bioinspiration & Biomimetics*, 5(4):045003, 2010.

[5] Alberto Aliseda, Alain Cartellier, F Hainaux, and Juan C Lasheras. Effect of preferential concentration on the settling velocity of heavy particles in homogeneous isotropic turbulence. *Journal of Fluid Mechanics*, 468:77–105, 2002.

[6] Halim Alwi, Christopher Edwards, and Chee Pin Tan. Fault tolerant control and fault detection and isolation. In *Fault Detection and Fault-Tolerant Control Using Sliding Modes*, pages 7–27. Springer, 2011.

[7] Klas Andersson, Isaac Kaminer, Vladimir Dobrokhodov, and Venanzio Cichella. Thermal centering control for autonomous soaring: stability analysis and flight test results. *Journal of Guidance, Control, and Dynamics*, 35(3):963–975, 2012.

[8] Ehsan Arabi, Benjamin C Gruenwald, Tansel Yucelen, and Nhan T Nguyen. A set-theoretic model reference adaptive control architecture for disturbance rejection and uncertainty suppression with strict performance guarantees. *International Journal of Control*, 91(5):1195–1208, 2018.

[9] Leemon Baird and Andrew Moore. Gradient descent for general reinforcement learning. *Advances in neural information processing systems*, 11, 1998.

[10] Laura Barnes, MaryAnne Fields, and Kimon Valavanis. Unmanned ground vehicle swarm formation control using potential fields. In *2007 Mediterranean Conference on Control & Automation*, pages 1–8. IEEE, 2007.

[11] Andrew B Barron, Kevin N Gurney, Lianne FS Meah, Eleni Vasilaki, and James AR Marshall. Decision-making and action selection in insects: inspiration from vertebrate-based theories. *Frontiers in Behavioral Neuroscience*, 9:216, 2015.

[12] Kelli AC Baumgartner, Silvia Ferrari, and Anil V Rao. Optimal control of an underwater sensor network for cooperative target tracking. *IEEE Journal of Oceanic Engineering*, 34(4):678–697, 2009.

[13] Jonathan Baxter and Peter L Bartlett. Direct gradient-based reinforcement learning. In *2000 IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 3, pages 271–274. IEEE, 2000.

[14] Dimitri Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1996.

[15] Luca Biferale, Fabio Bonaccorso, Michele Buzzicotti, Patricio Clark Di Leoni, and Kristian Gustavsson. Zermelo's problem: Optimal point-to-point navigation in 2d turbulent flows using reinforcement learning. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 29(10):103138, 2019.

[16] R Byron Bird, Warren E Stewart, and Edwin N Lightfoot. *Transport Phenomena*, volume 1. John Wiley & Sons, 2006.

[17] Horst Bleckmann and Randy Zelick. Lateral line system of fish. *Integrative zoology*, 4(1):13–25, 2009.

[18] Scott A Bollt and Gregory P Bewley. How to extract energy from turbulence in flight by fast tracking. *Journal of Fluid Mechanics*, 921, 2021.

[19] Emery N Brown, Robert E Kass, and Partha P Mitra. Multiple neural spike train data analysis: state-of-the-art and future challenges. *Nature Neuroscience*, 7(5):456–461, 2004.

[20] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[21] William Burnett, Scott Harper, Ruth Preller, Gregg Jacobs, and Kevin LaCroix. Overview of operational ocean forecasting in the us navy: Past, present, and future. *Oceanography*, 27(3):24–31, 2014.

[22] Theodore Castro-Santos. Optimal swim speeds for traversing velocity barriers: an analysis of volitional high-speed swimming behavior of migratory fishes. *Journal of Experimental Biology*, 208(3):421–432, 2005.

[23] Dominique Chabot. Trends in drone research and applications as the journal of unmanned vehicle systems turns five. *Journal of Unmanned Vehicle Systems*, 6(1):6–15, 2018.

[24] Chen Chen and Tianyu Zhang. A review of design and fabrication of the bionic flapping wing micro air vehicles. *Micromachines*, 10(2):144, 2019.

[25] Yufeng Chen, Hongqiang Wang, E Farrell Helbling, Noah T Jafferis, Raphael Zufferey, Aaron Ong, Kevin Ma, Nicholas Gravish, Pakpong Chirarattananon, Mirko Kovac, et al. A biologically inspired, flapping-wing, hybrid aerial-aquatic microrobot. *Science robotics*, 2(11):eaao5619, 2017.

[26] Zhe Chen. An overview of bayesian methods for neural spike train analysis. *Computational Intelligence and Neuroscience*, 2013, 2013.

[27] Bo Cheng. Flying of insects. *Bioinspired Structures and Design*, page 271, 2020.

[28] Yao-Wei Chin, Jia Ming Kok, Yong-Qiang Zhu, Woei-Leong Chan, Javaan S Chahl, Boo Cheong Khoo, and Gih-Keong Lau. Efficient flapping wing drone arrests high-speed flight using post-stall soaring. *Science Robotics*, 5(44):eaba2386, 2020.

[29] Pakpong Chirarattananon, Kevin Y Ma, Richard Cheng, and Robert J Wood. Wind disturbance rejection for an insect-scale flapping-wing robot. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 60–67. IEEE, 2015.

[30] Pakpong Chirarattananon, Kevin Y Ma, and Robert J Wood. Adaptive control for takeoff, hovering, and landing of a robotic fly. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3808–3815. IEEE, 2013.

[31] Pakpong Chirarattananon, Kevin Y Ma, and Robert J Wood. Perching with a robotic insect using adaptive tracking control and iterative learning control. *The International Journal of Robotics Research*, 35(10):1185–1206, 2016.

[32] Timothy H Chung, Michael R Clement, Michael A Day, Kevin D Jones, Duane Davis, and Marianna Jones. Live-fly, large-scale field experimentation for large numbers of fixed-wing uavs. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1255–1262. IEEE, 2016.

[33] Mark M Churchland and Stephen G Lisberger. Shifts in the population response in the middle temporal visual area parallel perceptual and motor illusions produced by apparent motion. *Journal of Neuroscience*, 21(23):9387–9402, 2001.

[34] Taylor S Clawson. *Neuromorphic sensing and control of autonomous micro-aerial vehicles*. PhD thesis, Sibley School of Mechanical and Aerospace Engineering, Cornell University, 2019.

[35] Taylor S Clawson, Silvia Ferrari, E Farrell Helbling, Robert J Wood, Bo Fu, Andy Ruina, and Z Jane Wang. Full flight envelope and trim map of flapping-wing micro aerial vehicles. *Journal of Guidance, Control, and Dynamics*, 43(12):2218–2236, 2020.

[36] Taylor S Clawson, Silvia Ferrari, Sawyer B Fuller, and Robert J Wood. Spiking neural network (snn) control of a flapping insect-scale robot. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 3381–3388. IEEE, 2016.

[37] Taylor S Clawson, Silvia Ferrari, Sawyer B Fuller, and Robert J Wood. Spiking neural network (snn) control of a flapping insect-scale robot. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 3381–3388. IEEE, 2016.

[38] Taylor S Clawson, Terrence C Stewart, Chris Eliasmith, and Silvia Ferrari. An adaptive spiking neural controller for flapping insect-scale robots. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7. IEEE, 2017.

[39] Noah J Cowan, Mert M Ankarali, Jonathan P Dyhr, Manu S Madhav, Eatai Roth, Shahin Sefati, Simon Sponberg, Sarah A Stamper, Eric S Fortune, and Thomas L Daniel. Feedback control as a framework for understanding tradeoffs in biology. *American Zoologist*, 54(2):223–237, 2014.

[40] Avik De, Rebecca McGill, and Robert J Wood. An efficient, modular controller for flapping flight composing model-based and model-free components. *The International Journal of Robotics Research*, 41(4):441–457, 2022.

[41] Markus Deittert, Arthur Richards, Chris Toomer, and Anthony Pipe. Dynamic soaring flight in turbulence. In *AIAA Guidance, Navigation, and Control Conference*, page 6012, 2009.

[42] Alexis Lussier Desbiens, Yufeng Chen, and Robert J Wood. A wing characterization method for flapping-wing robotic insects. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1367–1373. IEEE, 2013.

[43] Michael H Dickinson and Florian T Muijres. The aerodynamics and control of free flight manoeuvres in drosophila. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 371(1704):20150388, 2016.

[44] VA Drake and RA Farrow. The influence of atmospheric structure and motions on insect migration. *Annual review of entomology*, 33(1):183–210, 1988.

[45] Kamak Ebadi, Yun Chang, Matteo Palieri, Alex Stephens, Alex Hatteland, Eric Heiden, Abhishek Thakur, Nobuhiro Funabiki, Benjamin Morrell, Sally Wood, et al. Lamp: Large-scale autonomous mapping and positioning for exploration of perceptually-degraded subterranean environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 80–86. IEEE, 2020.

[46] Basil El Jundi and Marie Dacke. Insect orientation: the drosophila wind compass pathway. *Current Biology*, 31(2):R83–R85, 2021.

[47] Yonina C Eldar, Aharon Ben-Tal, and Arkadi Nemirovski. Robust mean-squared error estimation in the presence of model uncertainties. *IEEE Transactions on Signal Processing*, 53(1):168–181, 2004.

[48] Chris Eliasmith and Charles H Anderson. *Neural engineering: Computation, representation, and dynamics in neurobiological systems*. MIT press, 2003.

[49] Charles P Ellington. The novel aerodynamics of insect flight: applications to micro-air vehicles. *Journal of Experimental Biology*, 202(23):3439–3448, 1999.

[50] Nourhan Elmeseiry, Nancy Alshaer, and Tawfik Ismail. A detailed survey and future directions of unmanned aerial vehicles (uavs) with potential applications. *Aerospace*, 8(12):363, 2021.

[51] María José Fernández, Dwight Springthorpe, and Tyson L Hedrick. Neuromuscular and biomechanical compensation for wing asymmetry in insect hovering flight. *Journal of Experimental Biology*, 215(20):3631–3638, 2012.

[52] Silvia Ferrari. *Algebraic and adaptive learning in neural control systems*. PhD thesis, Princeton University, 2002.

[53] Silvia Ferrari. Multiobjective algebraic synthesis of neural control systems by implicit model following. *IEEE Transactions on Neural Networks*, 20(3):406–419, 2009.

[54] Silvia Ferrari and Greg Foderaro. A potential field approach to finding minimum-exposure paths in wireless sensor networks. In *2010 IEEE International Conference on Robotics and Automation*, pages 335–341. IEEE, 2010.

[55] Silvia Ferrari and Robert F Stengel. Classical/neural synthesis of nonlinear control systems. *Journal of Guidance, Control, and Dynamics*, 25(3):442–448, 2002.

[56] Silvia Ferrari and Robert F Stengel. Online adaptive critic flight control. *Journal of Guidance, Control, and Dynamics*, 27(5):777–786, 2004.

[57] Silvia Ferrari and Robert F Stengel. Online adaptive critic flight control. *Journal of Guidance, Control, and Dynamics*, 27(5):777–786, 2004.

[58] Silvia Ferrari and Robert F Stengel. Smooth function approximation using neural networks. *IEEE Transactions on Neural Networks*, 16(1):24–38, 2005.

[59] Edward Fiorelli, Naomi Ehrich Leonard, Pradeep Bhatta, Derek A Paley, Ralf Bachmayer, and David M Fratantoni. Multi-auv control and adaptive sampling in monterey bay. *IEEE Journal of Oceanic Engineering*, 31(4):935–948, 2006.

[60] Greg Foderaro, Craig Henriquez, and Silvia Ferrari. Indirect training of a spiking neural network for flight control via spike-timing-dependent synaptic plasticity. In *49th IEEE Conference on Decision and Control (CDC)*, pages 911–917. IEEE, 2010.

[61] NP Foster, I Postlethwaite, and DJ Walker. Rotorcraft control system design for rejection of atmospheric turbulence. In *IEE Colloquium on Multivariable Methods for Flight Control Applications*, pages 1–4. IET, 1994.

[62] Jessica L Fox, Adrienne L Fairhall, and Thomas L Daniel. Encoding properties of haltere neurons enable motion feature detection in a biological gyroscope. *Proceedings of the National Academy of Sciences*, 107(8):3840–3845, 2010.

[63] Trevor Gale, Erich Elsen, and Sara Hooker. The state of sparsity in deep neural networks. *arXiv preprint arXiv:1902.09574*, 2019.

[64] Boris Galkin, Jacek Kibilda, and Luiz A DaSilva. Uavs as mobile infrastructure: Addressing battery lifetime. *IEEE Communications Magazine*, 57(6):132–137, 2019.

[65] Hejia Gao, Wei He, Youmin Zhang, and Changyin Sun. Adaptive finite-time fault-tolerant control for uncertain flexible flapping wings based on rigid finite element method. *IEEE Transactions on Cybernetics*, 2021.

[66] Bartolome Garau, Alberto Alvarez, and Gabriel Oliver. Auv navigation through turbulent ocean environments supported by onboard h-adcp. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3556–3561. IEEE, 2006.

[67] Michael Garstang, Steven Greco, George D Emmitt, Tricia A Miller, and Michael Lanzone. An instrumented golden eagle's (aquila chrysaetos) long-distance flight behavior. *Animals*, 12(11):1470, 2022.

[68] Nikola Gavrilovic, Emmanuel Benard, Philippe Pastor, and Jean-Marc Moschetta. Performance improvement of small unmanned aerial vehicles through gust energy harvesting. *Journal of Aircraft*, 55(2):741–754, 2018.

[69] Jake Richard Gemerek. *Active vision and perception*. PhD thesis, Cornell University, 2020.

[70] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[71] Ashkan Ghanbarzadeh-Dagheyan, Nader Jalili, and Mohammad Taghi Ahmadian. A holistic survey on mechatronic systems in micro/nano scale with challenges and applications. *Journal of Micro-Bio Robotics*, 17(1):1–22, 2021.

[72] Samanwoy Ghosh-Dastidar and Hojjat Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.

[73] GH Good, PJ Ireland, GP Bewley, E Bodenschatz, LR Collins, and Z Warhaft. Settling regimes of inertial particles in isotropic turbulence. *Journal of Fluid Mechanics*, 759, 2014.

[74] Arnulf BA Graf, Adam Kohn, Mehrdad Jazayeri, and J Anthony Movshon. Decoding the activity of neuronal populations in macaque primary visual cortex. *Nature Neuroscience*, 14(2):239–245, 2011.

[75] Johanna Grönroos, Martin Green, and Thomas Alerstam. To fly or not to fly depending on winds: shorebird migration in different seasonal wind regimes. *Animal Behaviour*, 83(6):1449–1457, 2012.

[76] P Gunnarson, I Mandralis, G Novati, P Koumoutsakos, and John O Dabiri. Learning efficient navigation in vortical flow fields. *Nature Communications*, 12:7143, 2021.

[77] David A Handelman and Robert F Stengel. Combining expert system and analytical redundancy concepts for fault-tolerant flight control. *Journal of Guidance, Control, and Dynamics*, 12(1):39–45, 1989.

[78] Mostafa Hassanalian and Abdessattar Abdelkefi. Classifications, applications, and design challenges of drones: A review. *Progress in Aerospace Sciences*, 91:99–131, 2017.

[79] Sabine Hauert, Severin Leven, Maja Varga, Fabio Ruini, Angelo Cangelosi, Jean-Christophe Zufferey, and Dario Floreano. Reynolds flocking in reality with fixed-wing robots: communication range vs. maximum turning rate. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5015–5020. IEEE, 2011.

[80] Daniel R Hayes and James H Morison. Determining turbulent vertical velocity, and fluxes of heat and salt with an autonomous underwater vehicle. *Journal of Atmospheric and Oceanic Technology*, 19(5):759–779, 2002.

[81] Wei He, Xinxing Mu, Liang Zhang, and Yao Zou. Modeling and trajectory tracking control for flapping-wing micro aerial vehicles. *IEEE/CAA Journal of Automatica Sinica*, 8(1):148–156, 2020.

[82] Wei He, Zichen Yan, Changyin Sun, and Yunan Chen. Adaptive neural network control of a flapping wing micro aerial vehicle with disturbance observer. *IEEE transactions on cybernetics*, 47(10):3452–3465, 2017.

[83] Geoffrey A Hollinger, Arvind A Pereira, Jonathan Binney, Thane Somers, and Gaurav S Sukhatme. Learning uncertainty in ocean current predictions for safe and reliable navigation of underwater vehicles. *Journal of Field Robotics*, 33(1):47–66, 2016.

[84] John J Hopfield. Pattern recognition computation using action potential timing for stimulus representation. *Nature*, 376(6535):33–36, 1995.

[85] Jonathan P How and Michael Tillerson. Analysis of the impact of sensor noise on formation flying control. In *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, volume 5, pages 3986–3991. IEEE, 2001.

[86] Newton Howard. The future of the brain and beyond. In *2019 IEEE 18th International Conference on Cognitive Informatics & Cognitive Computing (ICCI*CC)*, pages 2–2. IEEE, 2019.

[87] Qinglei Hu, Bing Xiao, Bo Li, and Youmin Zhang. *Fault-Tolerant Attitude Control of Spacecraft*. Elsevier, 2021.

[88] David H Hubel and Torsten N Wiesel. Receptive fields of single neurones in the cat's striate cortex. *Journal of Physiology*, 148(3):574–591, 1959.

[89] Mircea Hulea, George-Iulian Uleru, Adrian Burlacu, and Constantin-Florin Caruntu. Bioinspired snn for robotic joint control. In *2020 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, pages 1–5. IEEE, 2020.

[90] Mircea Hulea, George Iulian Uleru, and Constantin Florin Caruntu. Adaptive snn for anthropomorphic finger control. *Sensors*, 21(8):2730, 2021.

[91] David G Hull et al. *Fundamentals of airplane flight mechanics*, volume 19. Springer, 2007.

[92] Kenneth W Iliff. Identification and stochastic control of an aircraft flying in turbulence. *Journal of Guidance and Control*, 1(2):101–108, 1978.

[93] Sara Janhäll. Review on urban vegetation and particle air pollution–deposition and dispersion. *Atmospheric Environment*, 105:130–137, 2015.

[94] Yusheng Jiao, Feng Ling, Sina Heydari, Nicolas Heess, Josh Merel, and

Eva Kanso. Learning to swim in potential flow. *Physical Review Fluids*, 6(5):050505, 2021.

[95] Eloy Urendes Jiménez, Antonio Flores Caballero, Francisco Molina Rueda, Javier Iglesias Giménez, and Roberto Oboe. Reverse-engineer the brain: Perspectives and challenges. In *Emerging Therapies in Neurorehabilitation*, pages 173–188. Springer, 2014.

[96] Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285, 1996.

[97] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society A*, 474(2219):20180335, 2018.

[98] Kiyosi Kawanisi and Ryohei Shiozaki. Turbulent effects on the settling velocity of suspended sediment. *Journal of Hydraulic Engineering*, 134(2):261–266, 2008.

[99] Klara Kihlström, Brett Aiello, Eric Warrant, Simon Sponberg, and Anna Stöckl. Wing damage affects flight kinematics but not flower tracking performance in hummingbird hawkmoths. *Journal of Experimental Biology*, 224(4):jeb236240, 2021.

[100] MAR Koehl and T Cooper. Swimming in an unsteady world. *Integrative and Comparative Biology*, 55(4):683–697, 2015.

[101] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in neural information processing systems*, 12, 1999.

[102] Lubomir Kostal, Petr Lansky, and Jean-Pierre Rospars. Neuronal coding and spiking randomness. *European Journal of Neuroscience*, 26(10):2693–2701, 2007.

[103] Shinsuke Koyama, Uri T Eden, Emery N Brown, and Robert E Kass. Bayesian decoding of neural spike trains. *Annals of the Institute of Statistical Mathematics*, 62(1):37–59, 2010.

[104] A Kronenburg, RW Bilger, and JH Kent. Modeling soot formation in turbulent methane–air jet diffusion flames. *Combustion and Flame*, 121(1-2):24–40, 2000.

[105] Jack W Langelaan. Gust energy extraction for mini and micro uninhabited aerial vehicles. *Journal of Guidance, Control, and Dynamics*, 32(2):464–473, 2009.

[106] Jack W Langelaan, Nicholas Alley, and James Neidhoefer. Wind field estimation for small unmanned aerial vehicles. *Journal of Guidance, Control, and Dynamics*, 34(4):1016–1030, 2011.

[107] Irving Langmuir. Surface motion of water induced by wind. *Science*, 87(2250):119–123, 1938.

[108] Kasey M Laurent, Bob Fogg, Tobias Ginsburg, Casey Halverson, Michael J Lanzone, Tricia A Miller, David W Winkler, and Gregory P Bewley. Turbulence explains the accelerations of an eagle in natural flight. *Proceedings of the National Academy of Sciences*, 118(23), 2021.

[109] Jonggu Lee, Seungwan Ryu, and H Jin Kim. Stable flight of a flapping-wing micro air vehicle under wind disturbance. *IEEE Robotics and Automation Letters*, 5(4):5685–5692, 2020.

[110] Edward R Levine and Rolf G Lueck. Turbulence measurement from an autonomous underwater vehicle. *Journal of atmospheric and oceanic technology*, 16(11):1533–1544, 1999.

[111] Lin Li, Il Memming Park, Sohan Seth, John S Choi, Joseph T Francis, Justin C Sanchez, and José C Príncipe. An adaptive decoder from spike trains to micro-stimulation using kernel least-mean-squares (klms). In *2011 IEEE International Workshop on Machine Learning for Signal Processing*, pages 1–6. IEEE, 2011.

[112] Theodore Lindsay, Anne Sustar, and Michael Dickinson. The function and organization of the motor system controlling flight maneuvers in flies. *Current Biology*, 27(3):345–358, 2017.

[113] Dernis J Linse and Robert F Stengel. Neural networks for function approximation in nonlinear control. In *1990 American Control Conference*, pages 674–681. IEEE, 1990.

[114] Jesus L Lobo, Javier Del Ser, Albert Bifet, and Nikola Kasabov. Spiking neural networks and online learning: An overview and perspectives. *Neural Networks*, 121:88–100, 2020.

[115] Edward N Lorenz. Deterministic nonperiodic flow. *Journal of the atmospheric sciences*, 20(2):130–141, 1963.

[116] Jan Lunze and Jan H Richter. Reconfigurable fault-tolerant control: A tutorial introduction. *European journal of control*, 14(5):359–386, 2008.

[117] Kevin Y Ma, Pakpong Chirarattananon, Sawyer B Fuller, and Robert J Wood. Controlled flight of a biologically inspired, insect-scale robot. *Science*, 340(6132):603–607, 2013.

[118] Kevin Y Ma, Samuel M Felton, and Robert J Wood. Design, fabrication, and modeling of the split actuator microrobotic bee. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1133–1140. IEEE, 2012.

[119] Dana Mackenzie. A flapping of wings. *Science*, 335(6075):1430–1433, 2012.

[120] David MacNeil and Chris Eliasmith. Fine-tuning and the stability of recurrent neural networks. *PloS one*, 6(9):e22885, 2011.

[121] S MahmoudZadeh, D Powers, K Sammut, and AM Yazdani. Differential evolution for efficient auv path planning in time variant uncertain underwater environment. *Robotics (cs. RO)*, 2016.

[122] James AR Marshall, Rafal Bogacz, Anna Dornhaus, Robert Planqué, Tim Kovacs, and Nigel R Franks. On optimal decision-making in brains and social insect colonies. *Journal of the Royal Society Interface*, 6(40):1065–1074, 2009.

[123] Martin R Maxey and James J Riley. Equation of motion for a small rigid sphere in a nonuniform flow. *The Physics of Fluids*, 26(4):883–889, 1983.

[124] MR Maxey. The gravitational settling of aerosol particles in homogeneous turbulence and random flow fields. *Journal of Fluid Mechanics*, 174:441–465, 1987.

[125] MR Maxey. The motion of small spherical particles in a cellular flow field. *The Physics of Fluids*, 30(7):1915–1928, 1987.

[126] MR Maxey and S Corrsin. Gravitational settling of aerosol particles in randomly oriented cellular flow fields. *Journal of the Atmospheric Sciences*, 43(11):1112–1134, 1986.

[127] Timothy G McGee and J Karl Hedrick. Path planning and control for multiple point surveillance by an unmanned aircraft in wind. In *2006 American Control Conference*, pages 4261–4266. IEEE, 2006.

[128] Rebecca McGill, Nak-seung Patrick Hyun, and Robert J Wood. Modeling and control of flapping-wing micro-aerial vehicles with harmonic sinusoids. *IEEE Robotics and Automation Letters*, 7(2):746–753, 2021.

[129] K Mikkola. Direction of insect migrations in relation to the wind. In *Insect Flight*, pages 152–171. Springer, 1986.

[130] Richard M Murray et al. Optimization-based control. *California Institute of Technology, CA*, pages 111–128, 2009.

[131] P Hyun Nak-seung, Rebecca McGill, Robert J Wood, and Scott Kuindersma. A new control framework for flapping-wing vehicles based on 3d pendulum dynamics. *Automatica*, 123:109293, 2021.

[132] Peter Nielsen. Turbulence effects on the settling of suspended particles. *Journal of Sedimentary Research*, 63(5):835–838, 1993.

[133] Sheila Nirenberg and Peter E Latham. Decoding neuronal spike trains: how important are correlations? *Proceedings of the National Academy of Sciences*, 100(12):7348–7353, 2003.

[134] Bernd R Noack, Konstantin Afanasiev, MAREK MORZYŃSKI, Gilead Tadmor, and Frank Thiele. A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics*, 497:335–363, 2003.

[135] Stephen M Nogar, Andrea Serrani, Abhijit Gogulapati, Jack J McNamara, Michael W Oppenheimer, and David B Doman. Design and evaluation of a model-based controller for flapping-wing micro air vehicles. *Journal of Guidance, Control, and Dynamics*, 41(12):2513–2528, 2018.

[136] Christopher T Orlowski and Anouck R Girard. Dynamics, stability, and control analyses of flapping wing micro-air vehicles. *Progress in Aerospace Sciences*, 51:18–30, 2012.

[137] António RC Paiva, Il Park, and Jose C Principe. A reproducing kernel hilbert space framework for spike train signal processing. *Neural Computation*, 21(2):424–449, 2009.

[138] Kundan Panta, Joseph Gramignano, Trevor Moser, Bo Cheng, and Azar Eslam-Panah. The interaction between a plunging wing and gusty environment. In *APS Division of Fluid Dynamics Meeting Abstracts*, pages E25–004, 2021.

[139] Il Park and José C Príncipe. Quantification of inter-trial non-stationarity in spike trains from periodically stimulated neural cultures. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5442–5445. IEEE, 2010.

[140] Il Memming Park, Sohan Seth, Antonio RC Paiva, Lin Li, and Jose C Principe. Kernel methods on spike train space for neuroscience: a tutorial. *IEEE Signal Processing Magazine*, 30(4):149–160, 2013.

[141] Shinkyu Park, Yaofeng Desmond Zhong, and Naomi Ehrich Leonard. Multi-robot task allocation games in dynamically changing environments. *2021 International Conference on Robotics and Automation (ICRA)*, 2021.

[142] Ron J Patton. Fault-tolerant control: The 1997 situation. *IFAC Proceedings Volumes*, 30(18):1029–1051, 1997.

[143] Colin J Pennycuick. Thermal soaring compared in three dissimilar tropical bird species, fregata magnificens, pelecanus occidentals and coragyps atratus. *Journal of Experimental Biology*, 102(1):307–325, 1983.

[144] Hoang Vu Phan and Hoon Cheol Park. Insect-inspired, tailless, hover-capable flapping-wing robots: Recent progress, challenges, and future directions. *Progress in Aerospace Sciences*, 111:100573, 2019.

[145] Sharon A Poessel, Joseph Brandt, Tricia A Miller, and Todd E Katzner. Meteorological and environmental variables affect flight behaviour and decision-making of an obligate soaring bird, the california condor gymnogyps californianus. *Ibis*, 160(1):36–53, 2018.

[146] Stephen B Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[147] James A Preiss, Wolfgang Hönig, Nora Ayanian, and Gaurav S Sukhatme. Downwash-aware trajectory planning for large quadrotor teams. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 250–257. IEEE, 2017.

[148] James A Preiss, Wolfgang Honig, Gaurav S Sukhatme, and Nora Ayanian.

Crazyswarm: A large nano-quadcopter swarm. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3299–3304. IEEE, 2017.

[149] Hans R Pruppacher and James D Klett. Microstructure of atmospheric clouds and precipitation. In *Microphysics of Clouds and Precipitation*, pages 9–55. Springer, 1978.

[150] Ramesh PS and Muruga Lal Jeyan. Mini unmanned aerial systems (uav)-a review of the parameters for classification of a mini uav. *International Journal of Aviation, Aeronautics, and Aerospace*, 7(3):5, 2020.

[151] Mark L Psiaki and Robert F Stengel. Analysis of aircraft control strategies for microburst encounter. *Journal of Guidance, Control, and Dynamics*, 8(5):553–559, 1985.

[152] Mark L Psiaki and Robert F Stengel. Optimal flight paths through microburst wind profiles. *Journal of Aircraft*, 23(8):629–635, 1986.

[153] Joy Putney, Rachel Conn, and Simon Sponberg. Precise timing is ubiquitous, consistent, and coordinated across a comprehensive, spike-resolved flight motor program. *Proceedings of the National Academy of Sciences*, 116(52):26951–26960, 2019.

[154] Joy Putney, Rachel Conn, and Simon Sponberg. Precise timing is ubiquitous, consistent, and coordinated across a comprehensive, spike-resolved flight motor program. *Proceedings of the National Academy of Sciences*, 116(52):26951–26960, 2019.

[155] Joy Putney, Tobias Niebur, Rachel Conn, and Simon Sponberg. An information theoretic method to resolve millisecond-scale spike timing precision in a comprehensive motor program. *bioRxiv*, 2021.

[156] P Randeni, AT Supun, Alexander L Forrest, Remo Cossu, Zhi Q Leong, Dev Ranmuthugala, et al. Determining the horizontal and vertical water velocity components of a turbulent water column using the motion response of an autonomous underwater vehicle. *Journal of Marine Science and Engineering*, 5(3):25, 2017.

[157] Gautam Reddy, Antonio Celani, Terrence J Sejnowski, and Massimo Vergassola. Learning to soar in turbulent environments. *Proceedings of the National Academy of Sciences*, 113(33):E4877–E4884, 2016.

[158] Gautam Reddy, Jerome Wong-Ng, Antonio Celani, Terrence J Sejnowski, and Massimo Vergassola. Glider soaring via reinforcement learning in the field. *Nature*, 562(7726):236, 2018.

[159] Amin Riazi and Umut Türker. The drag coefficient and settling velocity of natural sediment particles. *Computational Particle Mechanics*, pages 1–11, 2019.

[160] Stefan Ristevski, Ahmet T Koru, Tansel Yucelen, Kadriye Merve Dogan, and Jonathan A Muse. Experimental results of a quadrotor uav with a model reference adaptive controller in the presence of unmodeled dynamic. In *AIAA SCITECH 2022 Forum*, page 1381, 2022.

[161] Leif Ristroph, Gunnar Ristroph, Svetlana Morozova, Attila J Bergou, Song Chang, John Guckenheimer, Z Jane Wang, and Itai Cohen. Active and passive stabilization of body pitch in insect flight. *Journal of The Royal Society Interface*, 10(85):20130237, 2013.

[162] Allan R Robinson. Forecasting and simulating coastal ocean processes and variabilities with the harvard ocean prediction system. *Coastal ocean prediction*, pages 77–100, 1999.

[163] Allan R Robinson, Hernan G Arango, Alex Warn-Varnas, Wayne G Leslie, Arthur J Miller, Patrick J Haley, and Carlos J Lozano. Real-time regional forecasting. In *Elsevier oceanography series*, volume 61, pages 377–410. Elsevier, 1996.

[164] AR Robinson. Physical processes, field estimation and an approach to interdisciplinary ocean modeling. *Earth-Science Reviews*, 40(1-2):3–54, 1996.

[165] Eatai Roth, Robert W Hall, Thomas L Daniel, and Simon Sponberg. Integration of parallel mechanosensory and visual pathways resolved through sensory conflict. *Proceedings of the National Academy of Sciences*, 113(45):12832–12837, 2016.

[166] Seungwan Ryu and H Jin Kim. Development of a flapping-wing micro air vehicle capable of autonomous hovering with onboard measurements. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3239–3245. IEEE, 2017.

[167] Emilio Salinas and LF Abbott. Vector reconstruction from firing rates. *Journal of Computational Neuroscience*, 1(1-2):89–107, 1994.

[168] Bernhard Schölkopf, Ralf Herbrich, and Alex J Smola. A generalized representer theorem. In *International Conference on Computational Learning Theory*, pages 416–426. Springer, 2001.

[169] Benjamin Schrauwen and Jan Van Campenhout. Linking non-binned spike train kernels to several existing spike train metrics. *Neurocomputing*, 70(7-9):1247–1253, 2007.

[170] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions. *Journal of Mathematical Psychology*, 85:1–16, 2018.

[171] John M Seddon and Simon Newman. *Basic helicopter aerodynamics*. John Wiley & Sons, 2011.

[172] Terrence J Sejnowski et al. Time for a new neural code? *Nature*, 376(6535):21–22, 1995.

[173] Hazim Shakhatreh, Ahmad H Sawalmeh, Ala Al-Fuqaha, Zuochao Dou, Eyad Almaita, Issa Khalil, Noor Shamsiah Othman, Abdallah Khreishah, and Mohsen Guizani. Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges. *IEEE Access*, 7:48572–48634, 2019.

[174] Jeff S Shamma and Michael Athans. Guaranteed properties of gain scheduled control for linear parameter-varying plants. *Automatica*, 27(3):559–564, 1991.

[175] Jennie Si, Andrew G Barto, Warren B Powell, and Don Wunsch. *Handbook of learning and approximate dynamic programming*, volume 2. John Wiley & Sons, 2004.

[176] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[177] David Sigthorsson, Michael Oppenheimer, and David Doman. Insect sized flapping wing vehicles versus rotorcrafts, a comparative study. In *50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*, page 28, 2012.

[178] Ryan N Smith, Yi Chao, Peggy P Li, David A Caron, Burton H Jones, and Gaurav S Sukhatme. Planning and implementing trajectories for au-

tonomous underwater vehicles to track evolving ocean processes based on predictions from a regional ocean model. *The International Journal of Robotics Research*, 29(12):1475–1497, 2010.

[179] Joshua Solberg. Susceptibility of quadcopter flight to turbulence. Master's thesis, Sibley School of Mechanical and Aerospace Engineering, Cornell University, 2018.

[180] Jialei Song, Bret W Tobalske, Donald R Powers, Tyson L Hedrick, and Haoxiang Luo. Three-dimensional simulation for fast forward flight of a calliope hummingbird. *Royal Society open science*, 3(6):160230, 2016.

[181] Jialei Song, Yong Zhong, Haoxiang Luo, Yang Ding, and Ruxu Du. Hydrodynamics of larval fish quick turning: a computational study. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 232(14):2515–2523, 2018.

[182] Zhuoyuan Song and Kamran Mohseni. Facon: A flow-aided cooperative navigation scheme. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6251–6256. IEEE, 2017.

[183] Simon Sponberg, Thomas L Daniel, and Adrienne L Fairhall. Dual dimensionality reduction reveals independent encoding of motor features in a muscle synergy for insect flight control. *PLoS Computational Biology*, 11(4):e1004168, 2015.

[184] Simon Sponberg, Jonathan P Dyhr, Robert W Hall, and Thomas L Daniel. Luminance-dependent visual processing enables moth flight in low light. *Science*, 348(6240):1245–1248, 2015.

[185] Marc Steinberg, Jason Stack, and Terri Paluszkiewicz. Long duration autonomy for maritime systems: Challenges and opportunities. *Autonomous Robots*, 40(7):1119–1122, 2016.

[186] Robert F Stengel. Intelligent failure-tolerant control. *IEEE Control Systems Magazine*, 11(4):14–23, 1991.

[187] Robert F Stengel. Toward intelligent flight control. *IEEE transactions on Systems, Man, and Cybernetics*, 23(6):1699–1717, 1993.

[188] Robert F Stengel. *Optimal Control and Estimation*. Courier Corporation, 1994.

[189] Robert F Stengel. *Optimal control and estimation*. Courier Corporation, 1994.

[190] Terrence C Stewart and Chris Eliasmith. Large-scale synthesis of functional spiking neural circuits. *Proceedings of the IEEE*, 102(5):881–898, 2014.

[191] George Gabriel Stokes et al. On the effect of the internal friction of fluids on the motion of pendulums. *Pitt Press Cambridge*, 1851.

[192] Henry Stommel. Trajectories of small bodies sinking slowly through convection cells. *Journal of Marine Research*, 8(11):24–29, 1949.

[193] DAlexander Stratton and Roberts Stengel. Real-time decision aiding: aircraft guidance for wind shear avoidance. *IEEE Transactions on Aerospace and Electronic Systems*, 31(1):117–124, 1995.

[194] Jiabi Sun, Jin Song, Haoyao Chen, Xiaopeng Huang, and Yunhui Liu. Autonomous state estimation and mapping in unknown environments with onboard stereo camera for micro aerial vehicles. *IEEE Transactions on Industrial Informatics*, 16(9):5746–5756, 2019.

[195] Richard S Sutton and Andrew G Barto. A temporal-difference model of classical conditioning. In *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, pages 355–378. Seattle, WA, 1987.

[196] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[197] Nitin Sydney, Brendan Smyth, and Derek A Paley. Dynamic control of autonomous quadrotor flight in an estimated wind field. In *52nd IEEE Conference on Decision and Control*, pages 3609–3616. IEEE, 2013.

[198] Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, Georgina Cosma, Liam P Maguire, and T Martin McGinnity. A review of learning in biologically plausible spiking neural networks. *Neural Networks*, 122:253–272, 2020.

[199] Gang Tao, Shuhao Chen, Xidong Tang, and Suresh M Joshi. *Adaptive control of systems with actuator failures*. Springer Science & Business Media, 2004.

[200] Josin Tom and Andrew D Bragg. Multiscale preferential sweeping of particles settling in turbulence. *Journal of Fluid Mechanics*, 871:244–270, 2019.

[201] AAR Townsend. *The structure of turbulent shear flow*. Cambridge university press, 1980.

[202] Nguyen Khoi Tran, Eitan Bulka, and Meyer Nahon. Quadrotor control in a wind field. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 320–328. IEEE, 2015.

[203] Valérie Ventura. Spike train decoding without spike sorting. *Neural Computation*, 20(4):923–963, 2008.

[204] Lian-Ping Wang and Martin R Maxey. Settling velocity and concentration distribution of heavy particles in homogeneous isotropic turbulence. *Journal of Fluid Mechanics*, 256:27–68, 1993.

[205] Xiangwen Wang, Xianghong Lin, and Xiaochao Dang. Supervised learning in spiking neural networks: A review of algorithms and evaluations. *Neural Networks*, 125:258–280, 2020.

[206] Z Jane Wang. Insect flight: from newton's law to neurons. *Annual Review of Condensed Matter Physics*, 7:281–300, 2016.

[207] John P Wangermann and Robert F Stengel. Optimization and coordination of multiagent systems using principled negotiation. *Journal of Guidance, Control, and Dynamics*, 22(1):43–50, 1999.

[208] Steven Waslander and Carlos Wang. Wind disturbance estimation and rejection for quadrotor position control. In *AIAA Infotech@ Aerospace conference and AIAA unmanned... Unlimited conference*, page 1983, 2009.

[209] Hiroaki Watanabe, Ryoichi Kurose, Satoru Komori, and Heinz Pitsch. Effects of radiation on spray flame characteristics and soot formation. *Combustion and Flame*, 152(1-2):2–13, 2008.

[210] Stephan Weiss, Markus W Achtelik, Simon Lynen, Margarita Chli, and Roland Siegwart. Real-time onboard visual-inertial state estimation and self-calibration of mavs in unknown environments. In *2012 IEEE international conference on robotics and automation*, pages 957–964. IEEE, 2012.

[211] Liyan Wen, Gang Tao, Hao Yang, and Yi Yang. Aircraft turbulence compensation using adaptive multivariable disturbance rejection techniques. *Journal of Guidance, Control, and Dynamics*, 38(5):954–963, 2015.

[212] Christopher K Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[213] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

[214] W. M. Wonham and C. D. Johnson. Optimal bang-bang control with quadratic performance index. *Journal of Basic Engineering*, 86(1):107–115, 03 1964.

[215] Robert J Wood. Design, fabrication, and analysis of a 3dof, 3cm flapping-wing mav. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1576–1581. IEEE, 2007.

[216] Timothy D Woodbury, Caroline Dunn, and John Valasek. Autonomous soaring using reinforcement learning for trajectory generation. In *52nd Aerospace Sciences Meeting*, page 0990, 2014.

[217] Xingyu Xiang, Zhonghai Wang, Zijian Mo, Genshe Chen, Khanh Pham, and Erik Blasch. Wind field estimation through autonomous quadcopter avionics. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–6. IEEE, 2016.

[218] Shengjie Xiao, Kai Hu, Binxiao Huang, Huichao Deng, and Xilun Ding. A review of research on the mechanical design of hoverable flapping wing micro-air vehicles. *Journal of Bionic Engineering*, pages 1–20, 2021.

[219] Jianguo Xin and Mark J Embrechts. Supervised learning with spiking neural networks. In *Proceedings of the International Joint Conference on Neural Networks*, volume 3, pages 1772–1777. IEEE, 2001.

[220] Hengye Yang. Gravitational settling of inertial particles in a cellular flow. `https://youtu.be/o2OF5Et1opA`, August 2021.

[221] Hengye Yang. Hovering flight of an insect-scale flapping-wing robot with wing damage. `https://youtu.be/2-rcJG4ffXg`, October 2022.

[222] Hengye Yang. Square trajectory following of an insect-scale flapping-wing robot with state measurement error. https://youtu.be/ABREQB_Nzd4, October 2022.

[223] Hengye Yang, Gregory Bewley, and Silvia Ferrari. A fast-tracking-particle-inspired flow-aided control approach for air vehicles in turbulent flow. *Biomimetics*, 7(4):192, 2022.

[224] Hengye Yang, Dongheng Jing, Vahid Tarokh, Gregory Bewley, and Silvia Ferrari. Flow parameter estimation based on on-board measurements of air vehicle traversing turbulent flows. In *AIAA Scitech 2021 Forum*, page 0380, 2021.

[225] Hengye Yang, Saksham Kaushik, Taylor Clawson, Usama Bin Sikandar, Simon Sponberg Sponberg, Robert J Wood, and Silvia Ferrari. Adaptive full-envelope spiking neural network control of flapping-wing micro aerial vehicles. *Journal of Guidance, Control, and Dynamics*, to be submitted.

[226] Hengye Yang, Joy Putney, Usama Bin Sikandar, Pingping Zhu, Simon Sponberg, and Silvia Ferrari. A relative spike-timing approach to kernel-based decoding demonstrated for insect flight experiments. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2022.

[227] Edwin Yaz and Asok Ray. Linear unbiased state estimation for random models with sensor delay. In *Proceedings of 35th IEEE Conference on Decision and Control*, volume 1, pages 47–52. IEEE, 1996.

[228] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE Access*, 8:58443–58469, 2020.

[229] Somaiyeh Mahmoud Zadeh, David M.W Powers, Amir Mehdi Yazdani, Karl Sammut, and Adham Atyabi. Differential evolution for efficient auv path planning in time variant uncertain underwater environment. *arXiv preprint arXiv:1604.02523*, 2016.

[230] Chao Zhang and Claudio Rossi. A review of compliant transmission mechanisms for bio-inspired flapping-wing micro air vehicles. *Bioinspiration & biomimetics*, 12(2):025005, 2017.

[231] Xu Zhang, Greg Foderaro, Craig Henriquez, and Silvia Ferrari. A scalable weight-free learning algorithm for regulatory control of cell activ-

ity in spiking neuronal networks. *International Journal of Neural Systems*, 28(02):1750015, 2018.