DISSERTATION

BIOLOGICALLY INSPIRED PERCHING FOR AERIAL ROBOTS

Submitted by

Haijie Zhang

Department of Mechanical Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Spring 2021

Doctoral Committee:

Advisor: Jianguo Zhao

Thomas H. Bradley
Sudeep Pasricha
Stephen Guzik

ABSTRACT

BIOLOGICALLY INSPIRED PERCHING FOR AERIAL ROBOTS

Micro Aerial Vehicles (MAVs) are widely used for various civilian and military applications (e.g., surveillance, search, and monitoring, etc.); however, one critical problem they are facing is the limited airborne time (less than one hour) due to the low aerodynamic efficiency, low energy storage capability, and high energy consumption. To address this problem, mimicking biological flyers to perch onto objects (e.g., walls, power lines, or ceilings) will significantly extend MAVs' functioning time for surveillance or monitoring related tasks. Successful perching for aerial robots, however, is quite challenging as it requires a synergistic integration of mechanical and computational intelligence. Mechanical intelligence means mechanical mechanisms to passively damp out the impact between the robot and the perching object and robustly engage the robot to the perching objects. Computational intelligence means computation algorithms to estimate, plan, and control the robot's motion so that the robot can progressively reduce its speed and adjust its orientation to perch on the objects with a desired velocity and orientation.

In this research, a framework for biologically inspired perching is investigated, focusing on both computational and mechanical intelligence. Computational intelligence includes vision-based state estimation and trajectory planning. Unlike traditional flight states such as position and velocity, we leverage a biologically inspired state called time-to-contact (TTC) that represents the remaining time to the perching object at the current flight velocity. A faster and more accurate estimation method based on consecutive images is proposed to estimate TTC. Then a trajectory is planned in TTC space to realize the faster perching while satisfying multiple flight and perching constraints, e.g., maximum velocity, maximum acceleration, and desired contact velocity. For mechanical intelligence, we design, develop, and analyze a novel compliant bistable gripper with two stable states. When the gripper is open, it can close passively by the contact force between

the robot and the perching object, eliminating additional actuators or sensors. We also analyze the bistability of the gripper to guide and optimize the design of the gripper. At the end, a customized MAV platform of size 250 mm is designed to combine computational and mechanical intelligence. A Raspberry Pi is used as the onboard computer to do vision-based state estimation and control. Besides, a larger gripper is designed to make the MAV perch on a horizontal rod. Perching experiments using the designed trajectories perform well at activating the bistable gripper to perch while avoiding large impact force which may damage the gripper and the MAV. The research will enable robust perching of MAVs so that they can maintain a desired observation or resting position for long-duration inspection, surveillance, search, and rescue.

ACKNOWLEDGEMENTS

The author would like to thank his advisor, Dr. Jianguo Zhao and the rest of his graduate committee: Dr. Thomas H. Bradley, Dr. Sudeep Pasricha and Dr. Stephen Guzik for the time and expertise they lent while guiding his graduate research efforts at Colorado State University.

DEDICATION

I would like to dedicate this thesis to my wife, my daughter and my parents.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background

Recent years have witnessed the growing popularity of Micro Aerial Vehicles (MAVs) in recreational, scientific, and military applications. However, MAVs, especially those with multiple rotors, are facing a common critical problem: limited flight time. The reason is that the Renolds number (R_e) decreases with the flyer's size. Renolds number R_e is proportional to the flight velocity and chord length. And flight velocity is proportional to the square root of length [1]. Thus, the smaller and slower the MAV is, the lower Renolds number it operates at. Furthermore, a lower Renolds number will induce a smaller maximum lift coefficient C_L and a larger drag coefficient C_D [2, 3]. However, glide ratio C_L/C_D determines flight distance and $C_L^{1.5}/C_D$ determines flight time. On the other hand, energy storage and conversion also suffer at small scales. Batteries are the most common power supply for MAVs. However, the energy density of the battery is only about $0.15\,\mathrm{kW\,h/kg}$, while large aircraft fuels is about $12\,\mathrm{kW\,h/kg}$ [4]. As shown in Table 1.1, the flight time for commercial MAVs is usually less than 30 minutes.

Table 1.1: 10 commercial MAVs with longest flight time

Product	Flight time	Product dimensions	Weight
Autel Robotics EVO Drone	30min	7.8 x 3.8 x 4 inches	1.9 pounds
Sim Too Pro	30min	NA	5.95 pounds
DJI Phatom 4	28min	15 x 8.7 x 12.8 inches	8.82 pounds
DJI Mavic Pro	27min	12 x 12 x 12 inches	10 pounds
DJI Inspire 2	27min	59.1 x 59.1 x 39.4 inches	30 pounds
Parrot Bebop 2	25min	15 x 3.5 x 12.9 inches	1.1 pounds
DJI Phantom 3 Standard	25min	15 x 14 x 8.2 inches	8.2 pounds
DJI Phantom 3 Pro	23min	18 x 13 x 8 inches	9.2 pounds
3DR Solo	22min	18 x 18 x 10 inches	3.3 pounds
Yuneec Q500+	22min	22.2 x 16.5 x 9.4 inches	2.5 pounds

To address this problem, perching onto objects (e.g., walls, power lines, or ceilings) will significantly extend aerial robots' functioning time as they can save or even harvest energy after perching, while also maintaining a desired altitude and orientation for surveillance or monitoring [5]. Successful perching for aerial robots, however, is quite challenging as it requires not only intelligent mechanical mechanisms to robustly engage the robot to the perching objects but also fast and accurate estimation, planning, and control of the robot motion so that the robot can progressively reduce its speed and adjust its orientation to perch on the objects with a desired velocity and orientation.

In recent years, researchers have investigated perching capabilities for aerial robots from both the mechanical and control aspects. A detailed review can be found in [5], and here we will only review some representative work. For mechanical investigations, the focus is on how to design robust perching mechanisms to ensure successful perching. Doyle *et al.* developed an integrated, compliant, and underactuated gripping foot as well as a collapsing leg mechanism to enable a quadcopter to passively perch on the surface with moderate disturbances [6]. Daler *et al.* designed a new perching mechanism based on a compliant deployable pad and a passive self-alignment system. With this mechanism, active control during final touch down is not needed [7]. Pope *et al.* designed a mechanism to make a quadcopter fly, perch passively onto outdoor surfaces, climb, and take off again [8]. Graule *et al.* utilized controllable electrostatic adhesion to make a robotic insect perch and take off from surfaces made of various materials [9]. Kovac *et al.* designed a $4.6\,\mathrm{g}$ perching mechanism which allows UAVs to perch on vertical surface of natural and man-made materials [10]. For investigations from the control and planning aspect, researchers have focused on how to generate and track flight trajectories for perching. Moore *et al.* utilized linear quadratic regulator trees to plan and track trajectory for fixed-wing aircrafts to perch on power lines [11]. Mellinger *et al.* designed a trajectory for quadcopter aggressive maneuvers to realize flights through narrow gaps and perching on inverted surfaces [12]. They also controlled quadcopters to perch on inclined surfaces with a downward-facing gripper [13]. Mohta *et al.* leveraged visual servoing with two known points on the target surface to achieve perching using feedback from a monocular camera and an inertial measurement unit (IMU) [14]. In [15], a laser

sensor is used to detect the perching initiation distance, and a pitch up process is used to assist the deceleration. Further, in [16], they applied thrust after pitch up which reduces the timing and sensing requirement for the perching triggering.

However, it's nontrivial to accomplish reliable and robust perching. It requires both computational and mechanical intelligence. For the computational intelligence, first of all, the MAVs need to know their flight states (e.g. position, velocity, and orientation). GPS, LIDAR, and distance sensor are commonly used devices for estimating flight states. However, the GPS is not able to work in the indoor environment, LIDAR is heavy for MAVs and detecting perching objects might be difficult for the distance sensor in some cases. Meanwhile, other than implementing different kinds of sensors, biological flyers can detect objects and estimate their flight states with their eyes. Thus, a potential method to estimate flight states is to use the camera. Compared with the previous sensors, the camera is able to work in the indoor environment and provide more information with light weight. However, a monocular camera cannot provide states such as position (distance) or velocity. Meanwhile, from biological study [17], bees use a visual information named time-to-contact (also known as TTC and tau) to safely land on objects. Inspired by the biological research, we use the time-to-contact as the flight state for aerial robot perching.

The second aspect of computational intelligence lies in how to utilize the estimated flight states to design a trajectory as a reference for the perching process. Currently, there are two widely used time-to-contact trajectory references, namely constant tau and constant tau dot strategy. If the MAV follows the two tau references, the contact velocity between the MAV and the goal object would be almost zero or zero, which we name it soft contact. However, soft contact is not ideal for perching tasks since perching mechanisms (e.g., adhesion pad, microspine) usually require a non-zero contact velocity to functionalize [10, 11, 13]. In this case, a trajectory in TTC space needs to be designed for non-zero contact velocity. In this dissertation, we propose a constant tau dot based two stage strategy (CTDTS) and an inverse polynomial based two state strategy (IPTS). Both strategies can realize non-zero contact velocity and IPTS is potentially to be used to satisfy more flight constraints.

The last challenge for perching is to design a mechanical intelligent mechanism, a robust gripper to engage to the perching objects. In this dissertation, we present a bistable gripper design, which can switch between a stable open state and another stable closed state. Such a design has two advantages for perching compared with existing methods [8]. On one hand, it can leverage the impact force during the perching to passively close, increasing the robustness of the mechanism and eliminating the requirement for a sensor to detect the impact and an actuator to close the gripper. On the other hand, the gripper does not require additional energy input to maintain the stable states, making it ideal for applications requiring long-duration monitoring or surveillance.

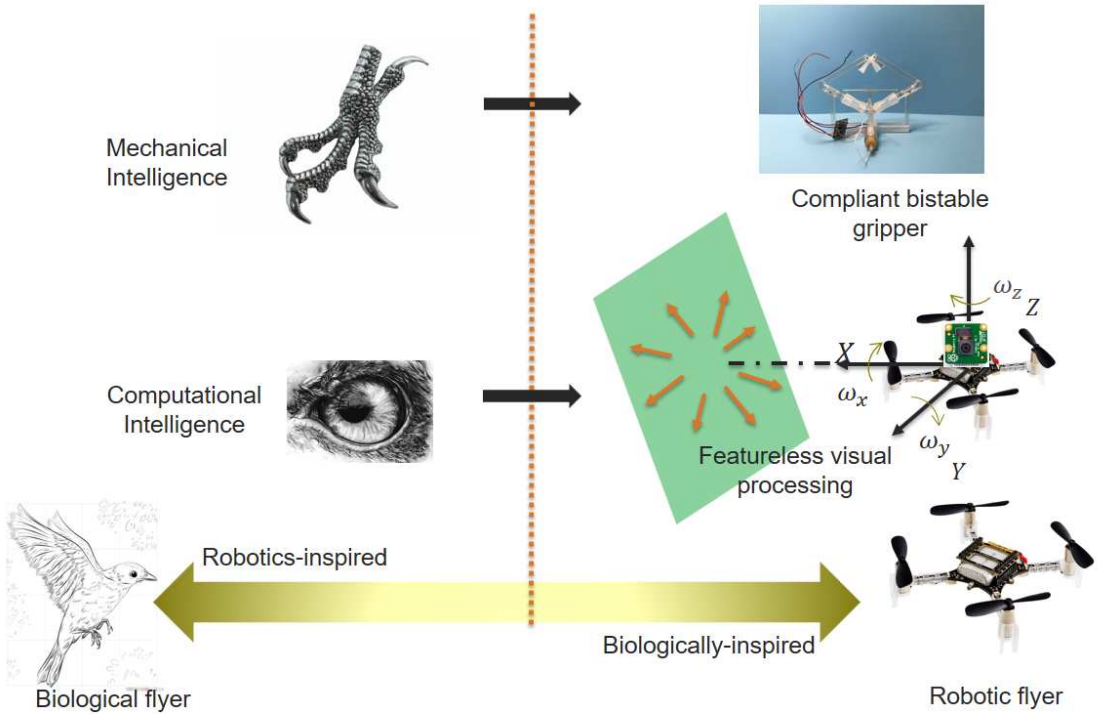


Figure 1.1: Mechanical intelligence and computational intelligence enable aerial robot perching

1.2 Outline and contributions of this dissertation

1.2.1 Outline

As shown in Figure 1.1, this dissertation focus on two main parts of aerial robot perching: computational and mechanical intelligence. Computational intelligence includes flight state estimation

and trajectory planning and mechanical intelligence details the compliant bistable gripper design. And in the end, a micro aerial vehicle is developed to integrate both computational and mechanical intelligence for aerial robot perching.

Chapter 2 first introduces the concept of time-to-contact and literature review on time-to-contact. Then an image based featureless TTC estimation method is introduced. As an extension to this featureless method, we consider angular velocities in the estimation method. A comparison between feature based method and featureless method is provided to show the advantages of the featureless method. And finally, several preliminary experiments using a mobile robot to estimate and control TTC with and without angular velocities are carried out.

Since the two widely used TTC references namely, the constant tau and constant tau dot strategy, cannot realize non-zero contact velocity, Chapter 3 provides CTDTS and IPTS for aerial robot perching. In this chapter, a palm-sized quadcopter, Crazyflie 2.0 is used to conduct the perching experiment. The TTC is estimated from a motion tracking system based on the definition of TTC: $TTC = X/V$, where X is the remaining distance to contact the object and V is the velocity. Experiments based on CTDTS and IPTS show that both strategies can realize no-zero contact velocity while IPTS can satisfy more flight constraints.

Chapter 4 introduces the bistable mechanism and the compliant bistable gripper as the mechanical intelligence. A mathematical model for the force-displacement characteristic of the gripper is provided and experiments are carried out to verify the accuracy of the model. In addition, an analysis for bistability is provided as a guideline for future bistable mechanism design. At the end, a series of perching experiments for both clipping and encircling method are conducted. The results show that with properly designed open and closing forces, the compliant bistable gripper is able to be used for aerial robot perching.

In the end, a customized MAV is designed in Chapter 5 to combine both computational intelligence and mechanical intelligence. A Raspberry Pi 3B+ is used as the onboard computer. The Raspberry Pi camera is used as the state estimation sensor. In addition, a large bistable gripper which is assembled with 3D printed Polylactic acid (PLA) parts and shape memory alloys (SMAs)

is designed as the perching mechanism for the larger customized MAV. The proposed CTDTS and IPTS are used as reference trajectories to control the MAV to perch on a horizontal rod.

1.2.2 Contributions

The overall research objective of this study is to enable aerial robots with perching capability based on biologically inspired information: time-to-contact.

The primary contributions to this field from this research are summarized below:

- Leveraged the featureless TTC estimation method to conduct the real-time robot motion control for the first time. Extended the featureless TTC estimation method when angular velocities exist during robot movement and verified that the expanded algorithm can achieve good estimation results.

- Proposed two novel perching trajectories (CTDTS and IPTS) to realize non-zero contact velocity in the TTC space, which solves the limitation of the two widely used TTC references since they can only achieve zero contact velocity. Besides, optimized the IPTS to achieve the fastest perching while satisfying different flight constraints.

- Expanded the usage of bistable mechanisms to the perching field by designing the bistable gripper for Crazyflie. And thoroughly analyze the bistability of the mechanism to generate a design guideline by selecting proper design parameters. Finally, Designed experiments to realize aerial robot perching and object grasping.

- Implemented an aerial robot platform equipped with the onboard computer, camera, and bistable gripper. Combined the computational intelligence and mechanical intelligence to enable the aerial robots with the autonomous perching capability.

Chapter 2

Computational Intelligence: State estimation

One important part of the MAV perching task is the feedback of the flight state. Unlike traditional flight states such as position, velocity, and orientation, we adopt time-to-contact (TTC) to estimate how much time is left to contact the goal object with the current velocity. In this chapter, first, we detail the time-to-contact background and related research, both the estimation algorithm and the control. Then we introduce a featureless computation method of time-to-contact. Based on that, we proposed a TTC estimation algorithm considering angular velocities. Several preliminary experiments are conducted: 1) The advantages of the featureless time-to-contact estimation algorithm compared with the feature-based method are verified. 2) Algorithm considering angular velocities accuracy is verified. 3) And at the end, the time-to-contact based mobile robot braking experiment is carried out to show the possibility of TTC being used as a flight state for aerial robot perching.

2.1 Introduction

In nature, various insects or animals rely on vision to control their motion to negotiate dynamic and uncertain environments. Insects can land on different surfaces such as ground or trees safely. Hummingbirds, paragons of precision flight, can brake to gently dock on a flower with pinpoint accuracy in a very short time [18]. Seabirds can adjust when to close wings before diving into the water for fish [19]. All of such elegant actions are accomplished only by animals' eyes, rather than distance sensors.

By analyzing the continuous images captured by eyes, insects and birds extract the so-called time-to-contact to guide their motion [20]. Time-to-contact (TTC) is defined as the time it would take a moving observer to make contact with an object or surface if the current velocity is maintained. Comprehensive experimental studies have shown that time-to-contact is a pervasive cue for animals' navigation. It has been discovered that bees keep a constant rate of image expan-

sion to perch on the flowers which is essentially a constant time-to-contact strategy [17]. Films of pigeons flying to a branch were analyzed, and the results showed that pigeons also adopt time-to-contact to perch [21]. By taking landing image sequences, fruit flies are also found to leverage the inverse of time-to-contact to control the landing and obstacle avoidance process [22]. Besides animals, drivers would also rely on time-to-contact to brake to avoid collision with obstacles or pedestrians [23].

TTC is a part of the more general tau theory originated from Gibson's research on the relationship between animals' visual information and their locomotion [24]. Based on Gibson's work, Lee first proposed the concept of TTC by pointing out that rather than distance or speed, drivers leverage TTC to determine when to accelerate or decelerate to drive safely [23]. Later, Lee introduced tau coupling [25] to guide motions in three-dimensional space simultaneously. The basic concepts of tau theory are as follows [26, 27]:

- A motion gap, denoted as $X(t)$, is the changing gap with respect to time t between the current state and the goal state. Motion gaps can be distance, force, angle, etc.

- Tau of a motion gap is the time to close this gap at its current closure rate $\dot{X}(t)$: $\tau = X(t)/\dot{X}(t)$. In the case with the gap being the distance, tau is the same as TTC. *In this chapter, we will use tau or TTC interchangeably since only the distance as a motion gap will be considered.*

- Tau-dot is the time derivative of tau. By maintaining a constant tau-dot, animals and insects can land or perch on surfaces with a full stop.

Inspired by biological studies, time-to-contact has been already applied to a variety of robotic platforms such as mobile robots and aerial robots to achieve landing, docking, chasing, obstacle avoidance, and navigation. For mobile robots, docking and obstacle avoidance are the two main tasks. Souhila *et al.* estimated time-to-contact from optic flow to navigate a mobile robot in cluttered indoor environment [28]. Kaneta *et al.* employed time-to-contact to avoid obstacles and chase another mobile robot, in which they obtained time-to-contact using the object size informa-

tion from consecutive images [29]. McCarthy *et al*. utilized time-to-contact to control the docking of a mobile robot in front of a vertical wall. They estimated time-to-contact using the divergence of optic flow from image sequences by considering the effects caused by the focus of expansion [30]. For aerial robot platforms, landing and navigation are the two main tasks. Kendoul proposed several time-to-contact controllers to control a quadcopter platform to realize the docking, landing, and navigation [31], of which the time-to-contact is estimated from GPS. To achieve safe landings, Izzo and Croon proposed different time-to-contact control methods [32]. They estimated time-to-contact from optical flow divergence and validated their control algorithm using a Parrot AR drone [33].

There are two widely used time-to-contact estimation methods: size based method [29, 34–37] and optic flow divergence based method [28, 32, 33, 38–40]. For the size based method, time-to-contact can be calculated from: $\tau = A/\dot{A}$, where A is size of a feature or object in the image and $\dot{A}$ is the time derivative of the size. To get the size of the objects, feature extraction and tracking are needed in successive images which is not only time-consuming but also a challenging problem in computer vision, especially in natural environments [34]. For the optic flow divergence based method [33], feature extraction and tracking are also needed for recovering optical flow to estimate the time-to-contact. Similarly, real time control is impeded by requirements for good features and time-consuming extraction and tracking processes.

Recently, the shift of pixel has been utilized to estimate the time-to-contact with good estimation results [41,42]. In addition, Horn *et al.* proposed a new method to estimate the time-to-contact without relying on feature extraction and tracking, and we call it featureless method in this chapter [43, 44]. This method directly manipulates all the intensities in two consecutive images based on the constant brightness assumption to estimate the time-to-contact. Without feature extraction and tracking, the computation time is shorter and the results are more reliable. Nevertheless, only a simplified case when a camera with linear velocities is considered. Further, the estimation method is not applied for the control of robots.

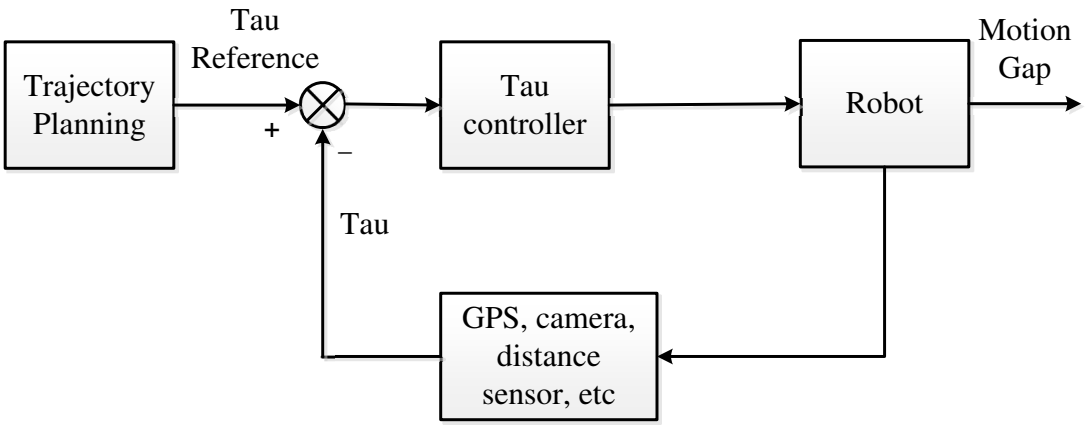


Figure 2.1: General idea for robot perching based on tau theory

In this chapter, we aim to exploit and extend the featureless direct method proposed by Horn *et al.* to estimate the time-to-contact, and then use the estimated time-to-contact to control the motion of mobile robots using the constant time-to-contact strategy [30] as shown in Figure 2.1. There are mainly three contributions in this chapter. First, we extend the featureless method to allow estimation for more general settings when angular velocities exist, which is ubiquitous for robotic platforms. Second, we improve the estimation results by using Kalman filtering on the estimated time-to-contact. Third, combining the estimation method and the constant tau theory, we design an error based controller with gain scheduling strategy to control the motion of a mobile robot for docking. The estimation and control methods presented in this chapter can be extended to other robotic platforms, especially for computation-constrained miniature robots [45, 46], for landing, docking, or navigation.

The rest of this chapter is organized as follows: section 2.2 describes the featureless method to estimate time-to-contact with and without angular velocities, Kalman filter and the control algorithm. Based on the estimation and control algorithms, section 2.3 details the experimental setup, results, and discussion. Section 2.4 concludes the chapter.

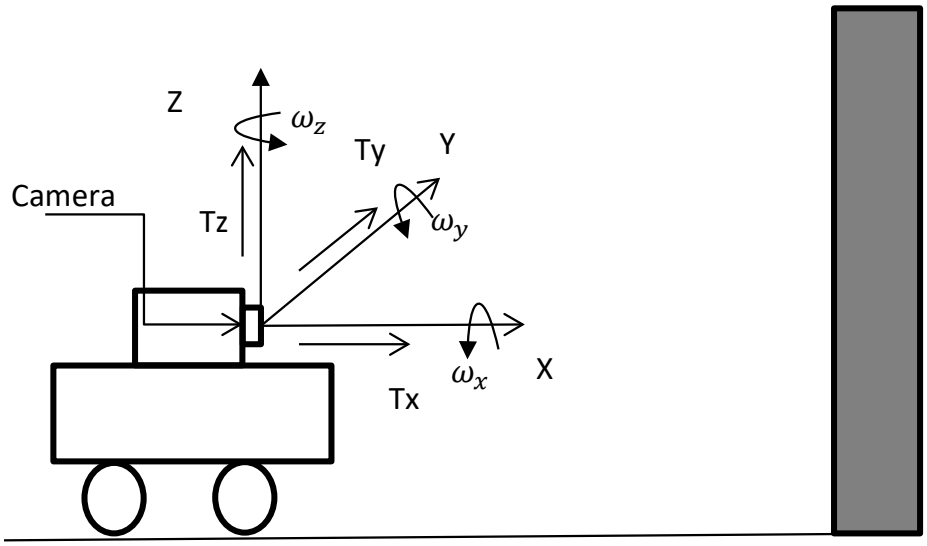


Figure 2.2: Docking experiment scenario

2.2 Featureless Based Control Method

In this section, the featureless method to estimate time-to-contact in [44] is introduced first. Then we extend this method by considering angular velocities. After that, Kalman filter is adopted to improve the performance of this algorithm. At the end, an error based proportional controller with gain scheduling is introduced.

Figure 2.2 shows a typical docking experiment scenario: a mobile robot carrying a camera moves toward a wall with translational velocities T_x, T_y, and T_z and angular velocities ω_x, ω_y, and ω_z. A camera frame is defined as follows: the origin is at the center of the image sensor, X axis along the optical axis, Y and Z axes parallel to the horizontal and vertical axis of the image sensor, respectively. A point with coordinates (X, Y, Z) in the camera frame is projected to the image frame with coordinates (x, y), where the image frame is a 2-dimensional coordinate frame in the image plane with the origin located at the principle point, and x and y axes parallel to the horizontal and vertical directions of the image plane, respectively. The goal of the braking is to let the robot progressively decrease its speed as it is approaching the wall to finally realize a soft contact with the wall.

2.2.1 Time-to-contact Without Angular Velocities

In computer vision, the well-known constant brightness assumption is the brightness I of the image point (x, y) at time t is constant, which can be written as follows [47]:

$$v_x I_x + v_y I_y + I_t = 0 \tag{2.1}$$

where $v_x = dx/dt$, $v_y = dy/dt$ are the optic flow vectors, $I_x = \partial I/\partial x$, $I_y = \partial I/\partial y$ are the brightness gradients, and $I_t = \partial I/\partial t$ is the rate of change for brightness with respect to time.

When the camera has both translational velocities and angular velocities as shown in Figure 2.2, we obtain the optic flow vector equation [48]:

$$\begin{cases} v_x = -\lambda\dfrac{T_y}{X} + x\dfrac{T_x}{X} + \omega_y\dfrac{xy}{\lambda} - \dfrac{\lambda^2 + x^2}{\lambda}\omega_z + y\omega_x \\ v_y = -\lambda\dfrac{T_z}{X} + y\dfrac{T_x}{X} - \omega_z\dfrac{xy}{\lambda} + \dfrac{\lambda^2 + y^2}{\lambda}\omega_y - x\omega_x \end{cases} \tag{2.2}$$

where λ is focal length. From Figure 2.2 we can know that time-to-contact can be calculated as:

$$\tau = \frac{X}{T_x} \tag{2.3}$$

Plugging the optic flow equation (2.2) into equation (2.1), we can solve the equation for time-to-contact, which can be classified into three cases in [43] if the angular velocities are not considered:

Case I

The robot moves along the optical axis which is perpendicular to an upright planar surface:

$$\tau = \frac{1}{C} = -\frac{\sum G^2}{\sum G I_t} \tag{2.4}$$

where $G = xI_x + yI_y$, $C = \frac{T_x}{X}$ and $\sum$ is an abbreviation of $\sum_{i=1}^{m}\sum_{j=1}^{n}$, where i, j are the pixel index in an $m \times n$ image. Unless otherwise stated, $\sum$ will have the same meaning in the rest of this chapter.

Case II

The robot translates in an arbitrary direction, but the optical axis is perpendicular to an upright planar surface:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y & \sum G I_x \\ \sum I_x I_y & \sum I_y^2 & \sum G I_y \\ \sum G I_x & \sum G I_y & \sum G^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \\ \sum G I_t \end{bmatrix} \tag{2.5}$$

where $A = -\lambda \frac{T_y}{X}$, $B = -\lambda \frac{T_z}{X}$.

Case III

The robot translates along the optical axis which is relative to an upright planar surface of arbitrary orientation:

$$\begin{bmatrix} \sum G^2 x^2 & \sum G^2 xy & \sum G^2 x \\ \sum G^2 xy & \sum G^2 y^2 & \sum G^2 y \\ \sum G^2 x & \sum G^2 y & \sum G^2 \end{bmatrix} \begin{bmatrix} P \\ Q \\ C \end{bmatrix} = - \begin{bmatrix} \sum G x I_t \\ \sum G y I_t \\ \sum G I_t \end{bmatrix} \tag{2.6}$$

where

$$P = -\frac{pT_x}{\lambda X_0} \quad Q = -\frac{qT_x}{\lambda X_0} \quad C = \frac{T_x}{X_0}$$

m and n are the slopes of planar surface in X and Y directions in camera frame and the surface equation can be written as

$$X = X_0 + pY + qZ \tag{2.7}$$

where X_0 is the intersection of the optical axis and the surface. In [44], the estimation algorithm for case II generates the best result when there is only translational movement. Therefore, we adopt the algorithm in case II for our docking experiments in section III.

2.2.2 Time-to-contact with Angular Velocities

The previous three cases assume there is no angular velocity for the camera, which is not true for general problems (an aerial robot with a camera is a typical example). As a result, we

need to extend the method to incorporate angular velocities. To include the angular velocities, we plug equation (2.2) into the constant brightness assumption equation and consider the surface in equation (2.7), leading to the following equation:

$$C(1+\frac{P}{C}x+\frac{Q}{C}y)(I_x\frac{A}{C}+I_y\frac{B}{C}+G)+JI_x+KI_y+I_t=0 \tag{2.8}$$

where U and V are the same with previous definition in case II. M, N and W are the same with previous definition in case III, and

$$\begin{cases} J=\dfrac{xy}{\lambda}\omega_y-\dfrac{\lambda^2+x^2}{\lambda}\omega_z+y\omega_x \\ K=\dfrac{\lambda^2+x^2}{\lambda}\omega_y-\dfrac{xy}{\lambda}\omega_z-x\omega_x \end{cases} \tag{2.9}$$

Note that with a gyroscope to measure ω_x, ω_y, and ω_z, we can know J and K for a certain image. The equation is similar with case IV when the robot is moving with an arbitrary trajectory and the orientation of the surface is arbitrary in [43]. We can formulate a least square problem to find the five unknown parameters U, V, M, N and W that minimize the following error sum over the interested image area:

$$\sum[C(1+\frac{P}{C}x+\frac{Q}{C}y)(I_x\frac{A}{C}+I_y\frac{B}{C}+G)+JI_x+KI_y+I_t]^2 \tag{2.10}$$

Letting $F=1+xP/C+yQ/C$ and $D=I_xA/C+I_yB/C+G$ yields

$$\sum[CFD+JI_x+KI_y+I_t]^2 \tag{2.11}$$

To solve the best values for the five unknown parameters, we differentiate the above sum with respect to the five parameters and set the results to zero. As a result, the solution is similar with the one stated in [43] which uses iterations to solve the nonlinear equation. Given an initial guess of P/C and Q/C, then F is known and we can solve for A, B and C using the following equation:

$$\begin{bmatrix} \sum F^2 I_x^2 & \sum F^2 I_x I_y & \sum F^2 I_x G \\ \sum F^2 I_x I_y & \sum F^2 I_y^2 & \sum F^2 I_y G \\ \sum F^2 I_x G & \sum F^2 I_y G & \sum F^2 G^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = - \begin{bmatrix} \sum F I_x I_\omega \\ \sum F I_y I_\omega \\ \sum F G I_\omega \end{bmatrix} \tag{2.12}$$

where $I_\omega = JI_x + KI_y + I_t$. Using the estimation of U, V and W, we can solve for the new M, N and W using

$$\begin{bmatrix} \sum D^2 x^2 & \sum D^2 xy & \sum D^2 x \\ \sum D^2 xy & \sum D^2 y^2 & \sum D^2 y \\ \sum D^2 x & \sum D^2 y & \sum D^2 \end{bmatrix} \begin{bmatrix} P \\ Q \\ C \end{bmatrix} = - \begin{bmatrix} \sum D x I_\omega \\ \sum D y I_\omega \\ \sum D I_\omega \end{bmatrix} \tag{2.13}$$

Based on the new P, Q and C, we can continue the iteration for new A, B and C. Eventually, a close approximation of time-to-contact can be obtained after several iterations.

For mobile robots, the case is simplified since the angular velocities ω_x, ω_y can be neglected due to relatively small rotation around X and Y axis compared to possible rotations around Z axis. Then we have $J = -\frac{\lambda^2 + x^2}{\lambda}\omega_z$ and $K = -\frac{xy}{\lambda}\omega_z$. Since in [43], case II gives the best results, here we use equation (2.5), and equation (2.5) can be rewritten as:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y & \sum G I_x \\ \sum I_x I_y & \sum I_y^2 & \sum G I_y \\ \sum G I_x & \sum G I_y & \sum G^2 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \end{bmatrix} = - \begin{bmatrix} \sum I_x I_\omega \\ \sum I_y I_\omega \\ \sum G I_\omega \end{bmatrix} \tag{2.14}$$

2.2.3 Kalman Filter

In the preliminary mobile robot docking experiments, the estimated time-to-contact might abruptly change due to the low quality camera and measurement errors from gyroscope, which caused the vibration of robot's speed [49]. To address this problem, we adopt Kalman filter to smooth the estimated time-to-contact. Kalman filter is widely used for obtaining more precise measurements by using Bayesian inference and estimating a joint probability distribution over the variables for each time frame even though there are noise and inaccuracies in the original measurements. Here we set τ as the system state and assume the differential equation is [50]:

$$\begin{cases} \tau_i = A_k \tau_{i-1} + B_k u_{i-1} + w_{i-1} \\ Z_i = C\tau_i + D u_{i-1} + v_i \end{cases} \tag{2.15}$$

where, u_{i-1} is the system input at time $i-1$, Z_i is measured time-to-contact at time i, $A = C = 1$ and $B = D = 0$, $w_{i-1} = v_i$ are the process and measurement noise respectively. Then the discrete Kalman filter time predicting equations can be written as:

$$\begin{cases} \bar{\tau}_i = \tau_{i-1} \\ \bar{P}_{ci} = P_{ci-1} + E_x \end{cases} \tag{2.16}$$

where $\bar{\tau}_i$ is predicted state estimate at time i and τ_{i-1} is the filtered state at time $i-1$. And the update can be written as follows:

$$\begin{cases} K_{ci} = \bar{P}_{ci}(\bar{P}_{ci} + E_z)^{-1} \\ \tau_i = \bar{\tau}_i + K_{ci}(Z_i - \bar{\tau}_i) \\ P_{ci} = (1 - K_{ci})\bar{P}_{ci} \end{cases} \tag{2.17}$$

where $E_x = E_z$ are process noise covariance and measurement noise covariance respectively. K_{ci} is the optimal Kalman gain at time i.

2.2.4 Error Based Proportional Controller with Gain Scheduling

After we get the estimation of time-to-contact, a controller can be designed to make the time-to-contact track some reference trajectory so that the robot can perform different tasks such as landing, docking and chasing. Let the reference value for time-to-contact τ_{ref} be a constant value. It has been verified that keeping a constant time-to-contact can realize a soft contact between the observer and object [30, 51].

For the experiment, we focus on the mobile robot docking problem shown in Figure 2.2. In this scenario the robot needs to control its speed as it is approaching the wall to realize a soft contact. For this problem, a mobile robot with a camera attached moves towards a vertical wall, and we

have the following equations:

$$\begin{cases} \dot{X} = -T_x, & X(0) = X_0 \\ \dot{T}_x = a, & T_x(0) = T_{x0} \end{cases} \tag{2.18}$$

where X is the distance from the camera to the wall, T_x is the robot velocity along X axis of the camera frame, and a is the acceleration which is in the same direction with T_x.

For this system, a standard proportional controller is widely used [30]:

$$a = K(\tau - \tau_{ref}) \tag{2.19}$$

where K is a constant proportional parameter. Even though this controller works, to get better control results, we design a new controller which is based on the error with gain scheduling strategy:

$$a = K_p(\tau - \tau_{ref}) \tag{2.20}$$

where

$$K_p = \begin{cases} K_{11}, & \tau \in [\tau_{ref}, \frac{8}{7}\tau_{ref}) \\ K_{12}, & \tau \in [\frac{8}{7}\tau_{ref}, \frac{9}{7}\tau_{ref}) \\ K_{13}, & \tau \in [\frac{9}{7}\tau_{ref}, +\infty) \\ K_{21}, & \tau \in (\frac{6}{7}\tau_{ref}, \tau_{ref}] \\ K_{22}, & \tau \in (\frac{5}{7}\tau_{ref}, \frac{6}{7}\tau_{ref}] \\ K_{23}, & \tau \in (0, \frac{5}{7}\tau_{ref}] \end{cases} \tag{2.21}$$

where, $K_{11} > K_{21}, K_{12} > K_{22}, K_{13} > K_{23}$. The proportional gains are based on the errors, so we call it error based controller in this chapter. Why we design different gains for the same absolute errors can be explained as follows. When X/T_x is larger than τ_{ref}, $\tau = X/T_x$ can be adjusted to reference value by increasing T_x. Since X is decreasing as long as $T_x > 0$, it works similarly with the standard proportional controller. But, when X/T_x is smaller than τ_{ref}, the increasing of X/T_x

is not guaranteed if T_x decreases slowly since X is also decreasing as long as $T_x > 0$. Therefore, we design larger proportional gains for positive errors to compensate the decreasing of X.

2.3 Experiment Results and Discussion

In this section, we conduct experiments to validate the performance of estimation and control algorithms using a mobile robot platform. First, we verify the computational efficiency and accuracy of the featureless direct method. Based on this estimation method, the mobile robot docking experiment is conducted. The performance of the error based controller with gain scheduling strategy is tested. Second, we validate the estimation algorithm with angular velocities using image sequences when the mobile robot has a specific angular velocity. Based on the extended algorithm, we implement the estimation method with angular velocity and realize the docking experiment control using the error based proportional controller with gain scheduling strategy when the robot has angular velocities. To improve the control performance, we also adopt Kalman filter when there is angular velocity.

For the general experiment setup, we developed an integrated system shown in Figure 2.3. A mobile robot (A4WD1 from Robotshop) with four wheel drive serves as the main platform. A Raspberry Pi 2 is used as the central processing unit to interface with a camera and a gyroscope, estimate the time-to-contact, and compute the control command. An Arduino board (Arduino Pro 328 from Sparkfun) is employed to achieve closed-loop speed control of the robot. A forward-facing Raspberry Pi camera module is mounted on the top of the mobile robot with a 3D printed fixture. To validate the estimation algorithm with angular velocity, a distance sensor (OPT2011 from Automationdirect) is used to get the ground truth of time-to-contact. And an ADC converter chip (MCP3008 from Sparkfun) is used to get the digital signal. A gyroscope (MPU6050 from Sparkfun) is also employed to feedback the angular velocity. Several papers with checkerboard patterns are randomly placed on a wall that the robot will move towards.

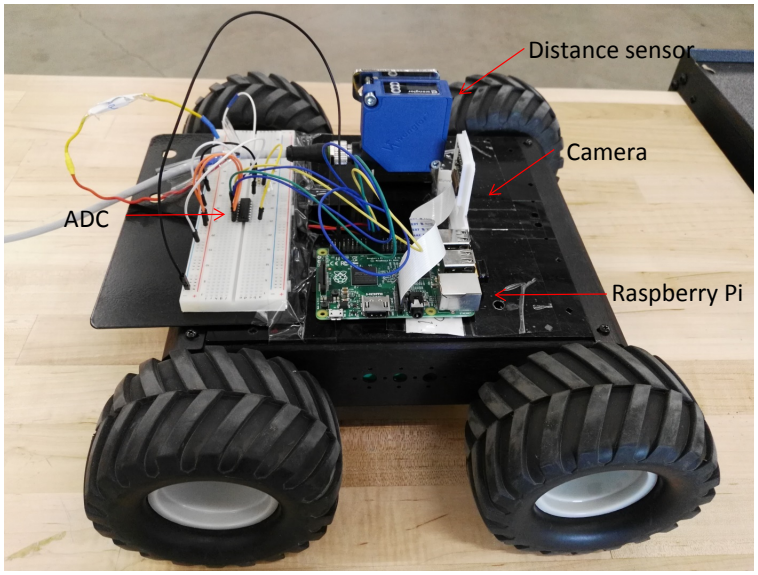


Figure 2.3: The robotic system used for onboard experiments.

2.3.1 Experimental Results without Angular Velocity

In this section, the estimation and control experiments of featureless direct method are conducted. First we verify the computational speed and accuracy of the method, then we carry out the docking experiment without angular velocities.

Estimation Experiment

In this experiment, we use image sequences to compare the computational speed and accuracy of featureless direct method with the feature based method. The images are taken while the robot is moving perpendicular to the wall with a constant speed. The initial distance is $3.81\,\mathrm{m}$ and the constant velocity of the robot is $0.57\,\mathrm{m/s}$. After taking 150 successive images, the robot will stop. We estimate the computational speed and accuracy for the optic flow feature based method, whose source code in Matlab is available [33]. Both estimation experiments are conducted on Matlab2015a on a desktop (Intel (R) Core (TM) i-74790, 3.6GHz CPU, 4Gb RAM, 64 bit). With a resolution of 192×72 pixels for each image, the computation time for each image pair is listed in Table 2.1 for the direct featureless method and optical flow based method. In Table 2.1, we also listed the average absolute error obtained by getting the mean of the difference between the ground

truth and estimation value. One of the estimation results of featureless direct method and optic flow based method are illustrated in Figure 2.4.

From the results, it is obvious that the featureless direct method is more computational efficient compared with optic flow based method. Since every time the optic flow based method will generate different estimations which depend on feature extraction and tracking, we run the optic flow based method for 5 times. The result shown in Figure 2.4 is the second estimation result. The 5 estimations generate average absolute error of 0.73, 1.12, 1.64, 1.70 and 1.61 respectively.

It is notable that the estimation error of the featureless direct method in equation (2.5) also relates to slope of the wall and true time-to-contact. Namely, larger surface slope or true time-to-contact generates larger estimation error.

Table 2.1: Comparison of different methods

Method	Time for each image pair (s)	Average absolute error
Featureless direct method	0.037	0.4821
Optic flow(L-K method) based method	0.14	1.3629

Control Experiment

In this experiment, the estimation algorithm for case II in section 2.1.2 and the control law in equation (2.20) are combined. The reference time-to-contact is set to be $2\,\mathrm{s}$. We set the gain scheduling parameters $K_{11} = 6, K_{12} = 8, K_{13} = 10, K_{21} = 2, K_{22} = 4, K_{23} = 6$ respectively. The robot moves towards a fronto-parallel wall with an initial speed T_{z0} of $0.38\,\mathrm{m/s}$, maximum speed T_{zmax} of $1.14\,\mathrm{m/s}$ and initial distance of $4.6\,\mathrm{m}$. As shown in Figure 2.5, for each control loop, the Raspberry Pi will control the camera to grab a color image with a resolution of 192×144, then convert the image into grayscale. Then the Raspberry Pi will run the estimation and control algorithm, and send the speed command to the Arduino board. Sample images during the motion are shown in Figure 2.6. Since part of each image includes ground information, we only use the upper half part of the image (192×72 pixels) for estimation and control. With such a resolution,

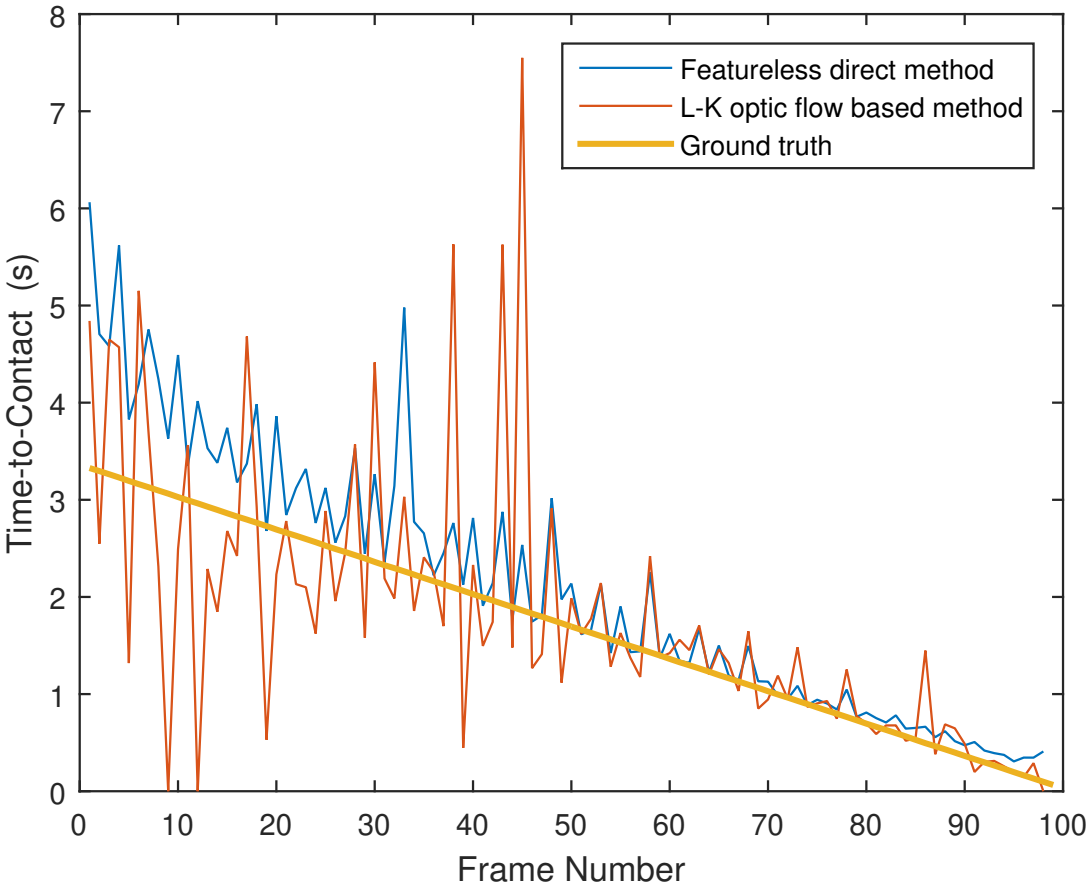


Figure 2.4: The estimation results for featureless direct method and optic flow based method

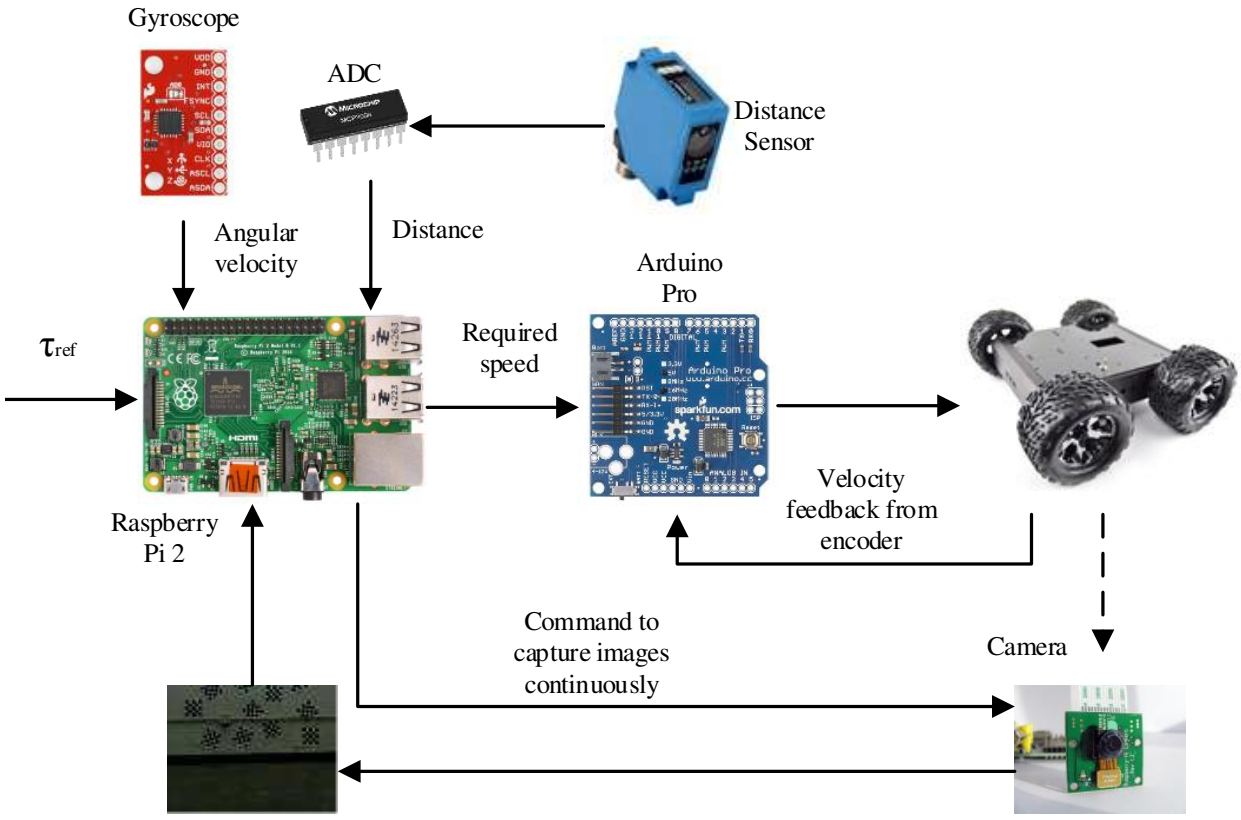
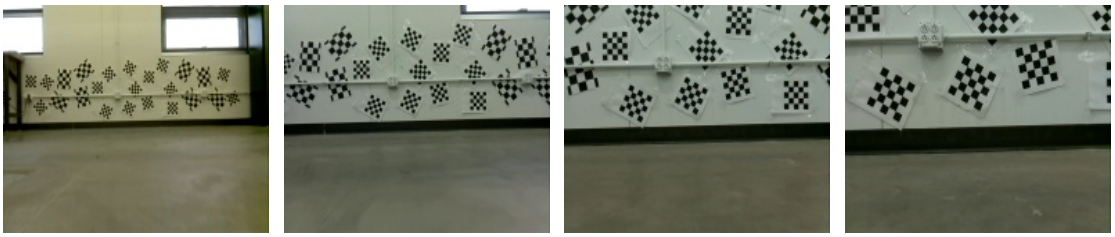


Figure 2.5: Schematic for closed-loop control using the featureless estimation method and proportional controller.

the Raspberry Pi can accomplish a control frequency of 20 Hz. Note that in this experiment, the gyroscope, distance sensor, and Kalman filter are not applied.

To see the control performance of the error based proportional controller with gain scheduling strategy, five experiments with the standard proportional controller are carried out. Figure 2.7 shows the estimated time-to-contact and robot speed with respect to time without gain scheduling strategy. Figure 2.8 shows the results of five docking experiments with gain scheduling strategy. From Figure 2.8(a), we can see that because of the low speed at the beginning, the estimated time-to-contact is much larger than the reference value, so the robot accelerates to decrease time-to-contact. After that, the estimated time-to-contact is smaller than the reference value, and the robot decelerates. Even there exist some vibrations, the time-to-contact approximately stays around the reference value. The low quality of the camera mainly contributes to these vibrations. Although there are some vibrations in the estimation, the robot is almost continuously decreasing speed as



(a) Image 1 **(b)** Image 25 **(c)** Image 50 **(d)** Image 73

Figure 2.6: Sample images for docking experiment. The four images are from the first experiment of the error based controller. The first image is dimmer than others because of different light conditions from the window.

shown in Figure 2.8(b), which realizes the safe docking task. However, some vibrations exist when the robot is going to stop. These vibrations are caused by the algorithm when the robot is close to any object [43, 52].

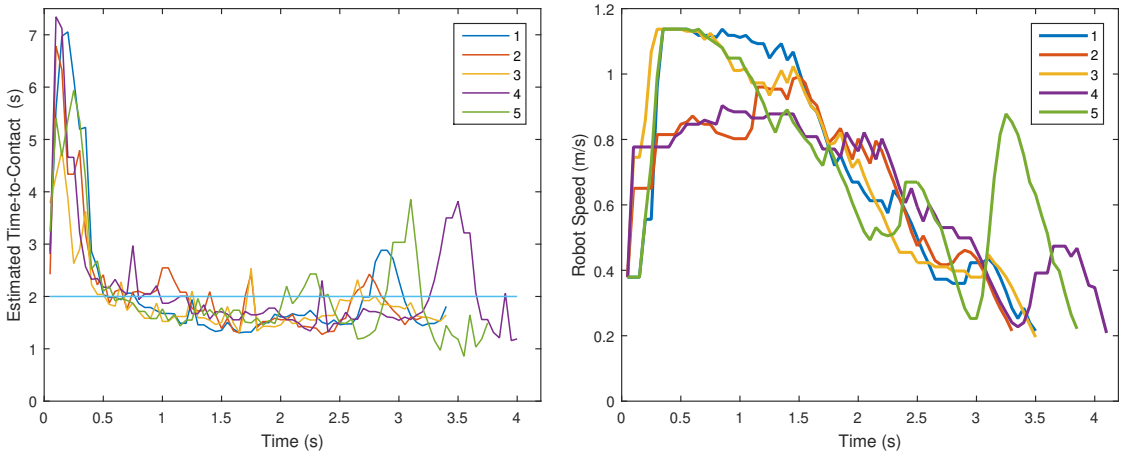
From the results we can see that error based proportional controller generates better performance since the average steady state error of time-to-contact is $0.36\,\mathrm{s}$ while the standard proportional controller gives the average steady state error of $0.45\,\mathrm{s}$. Especially when the robot is near the wall and there are less vibrations and the amplitude of the vibrations are smaller compared with the results of the standard proportional controller.

2.3.2 Experimental Results with Angular Velocities

In this section, we first validate the featureless estimation algorithm with angular velocity, then use the estimation algorithm and the error based gain scheduling controller to perform the docking experiment. To test the performance of the Kalman filter in time-to-contact estimation algorithm, we also perform docking control experiments with and without Kalman filter separately when there is angular velocity.

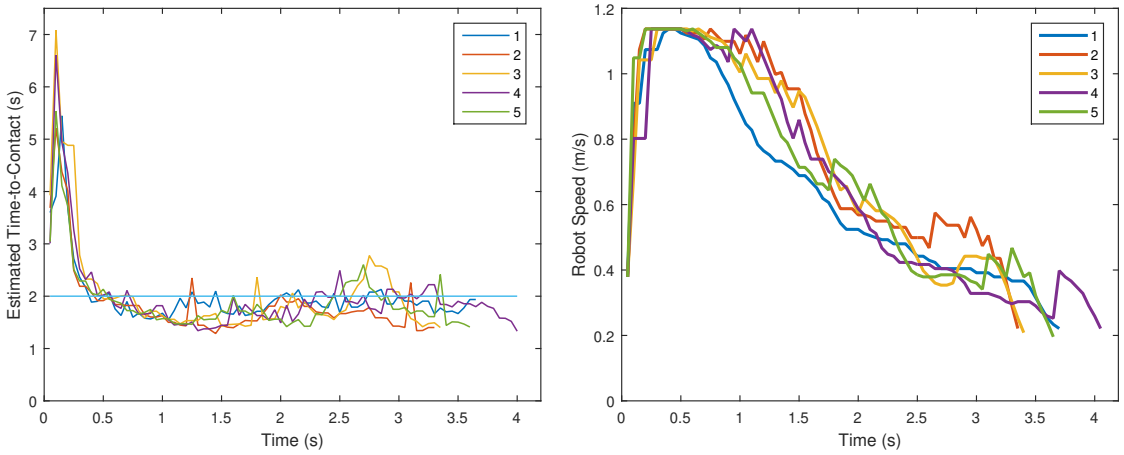
Estimation Experiment

To validate the time-to-contact estimation algorithm with angular velocities, first we need to get the ground truth of time-to-contact. We measure the distance with the distance sensor. Angular velocity is measured by the gyroscope and sent to Raspberry Pi 2 through I2C communication.



(a) Estimated Time-to-Contact value **(b)** Speed of Robot

Figure 2.7: Experiment results for standard proportional controller. (a) is the controlled time-to-contact of the 5 experiments with the standard proportional controller. The horizontal blue line is τ_{ref} =2 s. And (b) is the robot velocity during the experiment. The five colored curves represent the five experiment results in both figures. For the five experiments, the robot stopped after it took 68, 64, 68, 80, and 75 images, respectively.



(a) Estimated Time-to-contact **(b)** Speed of Robot

Figure 2.8: Experiment results for error based proportional controller with gain scheduling strategy. (a) is the controlled time-to-contact of the 5 experiments with the error based proportional controller. The horizontal blue line is τ_{ref} = 2 s. (b) is the robot velocity during the experiment. The five colored curves represent the five experiment results in both figures. For the five experiments, the robot stopped after it took 73, 66, 67, 80, and 72 images, respectively.

(a) Image 1 **(b)** Image 30 **(c)** Image 60 **(d)** Image 90

Figure 2.9: Sample images for estimated time-to-contact with angular velocity. The four images are from the experiment with angular velocity of which the mean is $-45\,^{\circ}/\mathrm{s}$ around Z axis.

The angular velocity which is about $-45\,^{\circ}/\mathrm{s}$ is applied by driving the wheels on the two sides with different speed. In this case, only angular velocity around Z axis exists. The images are acquired continuously right after the distance and angular velocity are sampled while the robot is moving.

The estimated time-to-contact is computed with equation (2.14) and the ground truth of time-to-contact value is calculated by: X/T_x. X is the distance from the distance sensor. To compare the performance of the extended algorithm with the algorithm which does not consider angular velocities, we also plotted the estimation results of equation (2.5). Figure 2.9 represents the sample images in the experiment. Figure 2.10 shows the estimation result with Kamlan filter for the experiment.

From Figure 2.10, we can see that the trend of the estimated time-to-contact of extended algorithm follows the ground truth. The extended algorithm gives the average absolute error of $0.43\,\mathrm{s}$, while the original algorithm gives the average absolute error of $0.87\,\mathrm{s}$. It verifies the necessity and the accuracy of the algorithm. Note that the smaller the angular velocity is, the less difference between the two algorithms is. Nevertheless, there are still some errors. The following reasons may contribute to the error: (1) Although the robot does not move, the small unevenness of the ground may lead to rotation about X and Y axis. (2) There is noise when gyroscope measures the angular velocity. (3) The quality of the camera is not good enough as it has light intensity vibration. (4) The slope of the wall also influence the estimation result since the algorithm does not consider the slope.

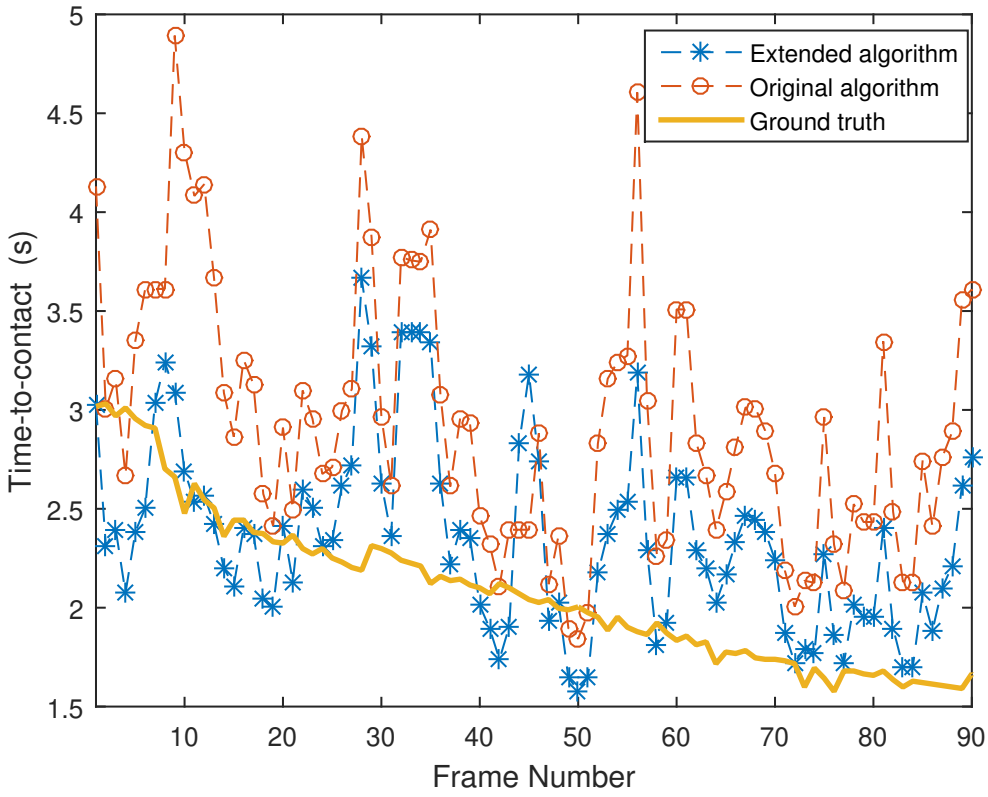
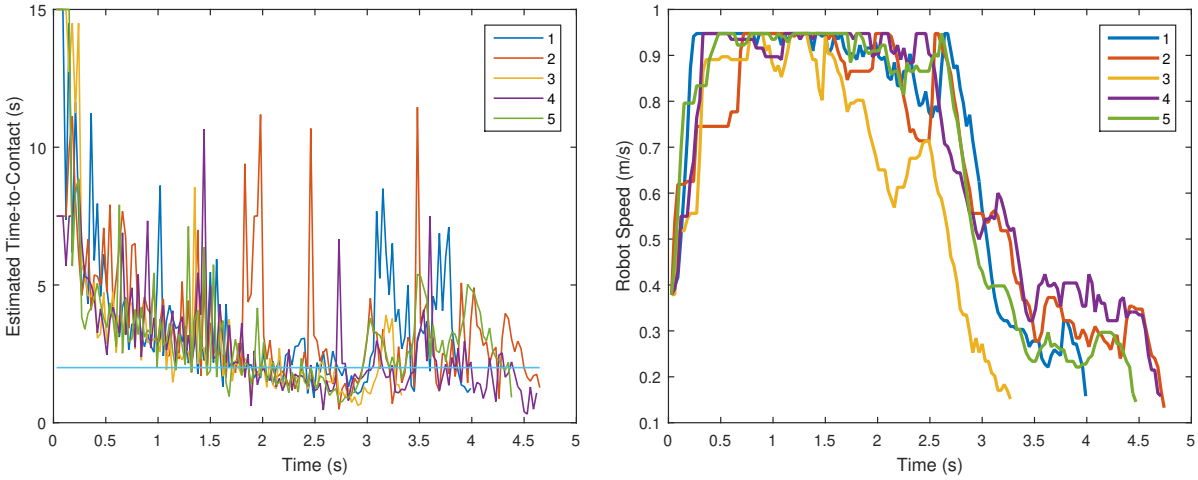


Figure 2.10: Time-to-contact with angular velocity

Control Experiment

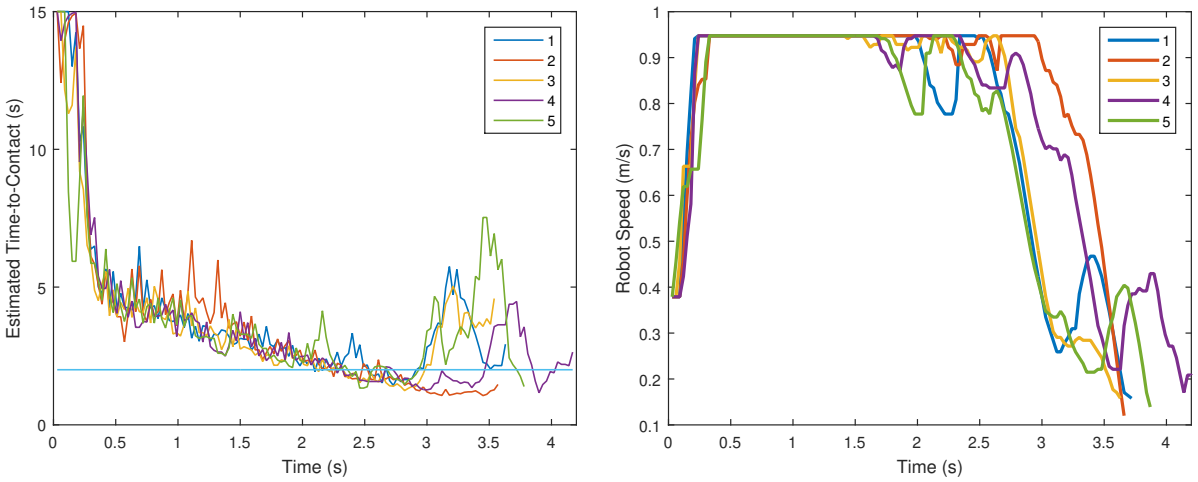
In this section, we applied the estimation algorithm with angular velocities to control the docking process of the mobile robot using the error based proportional controller with gain scheduling strategy. Two experiments are carried out: one with Kalman filter to smooth the estimation value and the other one without Kalman filter. In these two experiments, the reference value of τ is set to be $2\,\mathrm{s}$. The robot moves towards a fronto-parallel wall with an initial speed T_{x0} of $0.38\,\mathrm{m/s}$ and maximum speed T_{xmax} of $0.95\,\mathrm{m/s}$. The initial distance from the wall to robot is $4.2\,\mathrm{m}$.

From Figure 2.11 and Figure 2.12 we can see that, at the beginning, because of the low speed, the estimated time-to-contact is much larger than the reference value. Then the error based gain scheduling controller begins to work to decrease time-to-contact by acceleration. After several control loops, the estimated time-to-contact is maintained at the reference value. At the end, the estimated time-to-contact is less than the reference value because of the high speed and short distance to the wall. The error based controller begins to increase time-to-contact by deceleration.



(a) Estimated Time-to-contact **(b)** Speed of Robot

Figure 2.11: Experiment results without Kalman filter. (a) is the controlled time-to-contact of the 5 experiments without Kalman filter. The horizontal blue line is τ_{ref}=2 s. (b) is the robot velocity during the experiment. The five colored curves represent the five experiment results in both figures. For the five experiments, the robot stopped after it took 133,158,119,157 and 149 images, respectively.



(a) Estimated Time-to-contact **(b)** Speed of Robot

Figure 2.12: Experiment results with Kalman filter. (a) is the controlled time-to-contact of the 5 experiments with Kalman filter. The horizontal blue line is τ_{ref}=2 s. (b) is the robot velocity during the experiment. The five colored curves represent the five experiment results in both figures. For the five experiments, the robot stopped after it took 124, 122, 121, 142, and 129 images, respectively.

Correspondingly, the speed will increase first and then decrease. By comparing the two figures, it is obvious that Kalman filter plays an important role in smoothing the estimation results and minimizing the speed vibrations.

2.4 Chapter Summary

Time-to-contact is a biologically inspired concept which can be applied to control the motion of a robot to fulfill tasks such as landing, perching, or docking. In this chapter, a featureless method is employed to estimate time-to-contact from image sequences. Such a method does not need to extract and track features, resulting in more efficient computation compared with other feature based methods. An error based controller with gain scheduling is implemented together with the featureless estimation algorithm on a mobile robot platform. The speed of the mobile robot is successfully controlled to maintain a reference time-to-contact. In addition, we also extended the featureless estimation algorithm to incorporate angular velocities. Validation experiments and control experiments are carried out. With the Kalman filter, the estimation algorithm and control strategy lead to better performance. Future efforts will be focused on the landing control and navigation of aerial robots. The results presented in this chapter can be readily applied to miniature robots that only have the vision sensor for navigation and control.

Chapter 3

Computational Intelligence: Trajectory Planning

The second part of computational intelligence lies in the trajectory planning in TTC space. In last chapter we investigated how to leverage the constant tau strategy for braking to realize a soft contact. And a second widely used tau reference is the constant tau dot strategy which can realize zero contact velocity. However, in the perching scenario, a non-zero contact velocity is usually necessary to make the perching mechanism functionalize. In this chapter, we propose two tau based strategies, constant tau dot based two-stage strategy (CTDTS) and inverse polynomial based two-stage strategy (IPTS), to realize the non-zero contact velocity. And IPTS can satisfy more constraints with higher order polynomial. To verify the feasibility of CTDTS and IPTS for non-zero contact velocity based perching tasks. We conduct perching experiments with a palm-size quadcopter based on CTDTS and IPTS, respectively.

3.1 Introduction

As talk in chapter 1, researchers have investigated aerial robot perching from different perspectives. However, almost all of the existing investigations on perching are position-based, i.e., they require the precise position feedback either through global positioning systems (GPS) or motion tracking system, making them unsuitable for autonomous perching in situations where positions cannot be obtained (e.g., GPS-denied environments). In this chapter, we leverage the concept of time-to-contact (TTC), defined as the projected time to contact a surface with the current velocity, for the planning and control of aerial perching. Compared with position-based perching methods, TTC-based methods can utilize simple but effective strategies to achieve autonomous perching without complex planning and control. Further, it can be potentially realized with onboard lightweight vision sensors to estimate TTC, which is ideal for miniature aerial robots as they cannot carry heavy sensors (e.g., LIDAR).

TTC or tau has been widely found in controlling the motion for humans, animals, and insects. By estimating TTC from visual feedback, drivers can determine how to avoid collisions [23]. Bees keep a constant rate of image expansion (equivalent to TTC) to land on various vertical surfaces [17]. Pigeons are discovered to adopt TTC to safely perch on branches [21]. Seabirds can leverage TTC to adjust the timing to close wings before diving into the water for fish [19].

With biological inspirations, tau theory has also been recently applied to various robotic applications such as avoiding obstacles or landing on ground [31]. For existing tau-theory based planning and control, the general architecture is illustrated in Figure 2.1 and can be described as follows. First, a reference trajectory for tau or TTC is planned off-line based on the desired task (e.g., perching, docking, or landing). Then by comparing the reference tau with the estimated tau, which can be obtained from image feedback of cameras, GPS, or distance sensors, a controller is designed to control the robot's motion so that the reference tau can be tracked to accomplish the desired task.

Substantial work has been performed to address the estimation and control problem shown in Figure 2.1. Although extensive research has been carried out for estimation and control, the trajectory planning problem in tau theory is underexplored. Indeed, most of the existing research simply utilizes the constant tau dot strategy to generate the reference trajectory of tau [31]. However, directly applying constant tau dot strategy to perching can only control the contact velocity to be zero [30, 31], which is not always desired for robotic perching. In fact, most perching mechanisms require a substantial velocity in the direction perpendicular to the perching object to ensure that gripping mechanisms can robustly attach to the object [13]. For example, the mechanism in [13] requires a minimum normal velocity of $0.8\,\mathrm{m/s}$ for successful perching. In such cases with non-zero contact velocity, the existing tau theory is unlikely to work. To address this problem, we have extended the tau theory by proposing a two-stage strategy to control the contact speed to a specific value and validated the theory using a mobile robot [53]. However, the strategy can only satisfy two constraints (contact velocity and maximum deceleration), making robust perching not feasible as they normally require several constraints to be satisfied [13, 54]. In this chapter, we propose a

new planning strategy for TTC-based robotic perching and validate the proposed strategy using a palm-size quadcopter.

Our major contribution in this chapter is to leverage TTC or tau to accomplish robotic perching, which requires simpler planning and control compared with position-based approaches. Specifically, there are two contributions. First, we propose a new two-stage planning method to generate the reference trajectory for TTC. Such a method can generate optimal trajectories satisfying multiple constraints required for robust perching. Second, we validate the proposed planning strategy using a palm-size quadcopter by mapping the planned trajectory in tau space into the commands acceptable by the quadcopter. Note that in this chapter, although TTC is estimated from the motion tracking system in the experiments, we plan to integrate our vision-based estimation algorithm [49, 55] with the proposed strategy for vision-based perching using onboard cameras in the future using a larger quadcopter currently under developments.

The rest of this chapter is organized as follows. Section 3.2 describes existing planning strategies for tau, including the widely used constant tau dot strategy (CTDS) [31] and our recently proposed constant tau dot based two-stage strategy (CTDTS) [53]. The newly proposed two-stage strategy will also be discussed in detail. Based on the planned reference, section 3.3 presents a controller design to track the reference for perching with a quadcopter. To verify the performance of the proposed planning strategies and control methods, section 3.4 discusses and compares the simulation and experiment results.

3.2 Tau based trajectory generation

In this section, we first introduce two trajectory generation methods for tau (CTDS and CTDTS), discuss the need for new strategies for robust aerial perching, and detail the new inverse of polynomial based two-stage strategy (IPTS).

As shown in Figure 3.1, the perching problem aims to control the motion of an aerial robot flying towards a surface to contact the surface with a perching speed in a specific range so that the gripping mechanism can robustly attach to the surface [56]. Generally, the orientations should

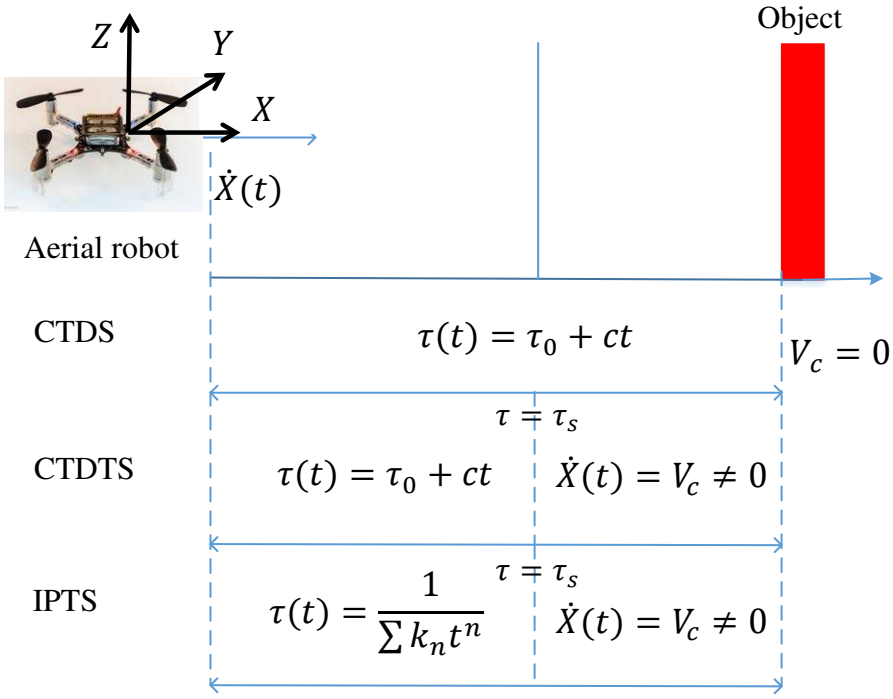


Figure 3.1: General idea for three tau based strategies for perching, constant tau dot strategy (CTDS), constant tau dot based two-stage strategy (CTDTS) and inverse of polynomial based two-stage strategy (IPTS).

also be adjusted appropriately before the final touchdown; however, in this chapter, we will not consider the attitude maneuvering as our main goal is to investigate the control of contact velocity through the use of tau. If we establish a coordinate frame attached to the surface with X along the perching direction, then the gap $X(t) < 0$ and velocity $\dot{X}(t) > 0$. The desired contact velocity is $V_c \in (V_l, V_u)$, where V_l and V_u are the lower and upper bound for successful perching velocity, respectively. For different perching mechanisms, V_l and V_u vary and can be obtained experimentally. Without loss of generality, we assume the initial velocity is larger than the required contact velocity $\dot{X}(0) > V_c$ since biological organisms generally decelerate to perch [5]. Let t_c be the time when contact occurs, then the non-zero contact velocity requirement is $\dot{X}(t_c) = V_c \neq 0$.

3.2.1 Constant tau dot strategy (CTDS)

For braking or landing problem, the constant tau dot strategy (CTDS) has been shown to guide animals or humans to smoothly decelerate to zero contact velocity [57, 58]. Also, the applications of CTDS in robotic field effectively demonstrate its potential for robot motion control [31].

To better understand the other two strategies, we briefly summarize the CTDS as follows. First, as mentioned in the introduction, tau or time-to-contact (TTC) is defined as:

$$\tau(t) = \frac{X(t)}{\dot{X}(t)} \tag{3.1}$$

For CTDS, suppose the rate of change of τ, i.e., the tau-dot, is a constant c, then we have:

$$\tau(t) = ct + \tau_0, \quad \tau_0 = \frac{X(0)\dot{X}(0)}{<}0 \tag{3.2}$$

where $X(0) < 0$ is the initial distance, and $\dot{X}(0) > 0$ is the initial velocity. Combining equations (3.1) and (3.2), we can solve for $X(t)$, $\dot{X}(t)$, and $\ddot{X}(t)$:

$$X(t) = X_0 \left(1 + c\frac{\dot{X}_0}{X_0}t\right)^{1/c} \tag{3.3}$$

$$\dot{X}(t) = \dot{X}_0 \left(1 + c\frac{\dot{X}_0}{X_0}t\right)^{1/c-1} \tag{3.4}$$

$$\ddot{X}(t) = \frac{\dot{X}_0^2}{X_0}(1-c) \left(1 + c\frac{\dot{X}_0}{X_0}t\right)^{1/c-2} \tag{3.5}$$

where $X_0 = X(0)$ and $\dot{X}_0 = \dot{X}(0)$. Based on the above equations, we cannot realize non-zero contact velocities with c having different values. In fact, we can discuss it based on the ranges of c according to [31]:

- When $c \leq 0$, $X(t)$, $\dot{X}(t)$, $\ddot{X}(t)$ converge asymptotically to 0 for $t \to \infty$. Therefore, perching is impossible since the time cannot be infinite when the contact occurs.

- When $0 < c \leq 0.5$, $X(t)$, $\dot{X}(t)$, $\ddot{X}(t)$ become 0 at the same finite time. Although perching can be accomplished in finite time, the contact velocity will always be zero.

- When $0.5 < c < 1$, $X(t)$ and $\dot{X}(t)$ become 0 at the same finite time, but $\ddot{X}(t)$ becomes ∞. Perching is infeasible since robots cannot have infinitely large accelerations.

- When $c = 1$, a robot will move towards the surface with a constant velocity—the initial velocity. Perching is feasible in this case, but the contact velocity is fixed.

- When $c > 1$, $\dot{X} \to \infty$, $\ddot{X} \to \infty$ as $X \to 0$. Perching is again infeasible since robots cannot have infinitely large velocities and accelerations.

For the general CTDS, c is chosen to be in $(0, 0.5]$ so that the distance, velocity, and acceleration can be zero in the same finite time to realize contact with a surface for landing or docking applications [31]. However, we can see that, with $c \in (0, 0.5]$, the desired non-zero contact velocity cannot be achieved since when the contact occurs, i.e., $X(t) = 0$, we have $1 + c\frac{\dot{X}_0}{X_0}t = 0$, so $\dot{X}(t)$ and $\ddot{X}(t)$ would also be zero [31]. This means when the contact or perching occurs, both the velocity and acceleration must be zero. In this case, the perching may fail if a non-zero contact velocity is required. To address this problem, we have proposed the constant tau dot based two-stage strategy (CTDTS) [53].

3.2.2 Constant tau dot based two-stage strategy (CTDTS)

As shown in Figure 3.1, CTDTS uses a constant tau dot strategy in the first stage to decelerate to the desired contact velocity V_c. The second stage initiates when tau is larger than a prescribed threshold τ_s (Note that $\tau \leq 0$ owing to the choice of frame setup as shown in Figure 3.1). In the second stage, the same forward velocity is maintained at V_c until perching occurs. Note that the initiation of the second stage is different from our previous CTDTS presented in [53], where a prescribed distance is used. Using tau is better than distance as tau specifies how soon will the robot contact the surface so that the robot can initiate an attitude maneuvering if necessary. Tau-based threshold is also adopted in biological systems (e.g., flies [59] or hawks [60]) for attitude

maneuvering or leg extensions before the final touchdown. In general, the magnitude of τ_s is rather small and we assume that $\tau_0 < \tau_s < 0$, where τ_0 is the initial tau.

Assume the second stage starts at time t_s with velocity V_c, then the reference trajectory for tau can be represented in two stages

$$\tau_{ref}(t) = \begin{cases} ct + \tau_0, & \text{if } 0 \leq t \leq t_s \\ \tau_s + t - t_s, & \text{if } t > t_s \end{cases} \tag{3.6}$$

With a specified threshold τ_s, desired contact velocity V_c, and τ_0 depending on the initial conditions, we need to solve the constant tau dot c and the switching time t_s to obtain $\tau_{ref}(t)$. A unique solution can be found for c and t_s by using the desired tau (τ_s) and velocity (V_c) at the stage transition: $\tau_{ref}(t_s) = \tau_s$ and $\dot{X}(t_s) = V_c$. In fact, the solution can be found as:

$$c = \frac{\log(\frac{\tau_s}{\tau_0})}{\log(\frac{V_c}{\dot{X}_0}) + \log(\frac{\tau_s}{\tau_0})} \tag{3.7}$$

$$t_s = \frac{\tau_s - \tau_0}{c} \tag{3.8}$$

With $\dot{X}_0 > V_c > 0$ and $\tau_0 < \tau_s < 0$, we can easily show that c will be in $(0, 1)$. However, the unique solution for c and t_s should satisfy the constraints for the acceleration capability of a flying robot which is limited by its motors. Note that for the first stage with a constant tau dot, the robot always decreases its speed when $c \in (0, 1)$, i.e., $\ddot{X}(t)$ is always negative. For the deceleration of the whole process, after analysis, we know that when $t \in [0, t_s]$ [53]:

- If $0 < c \leq 0.5$, $\ddot{X}(t)$ will monotonically increase. Since $\ddot{X}_0 < 0$, the maximum deceleration should be achieved at $t = 0$.
- If $0.5 < c < 1$, $\ddot{X}(t)$ will monotonically decrease. Since $\ddot{X}_0 < 0$, the maximum deceleration should be achieved at $t = t_s$;

Therefore, the solution of c should satisfy the following deceleration constraint:

$$\ddot{X}(0) = (1-c)\frac{\dot{X}_0^2}{X_0} < a_{max}, \qquad \text{if } 0 < c \leq 0.5 \tag{3.9}$$

$$\ddot{X}(t_s) = (1-c)\frac{\dot{X}_0^2}{X_0}\left(\frac{V_c}{\dot{X}_0}\right)^{\frac{1-2c}{1-c}} < a_{max}, \qquad \text{if } 0.5 < c < 1 \tag{3.10}$$

where a_{max} is the maximum acceleration/deceleration of the robot. With such constraints, we can see that a major limitation for CTDTS is that the unique solution c may not satisfy the constraints specified in equation (3.9) or (3.10), leading to no feasible solution for the reference tau. Therefore, new strategies should be developed to address this issue.

To compare the time required for whole perching process with the proposed planning strategy to be discussed in the next subsection, we obtain the total time for the two stages as [53]:

$$t = \frac{X_0}{c\dot{X}_0}\left[\left(\frac{V_c}{\dot{X}_0}\right)^{\frac{c}{1-c}} - 1\right] - \left(\frac{X_0}{\dot{X}_0}\right)^{\frac{1}{1-c}} \tag{3.11}$$

3.2.3 Inverse of polynomial based two-stage strategy (IPTS)

The shortcomings of CTDTS can be addressed by proposing new strategies with more parameters in the first stage while keeping the second stage the same. In this subsection, we discuss a new mathematical form of tau reference in the first stage based on the inverse of a polynomial:

$$\tau(t) = \frac{1}{\sum_{i=0}^{n} k_n t^n} \tag{3.12}$$

where k_i ($i = 0, 1, \ldots, n$) are parameters to be determined. $k_0 = 1/\tau_0$ can be determined by initial conditions, while k_1, k_2, ..., k_n can be determined from multiple constraints or optimizations for minimizing time, control effort, or energy, etc. As shown in Figure 3.1, the second stage for IPTS is still constant velocity after tau reaches a specified threshold τ_s.

With the proposed trajectory in equation (3.12), the distance, velocity, and acceleration in the first stage can be solved analytically from integration:

$$X(t) = X_0 \exp\left(\sum_{i=0}^{n} \frac{k_i}{i+1} t^{i+1}\right) \tag{3.13}$$

$$\dot{X}(t) = X_0 \sum_{i=0}^{n} k_i t^i \exp\left(\sum_{i=0}^{n} \frac{k_i}{i+1} t^{i+1}\right) \tag{3.14}$$

$$\ddot{X}(t) = X_0 \left[\sum_{i=1}^{n} i k_i t^{i-1} + \left(\sum_{i=0}^{n} k_i t^i\right)^2\right] \exp\left(\sum_{i=0}^{n} \frac{k_i}{i+1} t^{i+1}\right) \tag{3.15}$$

The inverse of polynomial approach has three major advantages compared with the constant tau-dot. First, with a larger n, more constraints can be satisfied since we need to solve for more parameters. Therefore, the limitation for CTDTS can be eliminated. Second, as will be shown in simulations and experiments, IPTS can generate a reference trajectory with a shorter perching time, since the robot can accelerate first and then decelerate to the desired velocity. On the contrary, as discussed in section 2.2, CTDTS can only decelerate to the desired velocity. Third, the resulting reference trajectories for both CTDS and CTDTS rely on the initial conditions (distance and velocity) as can be seen from equations (3.3, 3.4, 3.5, 3.8). However, as we will numerically show later, a fixed set of parameters for IPTS will work if the initial conditions are in a given range, greatly facilitating implementations since estimating distance and velocity from tau is a difficult problem [52, 61–63].

Since the number of parameters n in IPTS is flexible, we generally cannot obtain a unique solution for them. In this case, we will take the minimum time optimization as an example to obtain a unique solution and realize perching with the non-zero contact velocity requirement. For this optimization, several constraints are set: (1) Similar to CTDTS, the second stage initiates at time t_s while the tau reaches a specified threshold τ_s. (2) The velocity V_c at time t_s, i.e., the contact velocity, should be in a reasonable range (V_l, V_u). (3) The velocity for the whole process should be less than the maximum velocity V_{max}. (4) The acceleration/deceleration should be less than the maximum value limited by the capabilities of the motors. With such constraints, if we want to minimize the total time t for perching, the optimization can be formulated as:

$$\min_{k_1,k_2,\ldots,k_n} \quad t = f(k_1, k_2 \ldots k_n) \tag{3.16a}$$

$$\text{subject to} \quad \tau(t_s) = \tau_s \tag{3.16b}$$

$$V_l \leq \dot{X}(t_s) \leq V_u \tag{3.16c}$$

$$\dot{X}(t) \leq V_{max} \qquad \text{for } 0 \leq t \leq t_s \tag{3.16d}$$

$$\left|\ddot{X}(t)\right| \leq a_{max} \qquad \text{for } 0 \leq t \leq t_s \tag{3.16e}$$

where $t = t_s + |\tau_s|$ is the total time for the two stages with t_s the time for the first stage, which can be numerically solved from $\tau(t_s) = 1/(\sum_{i=0}^{n} k_n t^n) = \tau_s$. It should be noted that if we want to minimize the total perching time as formulated in equation (3.16), then the resulting parameters k_i will uniquely depend on the initial conditions, i.e., given a set of initial conditions, we can solve for a set of k_i. However, if we don't minimize the time, then it is possible that we can find a set of k_i that will only loosely depend on the initial conditions, i.e., as long as the initial conditions are in a range, a fixed set of k_i will make sure the perching constraints are satisfied.

3.3 Tau controller for aerial robots

We will verify the proposed trajectory generation methods in tau space using a palm-size and open source quadcopter (Crazyflie 2.0, Bitcraze). To this end, we need to design a controller to control the motion of the quadcopter to track the planned reference trajectories for successful perching. The modeling and control for quadcopters have been investigated intensively in literature [12, 31, 64–68]. In general, the control inputs for quadcopters are a combination of a thrust force and a torque vector, of which the directions are along or around the axis of a body frame attached to a quadcopter.

The Crazyflie quadcopter has an onboard and pre-tuned attitude controller to stabilize the orientations around a reference orientation $[\theta_{ref}, \phi_{ref}, \psi_{ref}]$ with θ, ϕ, and ψ the roll, pitch, yaw angle, respectively. The attitude controller is implemented with the onboard IMU and the autopilot. In order to control the robot to track the desired reference tau, we need to generate $[\theta_{ref}, \phi_{ref}, \psi_{ref}]$

and the thrust force T. ψ_{ref} is set to be zero which ensures the Crazyflie always faces the perching surface so that the perching mechanism can work properly.

As shown in Figure 3.2, θ_{ref}, ϕ_{ref}, and T are obtained as follows. First, we implement a tau controller in the desired perching direction X.

$$u_x = k_{px}(1 - \frac{\tau_{ref}}{\tau}) + k_{ix}\int(1 - \frac{\tau_{ref}}{\tau}) + k_{dx}\frac{d}{dt}(1 - \frac{\tau_{ref}}{\tau}) \tag{3.17}$$

In this controller, instead of directly using $\tau_{ref} - \tau$ as the error item, we leverage $1 - \tau_{ref}/\tau$ since such an error item has shown to have better performance [31]. Theoretically, we only need the tau controller in equation (3.17) to control the motion along perching direction (in the world frame shown in Figure 3.1, it is along the X direction). However, for our experiments later, we need to control the motion in Y and Z direction since the field of view (FOV) of our motion tracking system is limited. If the motion along Y and Z direction is not controlled, then the robot may fly outside the FOV. The position control in Y and Z is implemented as a PID controller:

$$\begin{cases} u_y = k_{py}(y_{ref} - y) + k_{iy}\int(y_{ref} - y) + k_{dy}\frac{d}{dt}(y_{ref} - y) \\ u_z = k_{pz}(z_{ref} - z) + k_{iz}\int(z_{ref} - z) + k_{dz}\frac{d}{dt}(z_{ref} - z) \end{cases} \tag{3.18}$$

where y_{ref} and z_{ref} are set to be zero. With the computed (u_x, u_y, u_z) expressed in world frame, we can obtain θ_{ref}, ϕ_{ref}, and T by mapping (u_x, u_y, u_z) into the body frame using the following nonlinear transformation [64, 69]:

$$\begin{cases} \phi_{ref} = \sigma_\phi[arcsin(\frac{u_x S_{\psi_{ref}} - u_y C_{\psi_{ref}}}{\sqrt{{u_x}^2 + {u_y}^2 + (u_z+g)^2}})] \\ \theta_{ref} = \sigma_\theta[arctan(\frac{u_x C_{\psi_{ref}} + u_y S_{\psi_{ref}}}{u_z + g})] \\ T = \sigma_T[m(u_x(S_\theta C_{\psi_{ref}} C_\phi + S_{\psi_{ref}} S_\phi) + u_y(S_\theta S_{\psi_{ref}} C_\phi - C_{\psi_{ref}} S_\phi) + (u_z + g) C_\theta C_\phi)] \end{cases} \tag{3.19}$$

where m and g are the mass of Crazyflie and gravitational acceleration, respectively. S and C corresponding to $\sin$ and $\cos$ respectively, and $\sigma()$ is a saturation function to ensure the computed

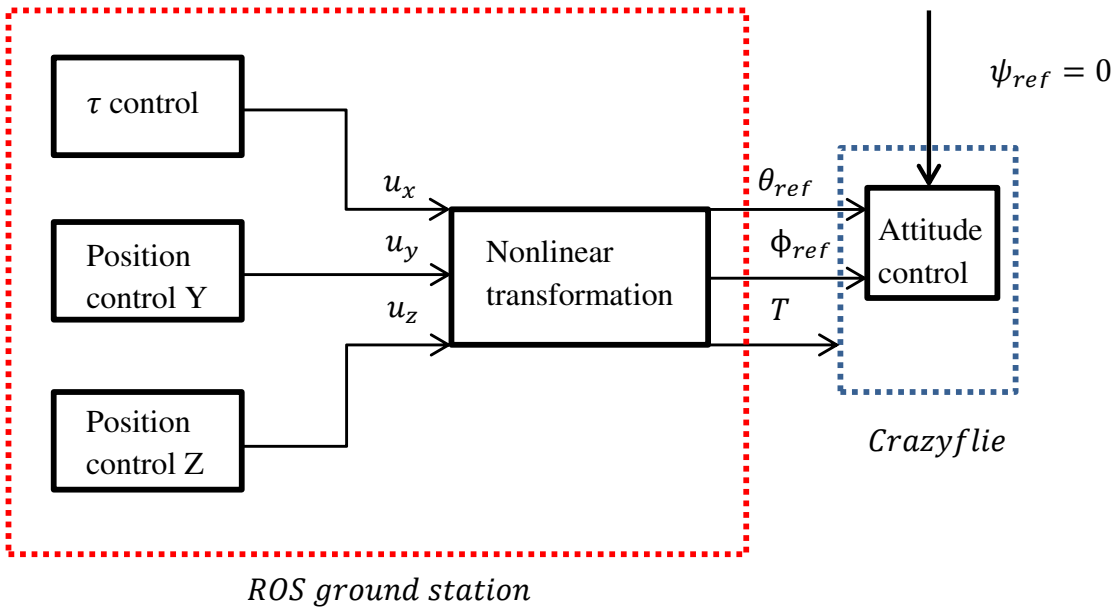


Figure 3.2: General control diagram for experiments. Three controllers are combined to generate the control command for Crazyflie.

roll, pitch and thrust are in a reasonable range. We choose the nonlinear transformation in equation (3.19) due to its simplicity compared with other similar transformations [70, 71].

To gain better control performance, we adopt gain scheduling in τ controller for k_{px} [49]:

$$k_{px} = \begin{cases} k_{px1} & \text{for } \frac{\tau_{ref}}{\tau} > 1 \\ k_{px2} & \text{for } \frac{\tau_{ref}}{\tau} \leq 1 \end{cases} \tag{3.20}$$

where $k_{px1} > k_{px2} > 0$. As explained in [49], the advantage of using different proportional gains is as follows. If $\tau_{ref}/\tau > 1$, we can get $\dot{X} > X/\tau_{ref}$, we need to decelerate to make $\dot{X} = X/\tau_{ref}$ to achieve $\tau_{ref}/\tau = 1$. However, the absolute value of X is also decreasing as the robot flying towards the perching surface. Therefore, we need larger decelerations compared with the case when $\tau_{ref}/\tau \leq 1$. We set $k_{px1} = 2.5k_{px2}$ in our experiments, which will be discussed in the next section.

3.4 Simulation and experiment results

To validate the effectiveness of the proposed planning and control algorithms, we perform both simulation and experiments to show that both CTDTS and IPTS can be leveraged for perching with a non-zero contact velocity with the IPTS being able to satisfy more constraints with a shorter perching time. We also demonstrate through simulations that IPTS will work when initial conditions are not exactly known.

3.4.1 Simulation Results

For simulation, we only simulate the trajectory planning for tau to compare the performance of different strategies discussed in section 3.2, leaving the controller implementation to experiments. The initial conditions and constraints are selected based on the experimental setup to be discussed in the next subsection. Based on the motion tracking system's field of view, the initial conditions are chosen as $X_0 = -3\,\mathrm{m}$ and $1.5\,\mathrm{m/s}$. Based on the perching mechanism (needle) and the perching surface (foam board), the bounds for the contact velocity are set as $V_u = 1\,\mathrm{m/s}$, $V_l = 0.7\,\mathrm{m/s}$. Considering the delay of wireless communication and capabilities of Crazyflie, the maximum velocity and acceleration/deceleration are selected as $V_{max} = 2.5\,\mathrm{m/s}$ and $|a_{max}| = 1.4\,\mathrm{m/s^2}$, respectively.

For the CTDS, in order to accomplish perching in finite time, $c \in (0, 0.5]$. Since the larger the tau dot is, the faster the perching would be completed. Therefore, we set the tau dot to be $c = 0.5$. With such a selection and the initial conditions, the tau reference for the whole perching process is:

$$\tau_{ref_{CTDS}}(t) = 0.5t - 2 \tag{3.21}$$

With such a tau reference, we simulate the CTDS and plot the distance, velocity, acceleration and TTC with the blue lines in Figure 3.3a, 3.3c, 3.3e and 3.3g. As shown in the figure, it takes $4\,\mathrm{s}$ to finish the perching process with *a zero contact velocity*.

For the CTDTS, based on the initial conditions and constraints, we choose the desired contact velocity $V_c = V_u = 1\,\mathrm{m/s}$ so that the perching can be achieved faster. With both the desired

switching tau $\tau_s = -0.5\,\mathrm{s}$ and contact velocity V_c, a unique solution exists for the constant c for tau dot and time duration for the first stage t_s. Considering the initial conditions and constraints, we can solve them as $c = 0.7337$ and $t_s = 1.92\,\mathrm{s}$. The solution of c is found to satisfy the constraint of the acceleration specified in equation (3.10). The resulting tau reference for the whole perching process is thus:

$$\tau_{ref_{CTDTS}}(t) = \begin{cases} 0.7337t - 2, & \text{if } t < 1.92 \\ t - 2.42, & \text{if } t \geq 1.92 \end{cases} \tag{3.22}$$

With such a tau reference, we simulate the CTDTS and plot the distance, velocity, acceleration and TTC with the orange lines in Figure 3.3a, 3.3c, 3.3e and 3.3g. As shown in the figure, it takes $2.42\,\mathrm{s}$ to finish the perching process with a contact velocity the maximum allowable one.

For the IPTS, we use a third order polynomial ($n = 3$) to solve this problem since it is the smallest order of polynomial that can satisfy the constraints in equation (3.16). With the initial conditions and constraints, the optimization is performed with fmincon, a Matlab built in optimization function. To avoid being trapped at local minima, we perform the optimization with different initial conditions for k_1, k_2, and k_3. The optimization algorithm generates the final results for the constants as $k_1 = -0.7166$, $k_2 = -0.1907$, $k_3 = 0.0067$. With such constants, the switching time $1.5148\,\mathrm{s}$. As a result, the tau reference for the whole perching process is:

$$\tau_{ref_{IPTS}}(t) = \begin{cases} \dfrac{1}{0.0067t^3 - 0.1907t^2 - 0.7166t - 0.5}, & \text{if } t < 1.5148 \\ t - 2.014, & \text{if } t \geq 1.5148 \end{cases} \tag{3.23}$$

With such a tau reference, we plot the distance, velocity, acceleration and TTC with the yellow lines in Figure 3.3a, 3.3c, 3.3e and 3.3g. Comparing the simulation results for the three different trajectories, we can conclude that IPTS can take the advantage of motor capabilities to decelerate and accelerate, while CTDTS can only allow for deceleration. Thus, the velocity of the first stage for IPTS is increased first and decreased, which induces the faster perching compared with the CTDTS. Further, CTDTS can only satisfy the acceleration constraint with a fixed contact velocity V_c, while IPTS can satisfy additional velocity constraints with a flexible contact velocity in a range

(V_l, V_u). For CTDTS, the acceleration figure verifies the maximum deceleration occurs at t_s when $c > 0.5$ as shown in equation (3.10).

From the simulation results, we can see that even the desired contact velocity V_c for IPTS is specified in a range (V_l, V_u), the optimization will make $V_c = V_u$ to realize the fastest perching. To investigate whether V_c will be always equal to V_u or not, we change the constraints to $V_{max} = 2\,\mathrm{m/s}$, $|a_{max}| = 1\,\mathrm{m/s^2}$. The resulting contact velocity from the optimization in this case is $0.94\,\mathrm{m/s}$. Therefore, for different constraints, the contact velocity may not be V_u, and the constraint $V_c \in (V_l, V_u)$ is necessary for the optimization of minimum time perching. In this case the tau reference for the whole perching process for IPTS is:

$$\tau_{ref_{IPTS}} = \begin{cases} \dfrac{1}{-0.2563t^3 + 0.295t^2 - 0.5745t - 0.5}, & \text{if } t < 1.7628 \\ t - 2.2628, & \text{if } t \geq 1.7628 \end{cases} \tag{3.24}$$

The simulation results for all the three cases under the new constraints are shown in Figure 3.3b, 3.3d, 3.3f and 3.3h. The reference trajectory for CTDS and CTDTS are plotted in the figure with blue and oranges lines, respectively. Note that for the new constraints, we set $V_c = 0.94\,\mathrm{m/s}$ for CTDTS to compare with the IPTS. The constant c for CTDTS can be solved as 0.7479 and the perching time is $2.504\,\mathrm{s}$. The IPTS switches stage at $t_s = 1.7628\,\mathrm{s}$ and perches at $t = 2.2628\,\mathrm{s}$. From the figure, we can see that the trends of the two simulations with different constraints (left and right column of Figure 3.3) are exactly the same, but the contact velocities are different.

To better visualize the results, we have listed all the constraints and the corresponding perching time in Table 3.1. As can be seen from the table, IPTS can achieve the best performance under the given constraints with a perching time of $2.014\,\mathrm{s}$ or $2.2628\,\mathrm{s}$ compared with $2.42\,\mathrm{s}$ or $2.504\,\mathrm{s}$ for CTDTS and $4\,\mathrm{s}$ for CTDS.

In addition, we also conduct several simulations to show the advantage of IPTS strategy for safe perching when the initial conditions are not exactly known. Under this situation, we don't aim to minimize the total perching time, but only to satisfy the contact velocity, maximum velocity

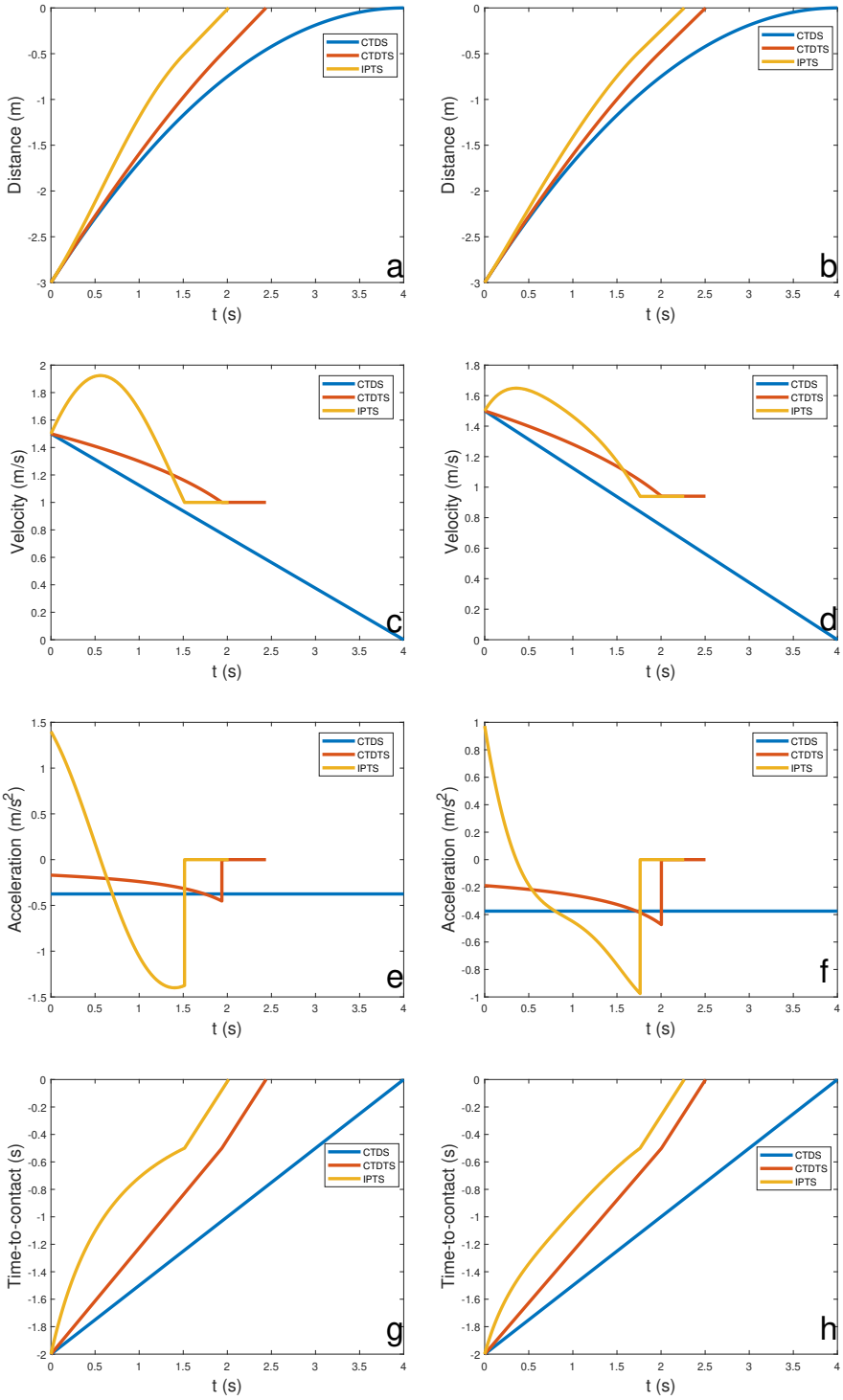


Figure 3.3: Simulation results with different constraints. The left column shows simulations with $V_{max} = 2.5\,\mathrm{m/s}$, $a_{max} = 1.4\,\mathrm{m/s^2}$. The right column shows simulations with $V_{max} = 2.0\,\mathrm{m/s}$, $a_{max} = 1.0\,\mathrm{m/s^2}$. From top to bottom are the distance, velocity, acceleration, and tau for CTDS, CTDTS, IPTS, respectively. The CTDTS and IPTS can both realize the nonzero contact velocity and IPTS generates the faster perching.

Constraints	Strategy	Contact velocity ($\mathrm{m/s}$)	Perching time (s)
$V_{max} = 2.5\,\mathrm{m/s}$ $a_{max} = 1.4\,\mathrm{m/s^2}$	CTDS	0	4
	CTDTS	1	2.42
	IPTS	1	2.014
V_{max} $2.0\,\mathrm{m/s}$ $a_{max} = 1.0\,\mathrm{m/s^2}$	CTDS	0	4
	CTDTS	0.94	2.504
	IPTS	0.94	2.2628

Table 3.1: Simulation results comparison

IPTS order	k	V_0 range	Z_0 range
3rd	$k_1 = -0.60, k_2 = -0.057, k_3 = 0.054$	$[1.9, 2.0]$	$[-3.9, -3.7]$
4th	$k_1 = -0.60, k_2 = 0.15, k_3 = -0.31, k_4 = 0.14$	$[1, 1.7]$	$[-2.9, -2]$
5th	$k_1 = -0.57, k_2 = 0.13, k_3 = -0.57, k_4 = 0.53, k_5 = -0.14$	$[1.4, 2.2]$	$[-3.4, -2]$

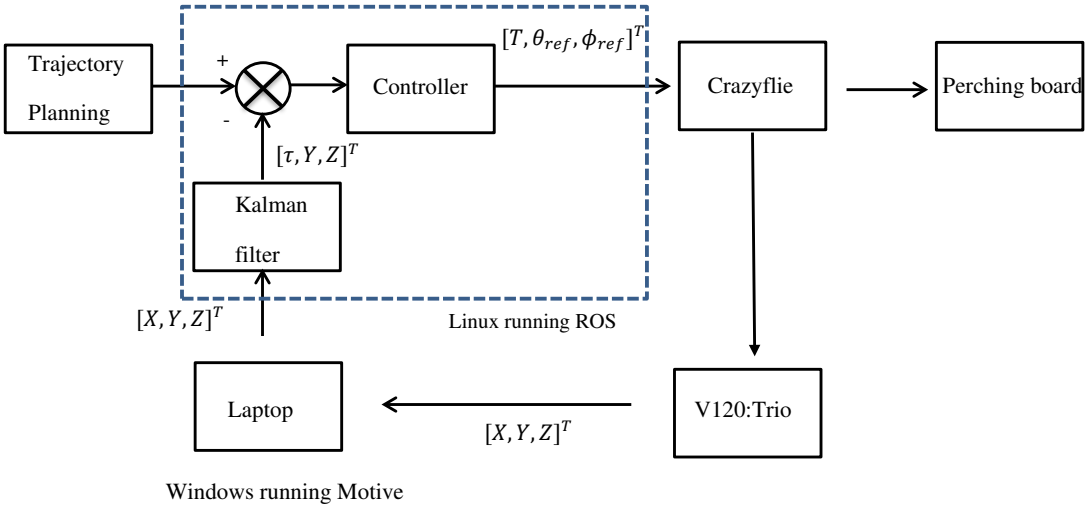
Table 3.2: Feasible initial conditions for different IPTS strategies

and maximum acceleration constraints. Thus the goal of safe perching can still be fulfilled. In this case, a fixed set of coefficients k_i can be found for a given range of initial conditions. Further, the range will expand if the order of the polynomial in IPTS increases. We have tested several examples for polynomials of different orders as shown in Table 3.2. From the simulations, we can find many sets of ks to satisfy the constraints. For a 3rd order polynomial, one example set is $k_1 = -0.60$, $k_2 = -0.057$, $k_3 = 0.054$. With this set of k, as long as the initial condition is $Z_0 \in [-3.88, -3.73]$ and $V_0 \in [1.89, 1.99]$, the constraints we used in the experiments can be satisfied. A 4th order polynomial is also investigated, and the allowable range for Z_0 and V_0 will increase to $[-2.92, -2]$ and $[1, 1.65]$, respectively. A 5th order polynomial will further increase the range for Z_0 and V_0 to $[-3.43, -2]$ and $[1.4, 2.15]$, respectively. Note that the ranges for V_0 and Z_0 shown in Table 3.2 are generated from a given set of ks. We can also find other sets of ks to generate different ranges. Therefore, we believe that if the initial conditions can be roughly estimated, then we can find a fixed set of ks to satisfy the constraints as long as the estimated initial conditions fall inside the range for the set of ks. Fortunately, estimating initial distance and velocity from TTC is highly possible as demonstrated in some recent research results [61, 63].

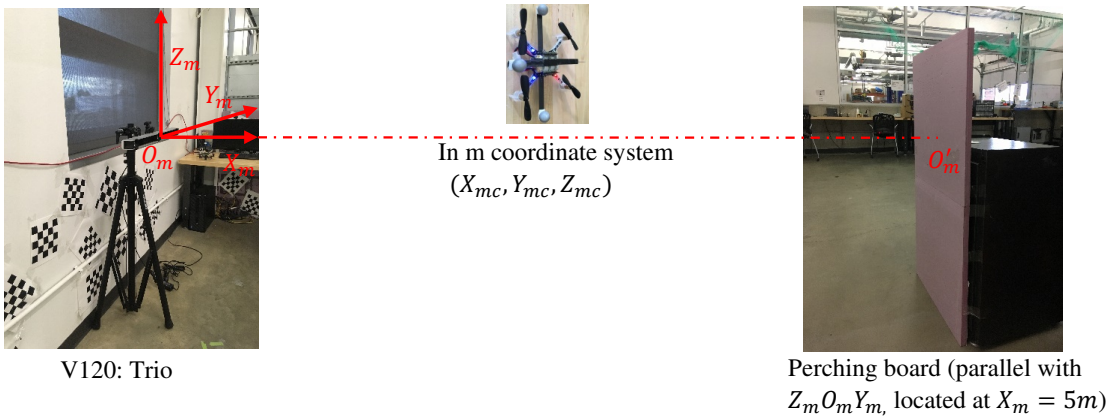
3.4.2 Experimental results

After simulations, we experimentally test the tau based trajectory planning and control to enable perching for aerial robots. Although our ultimate goal is to leverage vision as feedback to estimate tau [49,53,55,72], the hardware cannot provide accurate estimation with demanded computation time. In our initial step, we use a motion tracking system (V120: Trio, Optitrack) which can provide more accurate and faster feedback compared with monocular camera to track the position and orientation of the Crazyflie in the world frame. The V120: Trio is a pre-calibrated motion tracking system with three high speed cameras, of which the estimation error can be less than $1\,\mathrm{mm}$. The largest distance it can track with larger markers is about $5.2\,\mathrm{m}$. The quadcopter, Crazyflie, only weighs about $27\,\mathrm{g}$ (without markers for motion tracking) with a size of 92×92×29 mm. Due to the small size and light weight, it is very safe for indoor experiments. Currently, the perching mechanism is realized by a needle placed in front of Crazyflie, although more sophisticated mechanisms will be investigated in the future. A vertical foam board serves as the perching surface.

The experiment scheme is shown in Figure 3.4a. A motion tracking coordinate system is defined as shown in Figure 3.4b. In this system, origin is set at the optical center of the middle camera, X axis is defined along the optical axis of the middle camera, Z axis is defined upward, and Y axis is defined based on the former axis. The motion tracking system feedback the robot's position, which is sampled by a laptop running windows with Motive—a software provided by Optitrack. The same position is then transmitted to a desktop running Linux with Robot Operating System (ROS). On this desktop, we first estimate the current τ using $\tau = X/\dot{X}$, where X is the distance along X_m axis between the Crazyflie and the perching board placed at 5 meters away from the motion tracking system. And the velocity $\dot{X}$ is obtained from the position X using a Kalman Filter. Based on the estimated τ and position in Y and Z, we compute the control command based on the controllers in equations (3.17) and (3.18), which will then be mapped to $[T, \theta_{ref}, \phi_{ref}]^T$ through the nonlinear transformation. The computed $[T, \theta_{ref}, \phi_{ref}]^T$ is then wirelessly transmitted to the Crazyflie through the Crazyradio to control its motion.



(a) Experiment scheme



(b) Coordinate system setup

Figure 3.4: Experiment scheme and motion tracking system coordinate setup. The coordinate system origin O_m is projected as O'_m at the center of the perching board along X_m axis. The position and orientation of Crazyflie are measured by motion tracking system. The distance X between Crazyflie and the perching board is calculated by $X = X_{mc} - 5$. Then τ is calculated by definition and control command is generated by the τ controller and position controller.

For the perching experiments, we use the same initial conditions with the simulation: $X_0 = -3\,\mathrm{m}$, $\dot{X}_0 = 1.5\,\mathrm{m/s}$, $V_u = 1\,\mathrm{m/s}$, and $V_l = 0.7\,\mathrm{m/s}$. The constraints are also the same based on the Crazyflie's capability: $V_{max} = 2.5\,\mathrm{m/s}$, and $|a_{max}| = 1.4\,\mathrm{m/s^2}$. Note that even the maximum acceleration of the Crazyflie and velocity are larger than the chosen constraints, the delay from wireless transmitting the control command confines the control performance. The constraints and perching velocity range are obtained after several tests with a manual remote controller. To make the Crazyflie have the specific initial conditions, we first accelerate the initially hovering Crazyflie for a while. When $X_0 \approx -3\,\mathrm{m}$ and $\dot{X}_0 \approx 1.5\,\mathrm{m/s}$ are satisfied, we initiate the first stage of the two-stage tau based strategy at time t_0. When the feedback tau of Crazyflie is $\tau = \tau_s = -0.5\,\mathrm{s}$, the second stage is initiated at time t_s. After that, the Crazyflie is controlled to perch on the foam board with a constant speed.

The PID parameters for the controllers are selected as follows: $k_{px2} = 0.4k_{px1} = 31000$, $k_{ix} = k_{dx} = 2500$, $k_{py} = 14500$, $k_{iy} = 2000$, $k_{dy} = 4500$, $k_{pz} = 20000$, $k_{iz} = 1500$, $k_{dz} = 3500$, which are tuned based on the thrust signal range which is in $(0, 65535)$. Similarly, the saturation function is selected as:

$$\sigma_\phi(\omega) = \sigma_\theta(\omega) = \begin{cases} -25^\circ, & \text{if } \omega \leq -25^\circ \\ \omega, & \text{if } -25^\circ < \omega < 25^\circ \\ 25^\circ, & \text{if } \omega \geq 25^\circ \end{cases} \tag{3.25}$$

$$\sigma_T(T) = \begin{cases} 10000, & \text{if } T \leq 10000 \\ T, & \text{if } 10000 < T < 65000 \\ 65000, & \text{if } T \geq 65000 \end{cases} \tag{3.26}$$

We conduct five experiments for both the CTDTS and the IPTS, respectively. CTDS is not tested since it cannot achieve non-zero contact velocity required for perching. We fill all the five experimental results area with yellow and plot the mean value of the five experimental results in red. The results of CTDTS, with a total perching time of $3.24\,\mathrm{s}$, are shown in Figure 3.5a, 3.5c and 3.5e,

while the results of IPTS, with a total perching time of $2.90\,\mathrm{s}$, are shown in Figure 3.5b, 3.5d and 3.5f.

From the figures, we can see that both CTDTS and IPTS can be used for non-zero contact velocity perching. IPTS can generate faster perching while satisfy more constraints. For CTDTS, after t_0, there is still a small period that the Crazyflie keeps increasing speed. It is because from $t = 0\,\mathrm{s}$ to t_0, the Crazyflie is always accelerating and it cannot respond fast enough to decelerate immediately. Despite this small period, it is almost decreasing by controlling the tau to follow the reference. Note that there is a period of increasing speed because of the tracking performance of the tau controller. When $\tau_{ref} > \tau$, the Crazyflie increases its speed to increase the actual τ and vice versa. Finally, it follows τ_{ref} very well and the contact speed is about $0.97\,\mathrm{m/s}$. On the other hand, for the IPTS, similar to the simulation results, the speed first increases then decreases and finally contact the surface with a speed of about $0.91\,\mathrm{m/s}$. Also the τ tracks the reference value quite well, although there exists some discrepancy after t_0 which might due to the delay of the Crazyradio. For both CTDTS and IPTS, the tau reference in the second stage can be used for controlling the aerial robot to fly with almost a constant speed even though the speed slightly decreases which is again caused by Crazyradio delay.

3.5 Chapter Summary

Tau theory has been widely applied for robot motion control for tasks such as landing and docking. In this chapter, we propose two TTC or tau based two-stage strategies to realize perching for aerial robots with a non-zero contact velocity. Specifically, we design CTDTS and IPTS as the tau reference and develop the corresponding control laws. Simulation results have shown that both CTDTS and IPTS can accomplish a non-zero contact velocity with IPTS being able to satisfy more constraints and generate a shorter perching time. Furthermore, perching experiments with a palm-size quadcopter also validate the faster perching of the IPTS.

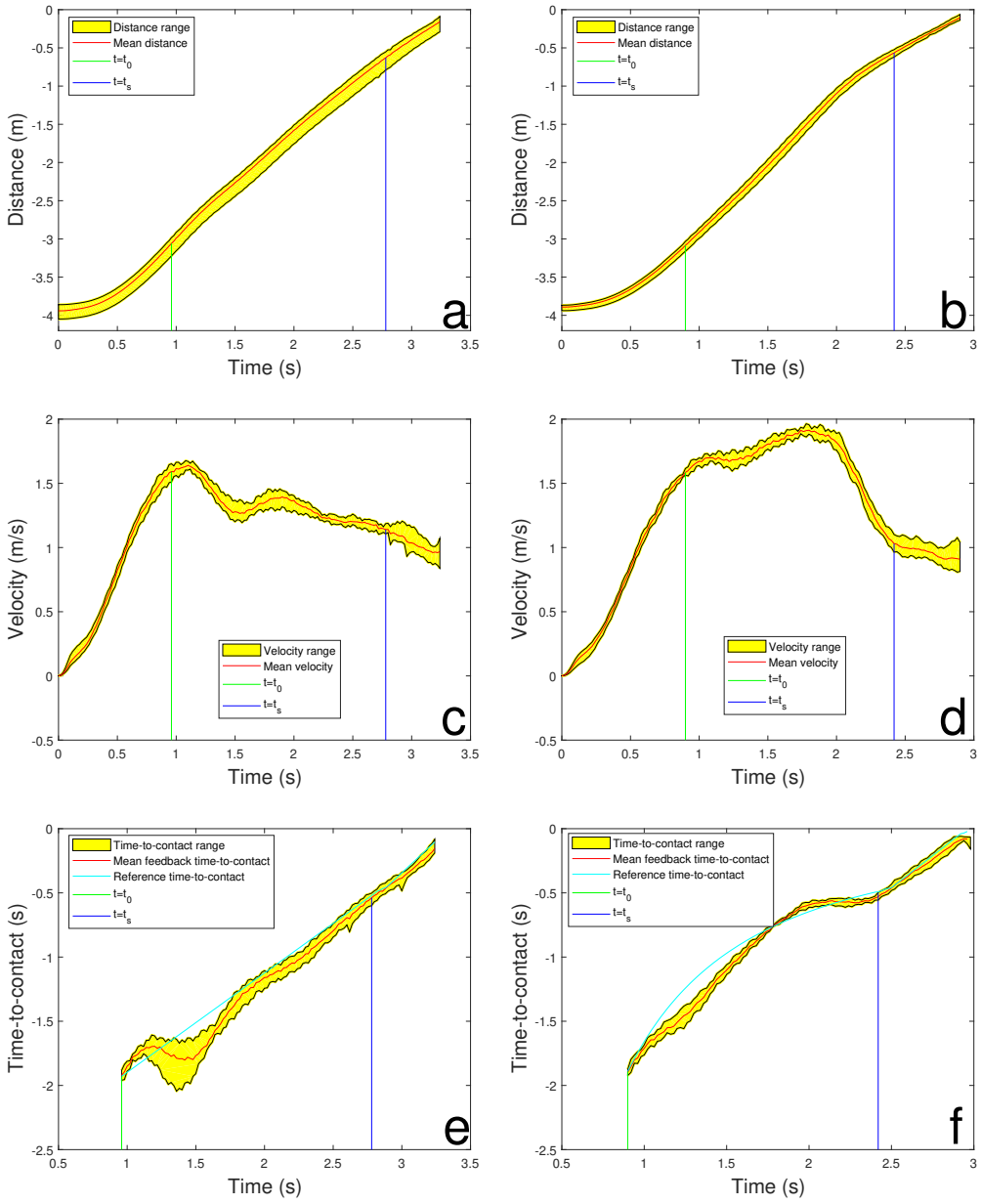


Figure 3.5: Experiment results for CTDTS (left column) and IPTS (right column). From top to the bottom are the distance, velocity, and tau for CTDTS and IPTS, respectively. The CTDTS and IPTS can realize the nonzero contact velocity, and IPTS requires shorter time for perching.

Chapter 4

Mechanical Intelligence: Gripper Design

With the proposed trajectory planning algorithm discussed in last Chapter, we can realize non-zero contact velocity. However, we used a needle as a perching mechanism in last chapter. In this chapter, we detail the design and analysis of a compliant bistable mechanism. The gripper has two unique characteristics. First, using bistability, it can passively switch from open to closed state using impact between the gripper and the perching object, eliminating additional sensors to detect the collision. Second, the gripper has two perching methods for different objects. For objects with a small height, the gripper can form a closed diamond shape to encircle the objects (encircling method). For objects with a large height, the gripper's two fingers can clip on each side of the objects to utilize the friction forces for perching (clipping method). We analyze the proposed gripper design to predict the required force for opening and closing the gripper. We also predict the size of objects that will allow for successful perching for the clipping method. All the theoretical analysis are experimentally verified. Finally, we integrate the mechanism onto a palm-size quadcopter, and demonstrate successful perching with both clipping and encircling methods onto different objects.

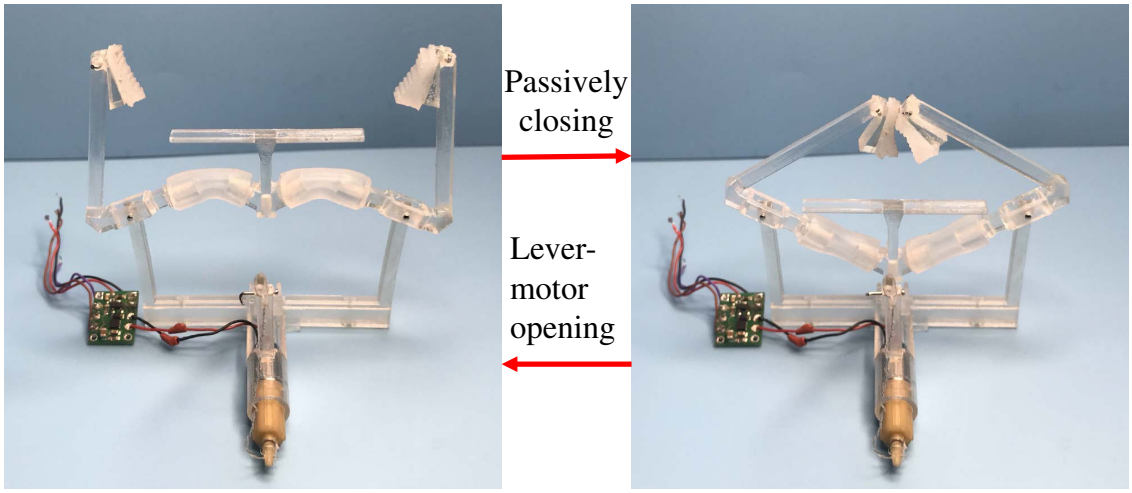
4.1 Introduction

In aerial robot perching tasks, besides the estimation, planning, and control, it is equally important that a lightweight and reliable perching mechanism should be designed to be rigidly attached to and easily released from the perching objects. Before introducing the compliant bistable gripper in Figure 4.1. We first briefly review several recent research on perching mechanism design (a detailed review can be found in [8, 73]). Based on the perching objects, we can categorize most perching methods into surface perching and rod perching. Surface perching means the perching object is a flat surface such as wall and ceiling, while rod perching means the object resembles a rod shape (e.g., tree branches). *For surface perching,* adhesion pad and microspine are widely used.

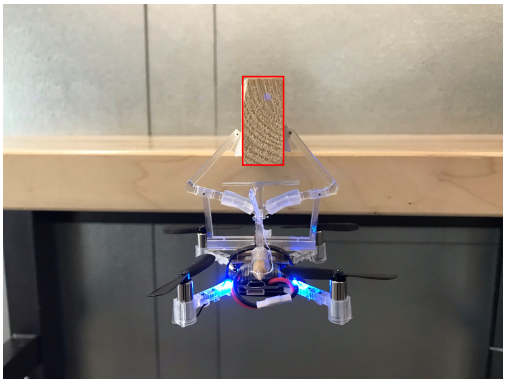
For instance, electrostatic adhesion is adopted in [9] to perch and detach the Robobee on surfaces with different materials. Anderson controlled a fixed-wing MAV to adhere itself to the perching surface with a sticky pad [74]. A perching mechanism using fiber-based dry adhesives and passive self-alignment system is implemented on a $300\,\mathrm{g}$ flying platform [75]. Kovač *et al.* presented a $4.6\,\mathrm{g}$ perching mechanism that could convert the impact into the snapping motion to stick needles into the surface [10]. Recently, they also proposed a spider inspired tensile anchoring modules to launch several tensile anchors on fixed objects to perch the MAV [76]. Mehanovic *et al.* proposed a bird-like pitch up strategy for the fix-wing drone to decrease the impact force and adjust the perching orientation [16]. Using gecko-inspired adhesive grippers, Thomas *et al.* controlled a MAV to perch on inclined surfaces [13]. Stanford Climbing and Aerial Maneuvering Platform (SCAMP) was developed for perching, climbing and taking off again [8]. *For rod perching*, a perching mechanism with grasping capability is usually adopted. For instance, a songbirds-inspired perching mechanism utilizes the weight of MAV to passively apply tendon tension to actuate the gripping foot [77]. Nguyen *et al.* designed a passively adaptive microspine grapple that could conform to the surface of convex perching targets such as tree branches [78]. Hang *et al.* designed a set of actuated landing gears which enables MAVs to perch or rest on many different objects [79].

In this chapter, we present a bistable gripper design, which can switch between a stable open state and another stable closed state. Such a design has two advantages for perching compared with existing methods [8]. On one hand, it can leverage the impact force during the perching to passively perch, increasing the robustness of the mechanism and eliminating the requirement for a sensor to detect the impact and an actuator to close the gripper. On the other hand, the gripper does not require additional energy input to maintain the stable states, making it ideal for applications requiring long-duration monitoring or surveillance.

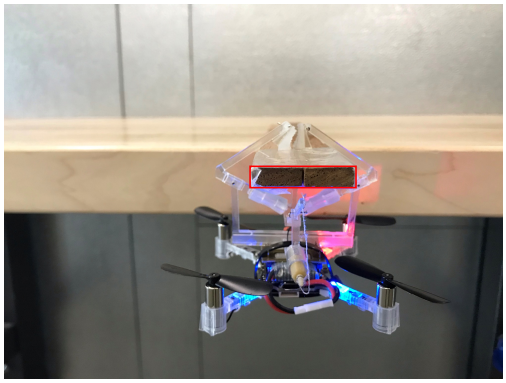
Bistable mechanisms have been widely used in different areas [80]. For grippers, Nguyen *et al.* designed a bistable gripper for grasping and releasing objects [81]. Thuruthel *et al.* [82] designed a soft bistable gripper to rapidly grasp unstructured objects. Besides gripper, the bistability for carefully designed mechanical structures has also been recently exploited for various applications



(a) Gripper 2.0



(b) Clipping perching



(c) Encircling perching

Figure 4.1: Proposed bistable gripper with two perching methods. (a) the two stable states of the gripper. It can be passively switched from open to closed state through impact force. It can switch from closed to open using a lever-motor system. (b) the clipping perching method which utilizes the friction force to hold the robot's weight. (c) the encircling perching method which relies on the closed diamond shaped formed by the fingers to hold the robot's weight.

including deployable structures [83, 84], jumping robots [85, 86], swimming robots [87], origami robots [88, 89], soft robots [90], shape morphing [91], and mechanical metamaterials [92, 93]. To the best of our knowledge, it has yet to see how bistability can be used for perching mechanisms except our previous work [94, 95].

There are three major contributions in this chapter. First, we utilize the concept of bistability to design a novel gripper, which suits MAV perching well since it can rely on impact forces to trigger the perching process, and does not require additional energy after perching. Second, we thoroughly analyze the bistability of the mechanism to generate a design guideline for selecting

proper design parameters. Third, we develop a complete mechatronics system for aerial perching by integrating the gripper onto a palm-size quadcopter, which can perform repeatable perching and releasing.

The rest of this chapter is organized as follows. Section 4.2 introduces the working principle of bistable mechanisms and the new gripper design. Section 4.3 analyzes the mechanism by deriving the equation for force-displacement characteristics and equation for grasping forces of the bistable gripper. The bistability of the gripper based on two important parameters is also discussed. Section 4.4 details the experiment setup and results to verify the force-displacement characteristics modeling and demonstrate the repeatable perching-releasing cycle for two different perching methods. Section 4.5 summarizes the chapter.

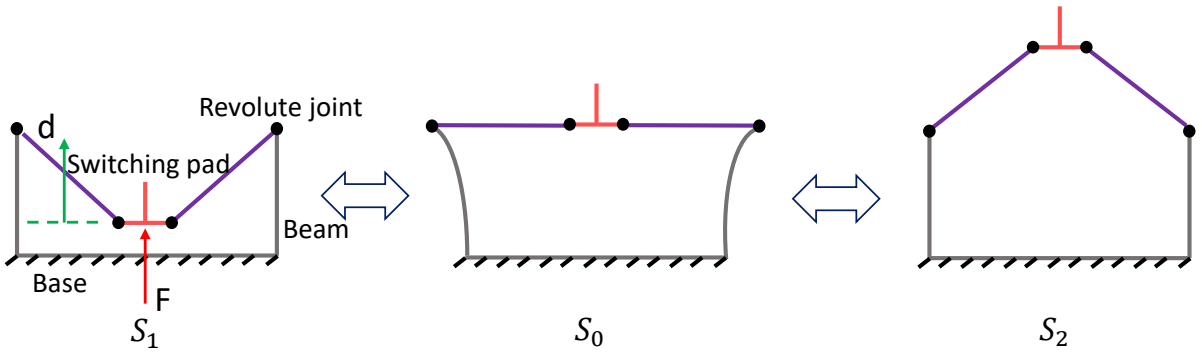


Figure 4.2: Schematic of a basic bistable mechanism for our bistable gripper. It has four revolute joints in black, two rigid links in purple, a switching pad in pink, and two beams (together with the base) in grey. It has two stable states at S_1 and S_2. When an upward force F is applied on the switching pad, the mechanism can switch from S_1 to S_2 through the intermediate state S_0. During the process, the two vertical beams will be pushed outside. If the force is removed, it can switch to the closest stable state S_1 or S_2 with the recovery forces generated from bending beams. And vice versa, a large enough downward force on the switching pad can make the mechanism switch from S_2 to S_1 through S_0.

4.2 Bistable gripper design

Our gripper design is based on a basic bistable mechanism as shown in Figure 4.2. The mechanism consists of four revolute joints, two rigid links, a switching pad, and two beams connected to a rigid base. It has two stable configurations illustrated as S_1 and S_2, where no force input is

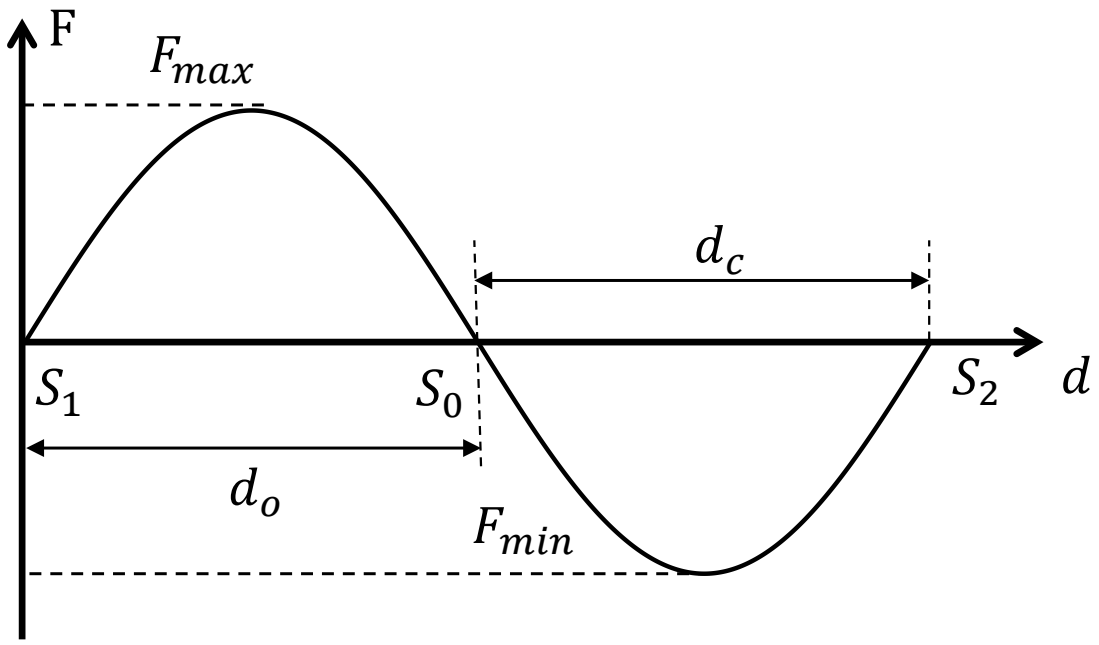


Figure 4.3: Force-displacement characteristic for the basic bistable mechanism. The figure shows the force required to maintain the switching pad at a specific travel distance. The maximum force for this transition from S_1 to S_2 is F_{max} and the minimum force is F_{min}. The displacement the switching pad needs to travel is d_o from S_1 to S_0 and d_c from S_0 to S_2.

needed to maintain the configuration. It can switch between S_1 and S_2 through external forces. For instance, it can switch from S_1 to S_2 when a force F is applied upward on the switching pad. During the process, the rigid links will rotate, and the two beams will be bent outside. When it arrives configuration S_0, no force is needed to maintain the current unstable state. After passing S_0, an opposite force is required to maintain the current configuration. Otherwise, it can switch to S_2 with the recovery force from bending beams. And vice versa, a downward force can also be applied on the switching pad to make the mechanism switch from S_2 to S_1 through S_0. The relationship between the force F and displacement d is called force-displacement characteristic as shown in Figure 4.3. In the figure, we have two critical forces: the maximum force F_{max} and the minimum force F_{min}, meaning the force needs to be larger than F_{max} to switch from S_1 to S_2, larger than $|F_{min}|$ to switch from S_2 to S_1. d_o and d_c are the corresponding displacements between two neighboring zero forces.

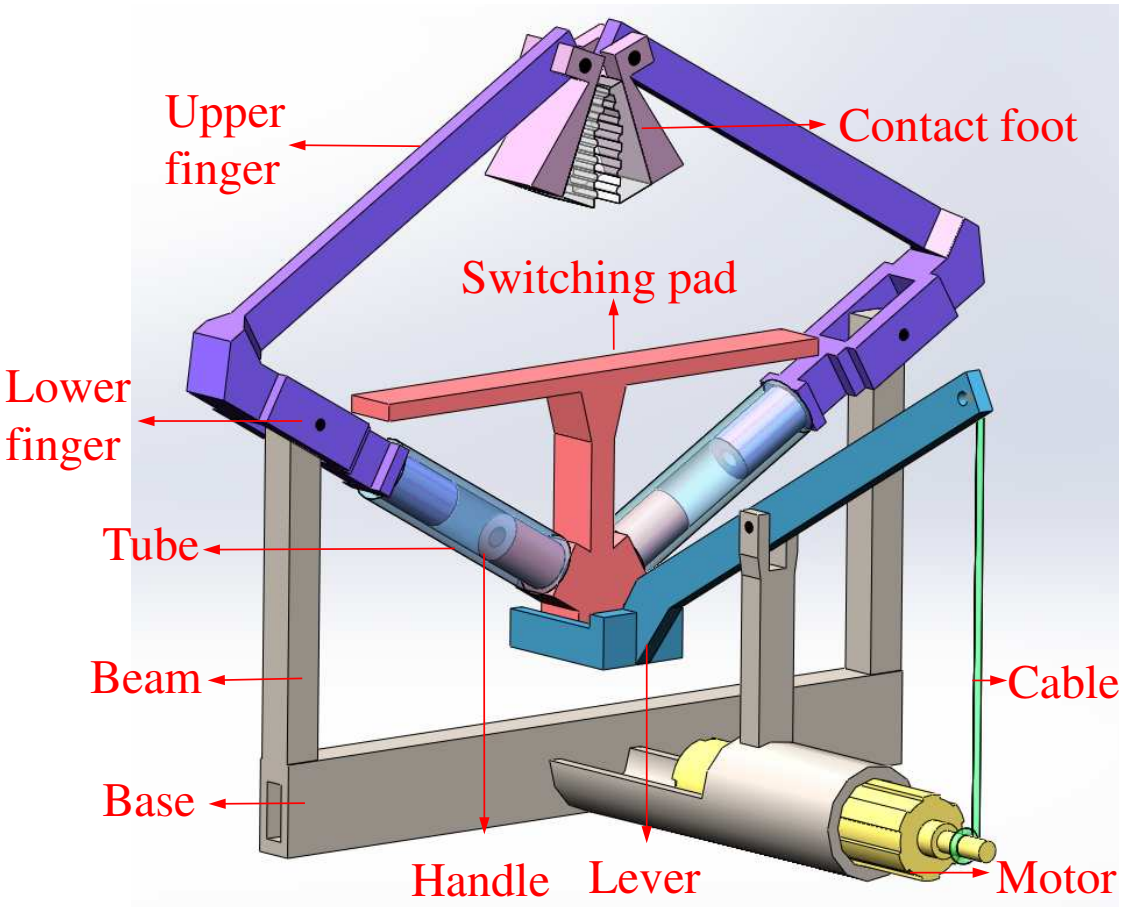


Figure 4.4: Solid model for bistable gripper in the closed stable state. The gripper consists of a base with beams, fingers, a switching pad, contact feet, tubes, and a lever-motor releasing system. One side of the lever can be dragged by the motor while the other side will push the bottom of the switching pad upward to open the gripper.

Our new design is based on the basic bistable mechanism (Figure 4.2) with a key difference: the basic mechanism has a symmetric force-displacement characteristic (i.e., $F_{max} = |F_{min}|$), but our bistable gripper can have a tunable force-displacement profile with different magnitudes of F_{min} and F_{max} that can deal with the requirements of perching: easy to close (a small impact force can close the gripper), and stable to hold (only a large force can open the gripper). Such tunable performance is accomplished by replacing the two revolute joints attaching to the switching pad with elastic/compliant joints. Details on why such a change can lead to tunable performance are elaborated in section 4.3.3.

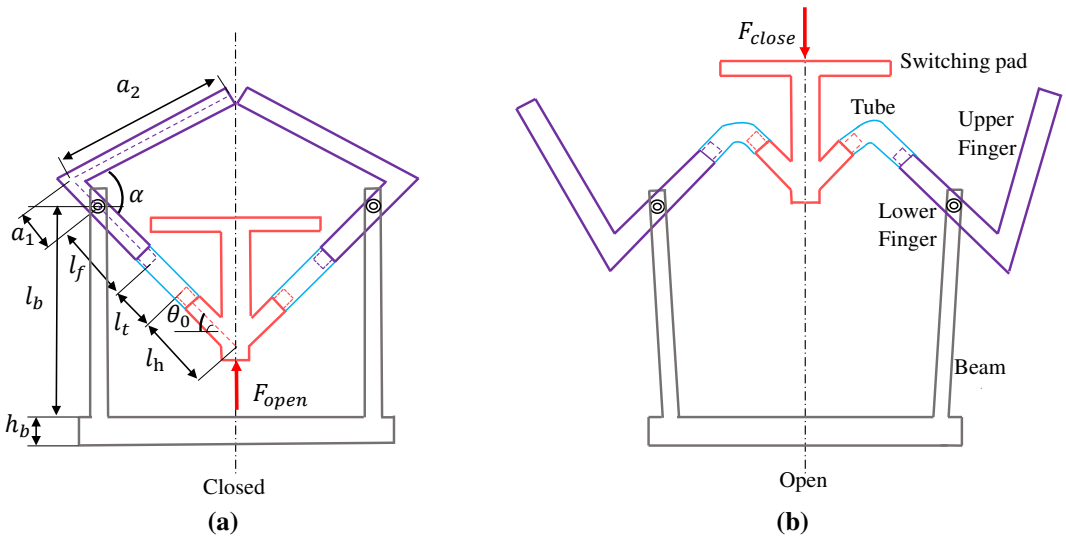


Figure 4.5: The schematic of the bistable gripper in two stable states: The left figure is the closed state, where no strain energy is in the tubes or beams. A force F_{open} can be applied on the bottom of the switching pad to open it. The right figure is the open state, where strain energy is stored in both beams and tubes with beams being pushed outward and tubes being bent. A force F_{close} can be applied on the top of the switching pad to close it. Some unnecessary parts such as the contact feet and lever-motor system are not shown for simplicity

.

Our new design can be illustrated using a solid model (Figure 4.4). It consists of the following elements: a base with two vertical beams, two fingers (consisting of lower finger and upper finger) connected to the beams via rotational joints, a switching pad that will contact the perching object to close and be pushed by the lever to open, two contact feet attached to the end of the two upper fingers, two elastic tubes connecting the fingers to the switching pad, and a cable-driven lever actuated by a DC motor. We choose tubes as compliant joints instead of traditional torsional springs since tubes can be easily customized to different lengths to generate different torsional stiffness. Also, the tubes are chosen as the compliant joints for convenient fabrication [96], but it can be replaced with any compliant materials (e.g., multimaterial 3D printing [97]).

With such a design, the gripper has a closed stable state and an open stable state as shown in Figure 4.1a. It can switch from the initially closed state to the open state as follows. If an upward force from the cable-driven lever is applied on the bottom of the switching pad, the two fingers will pivot around the rotational joints to push the two vertical beams outward and bend the two tubes,

storing strain energy in the beams and tubes. When the switching pad passes a critical point, the stored strain energy will push the switching pad upward without any input. Eventually, the gripper will stop at the open stable state. Similarly, the gripper can switch from the open state to the closed state by applying a downward force on the top of the switching pad. For perching, this force can directly come from the impact force.

Compared with our old gripper [94], we revised the design in three major aspects. First, the old gripper has three beams on the base and three fingers rotating on each beam. Although the three fingers can form a closed shape to hang on a perching object, they complicate the perching preparation of the old gripper since the MAV needs a good yaw angle to initiate the perching. Only in the desired range for the yaw angle could the fingers grab the perching object successfully. To address this issue, we use only two fingers rotating on two beams in the new design. Such a design can increase the range of perching yaw angle for about 60°. Second, besides the encircling perching method that the old gripper utilizes, the new gripper can also perch with a new clipping method for large objects that the gripper cannot encircle. As shown in Figure 4.1b, when the new gripper is closed, the two feet can contact the two sides of the object to generate friction forces to clip on the object. To generate large enough friction forces, each foot is composed of a soft film made from elastomers glued onto a rigid base, which can freely rotate about the upper finger. With the rotational feet, the gripper can adapt to different shaped objects in natural environments. Third, we design a new lever-motor system to open the new gripper to release the MAV from the perching state. In the old gripper, the releasing mechanism is based on heat and not repeatable. As shown in Figure 4.4, the new lever-motor system consists of a lever driven by a motor through a cable. One side of the lever is connected to a motor shaft through a cable, while the other side is under the switching pad. After opening, the motor will be controlled to rotate in the opposite direction to release the cable, and the lever will switch back to the original position due to the heavier switching side.

4.3 Bistable gripper analysis

To use the designed gripper for perching, we need to analyze the relationship between the force applied to the switching pad and the resulting displacement: the force-displacement characteristic. Such a relationship can determine the required force to switch between the two stable states. After that, we derive the friction force that can be generated by the gripper for clipping perching. In the end, we investigate the underlying working principle and show how the design parameters will influence the mechanism's bistability by defining a bistability index.

4.3.1 Force-displacement characteristic

To analyze the force-displacement characteristic, we redraw a simplified sketch of the new gripper in Figure 4.5 for both open and closed state. Since the lever-motor system does not influence the force-displacement characteristic, it is not drawn in the sketch. As shown in Figure 4.5, l_h is the distance from the end of the switching pad handle to the centerline of the switching pad. l_t is the length of the hollow tube. l_f is the distance from the pivot to the end of the lower finger. a_2 is the upper finger's length, while a_1 is the length from the centerline of the upper finger to the pivot. The angle between the upper and lower finger is α. The distance from the pivot to the base is l_b. The angle between the handle of the switching pad and the horizontal direction is θ_0. h_b is the height of the base.

The bistability is generated by the deformation of the elastic tube and the vertical beam (see section 4.3.3 for a detailed analysis). Therefore, we will model the statics for the gripper by considering these two elements. For the flexible tube, we model it with the PRBM [80], a widely used technique for compliant mechanisms. Specifically, we model the tube as two rigid links connected by a rotational joint. Note that there are more complicated models with more joints [97] as well as models with Beam Constraint Model (BCM) which takes into account the nonlinearities arising from load equilibrium applied in the deformed configuration [98], but we use the PRBM for simplicity. To represent the tube's resistance to bending, we assume a torsional spring associated with the joint. The joint locates at γl_t away from the end connected with the fingers, where l_t

is the tube length. Detailed γ values can be found in [80]. In this paper, we use $\gamma = 0.83$ to maximize the pseudo-rigid-angle while achieve an accurate estimation [80]. The spring constant for the torsional spring is $k_\theta = \pi\gamma^2 E_{yt} I_t / l_t$, where E_{yt} and I_t are Young's modulus and second moment of inertia of the tube, respectively. For the beam, we model it as a linear spring which can only be compressed in the horizontal direction, since its outward displacement is small. The spring constant is $k_d = 3E_{yb} I_b / l_b^3$, where E_{yb} and I_b are Young's modulus and second moment of inertia of the beam, respectively. We ignore the change of l_b in vertical direction since the change is estimated to be only 0.63% of the original beam length in our design.

With the above models for tubes and beams, Figure 4.5 can be redrawn in Figure 4.6 for mathematical derivation. The green lines represent the initial closed configuration C_0, and the red lines represent one of the configurations during state transition C_1. Due to the symmetry of the gripper, the switching pad at the centerline can only move in the vertical direction with displacement d.

Since the applied force F is the only input and the switching process is quasi-static, the force-displacement characteristic between F and d can be derived from the total strain energy E in linear springs for beams and torsional springs for tubes through the following equation [87]:

$$F = \frac{\partial E}{\partial d} \tag{4.1}$$

From the assumptions about linear and torsional springs, the strain energy in the two beams can be written as

$$E_b = k_d {d_b}^2$$

where d_b is the horizontal displacement of the linear spring. It can be solved from the following geometrical relationship

$$H^2 + {L_0}^2 = (H - d)^2 + (L_0 + d_b)^2$$

where $L_0 = (l_f + \gamma l_t)\cos\theta_0$, $H = (l_f + \gamma l_t)\sin\theta_0$ are constants (Figure 4.6). With this equation, we can solve d_b as a function of d: $d_b = \sqrt{2Hd + {L_0}^2 - d^2} - L_0$. The strain energy in the two tubes can be written as

$$E_t = k_\theta(\theta_1 - \theta_0)^2$$

where θ_1 is the angle between the lower finger and the horizontal axis at configuration C_1, which can also be represented as a function of d: $\theta_1 = \arctan{(H-d)/(L_0+d_b)}$. Therefore, the total strain energy E is

$$E = E_t + E_b = k_\theta(\theta_1 - \theta_0)^2 + k_d {d_b}^2 \tag{4.2}$$

Plugging the energy into equation (4.1), we can obtain the force-displacement characteristic as

$$F(d) = -\frac{2}{L_0 + d_b}\left[-k_d d_b(H-d) + k_\theta(\theta_1 - \theta_0)\right] \tag{4.3}$$

4.3.2 Friction force

Our new gripper has a new perching method called clipping, for which friction forces are utilized for perching onto objects with a large height. To ensure successful clipping, it is necessary to analyze the friction force generated by the two contact feet for a perching object with a given size.

The clipping scenario is also depicted in Figure 4.6, where a rectangular purple object is placed vertically with the contact feet clipping on it. We assume the surface of the perching object is flat and in the vertical direction. When the gripper is closed, its fingers will contact the surfaces to generate a normal force F_n, resulting in a friction force f that acts on the contact point to support the MAV's weight. a_3 is the distance from the beam pivot to the contact point, and φ is the angle between the horizontal direction and a_3 (Figure 4.6). a_3 can be solved based on the geometric relationship shown in Figure 4.5.

$$a_3 = \sqrt{a_1^2 + a_2^2 - 2a_1 a_2 \cos\alpha} \tag{4.4}$$

φ can also be solved similarly based on the geometric relationship shown in Figure 4.6.

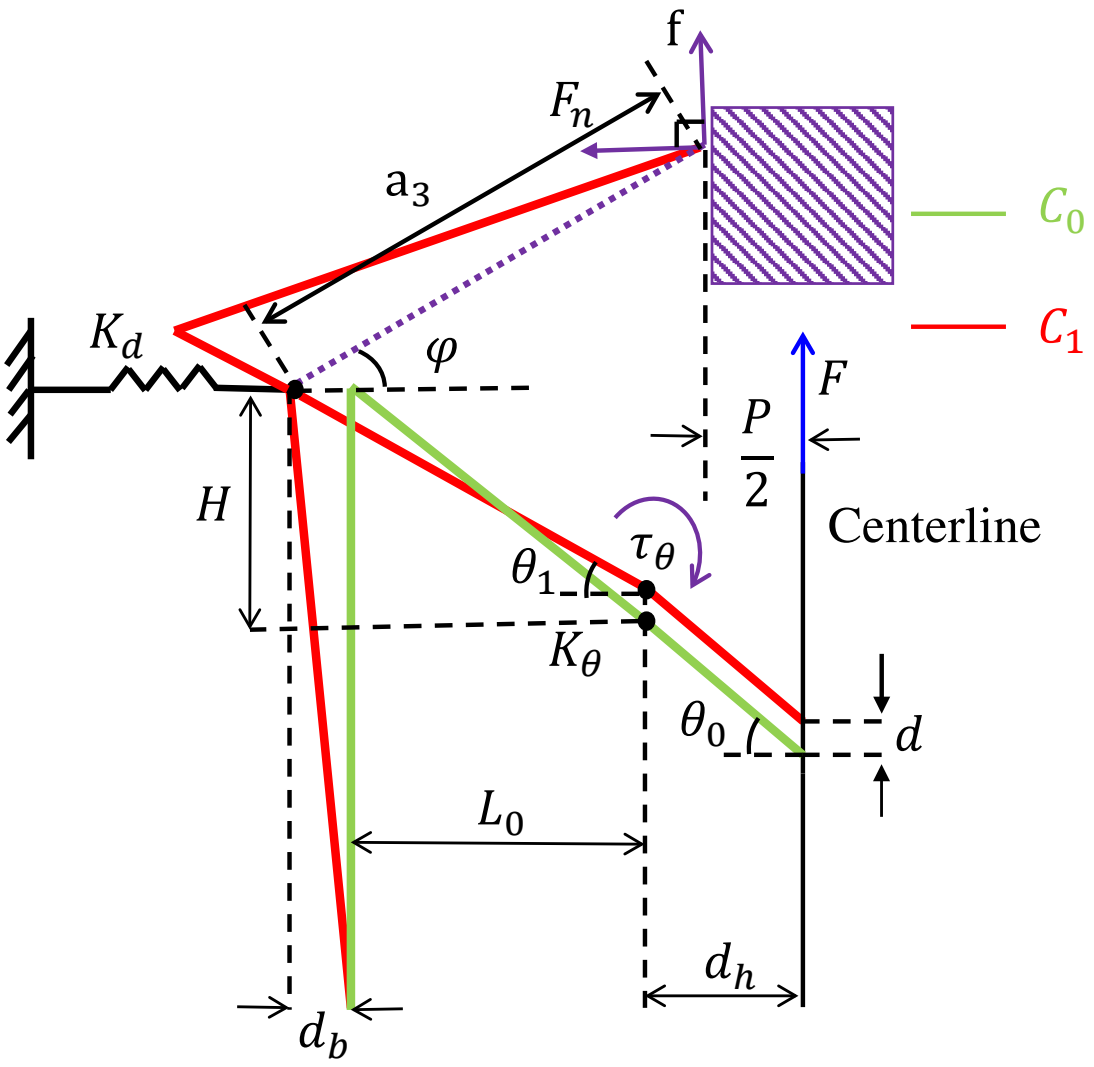


Figure 4.6: Sketch for mathematical modeling of the gripper. Only the left part of the gripper is shown. The bending beam is modeled as a linear spring, and the tube is modeled as a torsional spring. The green lines represent the initial closed state C_0 and red lines represent one of the configurations during state transition C_1. For clarity, the upper fingers are not drawn in C_0. For the clipping scenario, a purple rectangle is drawn as the perching object and normal force F_n and friction force f are drawn in purple at the contact point.

$$\varphi = \pi - \arccos \frac{a_1^2 + a_3^2 - a_2^2}{2a_1a_3} - \theta_1 \tag{4.5}$$

For a given design of the gripper, the size of the perching object can determine whether the perching is successful or not. Therefore, we need to solve the range of sizes for the object that will allow for successful perching. To do this, we first derive the vertical displacement d for the switching pad given the object's size P. Then, we obtain the normal force F_n from d. Finally,

we can determine successful perching by checking if $\mu F_n \geq f = mg/2$, where μ is the friction coefficient, m is the mass of the MAV.

The relationship between the object size P and the displacement d can be obtained from the geometrical relationship (Figure 4.6)

$$P = 2\left(d_b + L_0 + d_h - a_3 \cos\varphi\right) \tag{4.6}$$

where $d_h = [(1-\gamma)l_t + l_h]\cos\theta_0$. From this equation, we can numerically solve d given P since d_b and φ are functions of d. To obtain the normal force F_n from d, we analyze the statics using free body diagram for fingers of the gripper. As shown in Figure 4.6, there are four torques acting on the finger: recovering torque from the tube in clockwise direction τ_θ, torque generated by F_d from linear spring acting on tube pivot τ_{k_d} in clockwise direction, torque generated from F_n in counter-clockwise direction τ_{F_n}, and torque generated by f in counter-clockwise direction τ_f. If we assume MAV is able to perch on the object, i.e., $f = mg/2$, we can have the torque equilibrium equation

$$\tau_{F_n} + \tau_f = \tau_\theta + \tau_{k_d} \tag{4.7}$$

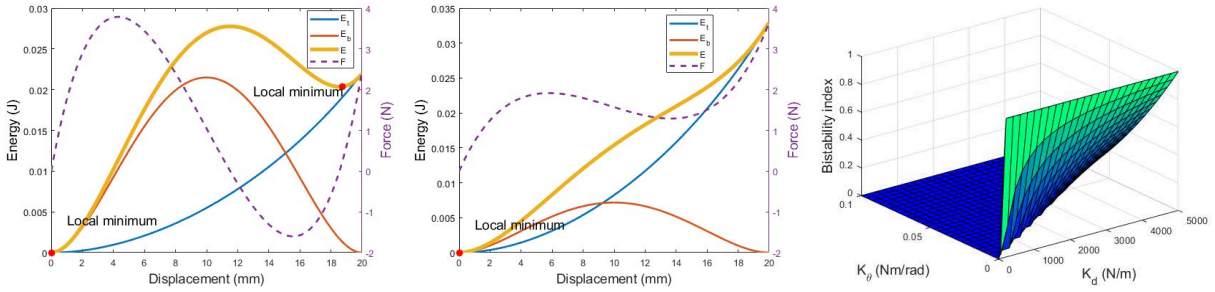
By solving equation (4.7), the normal force F_n can be obtained as a function of d

$$F_n(d) = \frac{k_d d_b(H-d) + k_\theta(\theta_0 - \theta_1) - mg(d_h - P/2)/2}{a_3 sin\varphi + H - d} \tag{4.8}$$

With $F_n(d)$, we can see if the clipping perching will be successful by checking if $\mu F_n \geq f = mg/2$.

4.3.3 Bistability analysis

If the parameters in equation (4.3) are not chosen appropriately, the mechanism may become monostable, meaning it only has one stable state. To provide design guidelines to generate the bistability required for perching, we investigate how two important design parameters will influence the bistability of this mechanism.



(a) **(b)** **(c)**

Figure 4.7: Bistability analysis. (a) The gripper is bistable if $k_d = 3000N/m$ and $k_\theta = 0.02Nm/rad$. (b) the gripper is monostable if $k_d = 1000N/m$ and $k_\theta = 0.03Nm/rad$. (c) Bistability index will change with respect to k_d and k_θ.

We first qualitatively investigate the reason for bistability using simulations. The bistability of the designed gripper is generated by the competition of potential energy from the tubes (E_t) and beams (E_b). To see this, we plot E_t and E_b as well as the total energy (E) for a bistable (Figure 4.7a) and monostable case (Figure 4.7b). From the two figures, E_t will monotonically increase because the tube will increasingly bend as the displacement increases. But E_b will increase first and then decrease because d_b (Figure 4.6) will first increase and then decrease. Combining E_t and E_b, the total energy $E = E_t + E_b$ can have either a single minimum at the initial configuration ($d = 0$, Figure 4.7b) or two minima (Figure 4.7a), with different choices of k_d and k_θ. Note that different k_d can be realized by choosing different thicknesses for the beam, while different k_θ can be achieved using tubes with different lengths.

In addition to the potential energy, we can also determine the bistable or monostable from the force-displacement characteristic. In the quasi-static state transition case, the force-displacement in Eqn. (4.3) is the first-order derivative of the potential energy, which can tell the direction of the potential energy curve. If the force is always positive, the energy will be monotonically increasing as in the monostable case (Figure 4.7b). If the initial positive force becomes negative at some d, the potential energy will decrease and have a local minimum as in the bistable case (Figure 4.7a). In other words, the system is bistable if there exists negative force in Eqn. (4.3) and monostable if $F \geq 0$ for all d.

With the observations for the force, we define a bistability index denoted as BI to numerically investigate how k_d and k_θ will influence the bistability

$$BI = -\frac{F_{min}}{F_{max}} \tag{4.9}$$

where F_{min} and F_{max} is the minimum and maximum force in the force-displacement characteristic of the bistable mechanism, respectively. For the gripper, $0 \leq BI \leq 1$. $BI = 0$ for all monostable mechanism since $F_{min} = 0$ at the initial configuration. $BI \leq 1$ means the magnitude of F_{min} is less or equal than F_{max}. This can be explained by looking at the slope of the energy curve. Because the decreasing of energy is only generated by E_b, the negative slope cannot be larger than the positive slope. The extreme case $BI = 1$ happens when $k_\theta = 0$, which means the tube is a traditional rotational joint without any torsional stiffness. This case will be the one illustrated in Figure 4.2.

To systematically explore how will k_d and k_θ influence the bistability, we plot BI with respect to k_d and k_θ as shown in Figure 4.7c. In the simulation, we have $k_d \in [0, 5000]$ with a step size of 50 N/m and $k_\theta \in [0, 0.1]$ with a step size of 0.001 Nm/rad. The plot indicates that larger k_d will increase the bistability index because E_b will dominate E_t, making the shape of the total energy closer to E_b with two minima (Figure 4.7a). Larger k_θ will decrease the bistability index, because E_t will dominate E_b, making the shape of the total energy closer to E_t with a single minimum.

4.4 Experiment

In this section, we detail the fabrication of the gripper and experimentally test the force-displacement characteristic and compare it with the theoretical results. We also verify the object sizes for successful perching using the clipping method. Finally, perching experiments using both encircling and clipping methods are carried out on different objects in both controlled and uncontrolled environments.

Table 4.1: Design parameters of the gripper

$a_1(\mathrm{mm})$	$a_2(\mathrm{mm})$	$\alpha(^\circ)$	$\theta_0(^\circ)$	$h_b(\mathrm{mm})$
9	37.8	60	30	6.5
$l_b(\mathrm{mm})$	$l_f(\mathrm{mm})$	$l_t(\mathrm{mm})$	$l_h(\mathrm{mm})$	
23.8	15	6	8.81	

4.4.1 Gripper fabrication and Perchflie

Most of the parts of the gripper are 3D-printed and then assembled. Five different parts (Figure 4.4) are 3D-printed using veroclear material with an Objet printer (Objet30 pro, Stratasys): one base with two vertical beams and the motor enclosure, two fingers, one switching pad, two contact feet, and one lever. The fingers are connected to the switching pad through a tube with an inner diameter of $1.5875\,\mathrm{mm}$ and an outer diameter of $6.35\,\mathrm{mm}$ (ULTRA-C-062-3, Sain-Tech). The film attached to the contact feet is fabricated from curable elastomers (Ecoflex30, Smooth-On). The DC motor (GH6124s, Gizmoszone) weighs less than $1.5\,\mathrm{g}$ and can provide $200\,\mathrm{g\,cm}$ torque. And the motor driver (DRV8838, Pololu) can provide a continuous current of $1.7\,\mathrm{A}$ with less than $1\,\mathrm{g}$ weight. The detailed design parameters for the gripper are shown in Table 4.1. The parameters are chosen to make the gripper easy to close but stable to hold. With the design parameters, the theoretical switching forces for two directions are $F_{max} = 2.15\,\mathrm{N}$ (opening force) and $F_{min} = -0.41\,\mathrm{N}$ (closing force) respectively. The gripper system weighs about $8\,\mathrm{g}$ including the motor driver. It is then attached to the Crazyflie (Crazyflie2.0, Bitcraze) using a zip tie as shown in Figure 4.8. The whole system, termed as Perchflie, is about $40\,\mathrm{g}$ including a flow deck on the bottom for stable motion control.

As shown in Figure 4.4, the lever is connected to the stand with a shaft. The whole length of the lever is about $52\,\mathrm{mm}$, of which both the pushing side and the dragging side is about $26\,\mathrm{mm}$. With this dimension, the force and travel distance are the same for both sides. A string is coiled on the motor and the other side is tied to the dragging side of the lever. With this lever-motor system, a full opening procedure requires about $2\,\mathrm{s}$ at the full motor speed.

Figure 4.8: Perchflie. Gripper is attached to the Crazyflie with a zip tie. The whole system is about $40\,\mathrm{g}$ including a flow deck.

4.4.2 Force-displacement characteristic experiment

To verify our mathematical model that predicts the activation force, we first conduct experiments to obtain the force-displacement characteristic. The experiment setup is shown in Figure 4.9. The main test machine is a motorized tension/compression test stand (ESM303, Mark-10). With a force gauge (M5-2, Mark-10) connected, the stand can move with a constant speed both upward and downward while measuring both tension and compression force. The measuring range of M5-2 is $10\,\mathrm{N}$ with a precision of $0.002\,\mathrm{N}$. And a software (MESUR^{TM} gauge Plus, Mark-10) is used for recording the force and displacement data.

We separate the experiments into two parts to minimize possible hysteresis: dragging for the opening force and pushing for the closing force. In the dragging experiment, the gripper starts with the closed state and ends at $0\,\mathrm{N}$ when no external force is needed to switch it to the open state. The switching pad is connected to the force gauge through a string. While the switching pad is dragged to move upward with a constant speed, the software records the displacement and force data. For the pushing experiment, the gripper starts with the open state and ends at $0\,\mathrm{N}$ when no external force is needed to switch it to the closed state. During the experiments, the force gauge moves downward to push the switching pad. 10 pushing and 10 dragging experiments are carried out, and the individual pushing and dragging experiment data is combined to generate a whole force-displacement characteristic figure. The experiment results are plotted in Figure 4.10. The yellow

shaded area shows the distribution range (maximum and minimum force at each displacement) of the experiment result. The dashed red line is the theoretical result, and the solid blue line shows the mean value of the 10 combined experiment results. To quantify the experiment results, there are several important parameters, i.e., maximum force F_{max}, minimum force F_{min}, maximum opening displacement d_o (displacement between the first two zero forces), and maximum closing displacement d_c (displacement between the last two zero forces). We show the mean of these 4 parameters in 10 experiments together with the theoretical value from the simulation in Table 4.2.

From Figure 4.10 and Table 4.2, the experimental results are reasonably accurate. The error mainly comes from our simplified models. First, we model the beams as linear springs and the tubes as torsional springs, but they may not exactly follow the spring laws. Second, the fabrication and assembly process may also introduce some errors for the exact dimensions for each of the components. From Figure 4.10, the error increases when the opening force is decreasing for the opening experiment. The largest error (10.94%) occurs with the maximum opening displacement. The reason is that the tubes are compressed since they are parallel to the base. This period corresponds to the lagging part of the experiment results. The compression will result in a smaller l_t, which will increase the bending stiffness k_θ based on PRBM. As analyzed in section 4.3, larger k_θ will increase BI and make the system less bistable. This will make the force-displacement characteristic decrease more slowly. To better illustrate this phenomenon, we simulate several cases with six different tube lengths ($5\,\mathrm{mm}$ to $7\,\mathrm{mm}$ with a step size of $0.4\,\mathrm{mm}$) and plot the force-displacement characteristics to compare the difference (Figure 4.11). The simulation results show that the force-displacement characteristics are almost the same before $9\,\mathrm{mm}$ displacement. After $9\,\mathrm{mm}$, grippers with longer tubes tend to have a larger force to make the system more bistable. As a result, the force profiles for grippers with shorter tubes decrease slower than the longer ones, which explains the lagging of the experiment results.

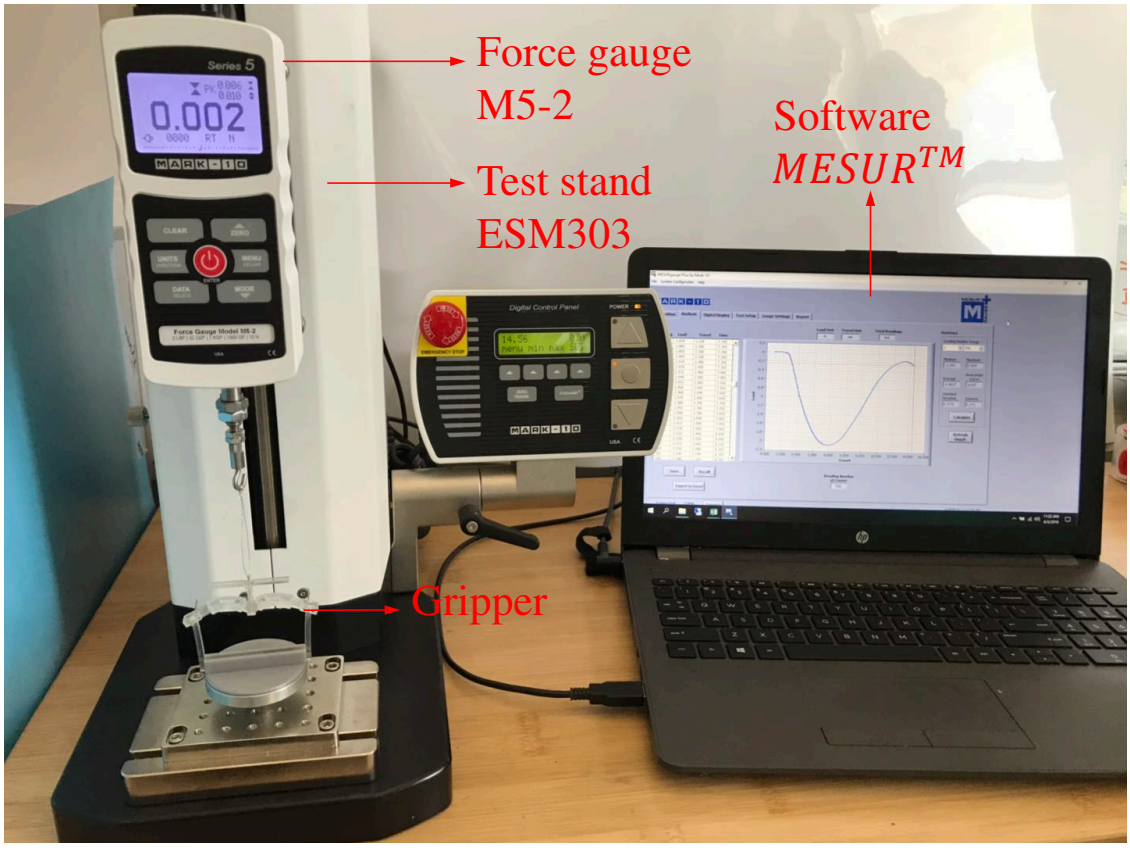


Figure 4.9: Force-displacement characteristic experiment setup. The gripper is attached to the test stand and a hook is attached to the force gauge. In dragging experiment, the hook will pull the switching pad with a string. In pushing experiment, the hook will directly push the switching pad. Meanwhile, the software will record the corresponding displacement and tension/compression force.

Table 4.2: Experiment data and simulation data comparison

	F_{max}(N)	F_{min}(N)	d_o(mm)	d_c(mm)
Simulation	2.15	-0.41	12.62	4.89
Experiment	2.12	-0.39	14.17	5.23
Error(%)	1.4	5.1	10.94	6.5

4.4.3 Friction test experiment

As the gripper can generate different normal forces F_n and friction forces f on different sized objects, we experimentally test the prediction for successful perching on objects with different sizes in this subsection.

We use 3D printed objects made from Polylactic Acid (PLA) with different sizes as the perching object. First, we experimentally test the friction coefficient μ between Ecoflex 30 and 3D

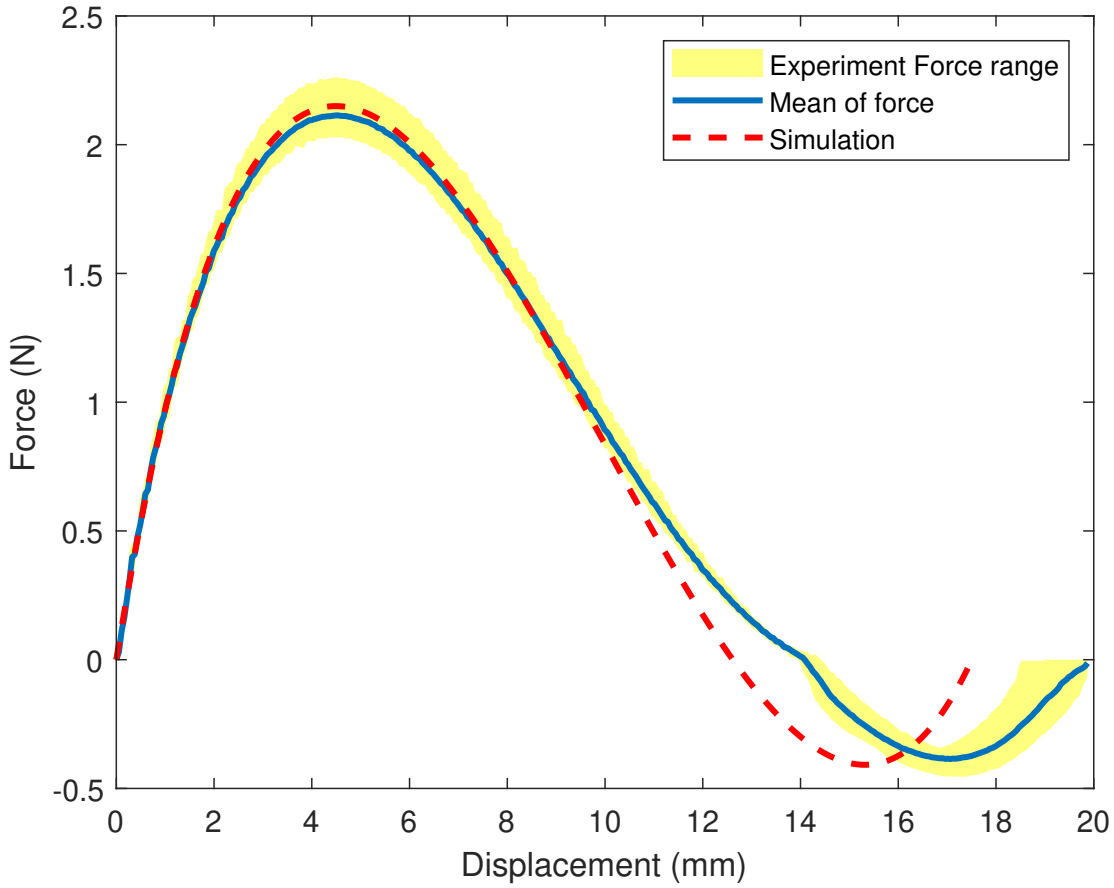


Figure 4.10: Force-displacement characteristic experiment results. 10 pushing and 10 dragging experiments are carried out. The yellow shaded area shows the experiment force range. The blue line shows the mean of forces from 10 experiments. The dashed red line shows the simulation result from mathematical models.

printed PLA. Since Ecoflex 30 is soft, the friction coefficient between it and PLA is not constant. We experimentally test and find that the friction coefficient varies with normal force. In this experiment, we designed a container with a mass of $3.08\,\mathrm{g}$. Then we added different weights from $0\,\mathrm{g}$ to $65\,\mathrm{g}$ with a step size of $5\,\mathrm{g}$, and use the test stand to horizontally drag the ecoflex 30 on 3D printed PLA surface. The maximum friction force before relative motion occurs is recorded to calculate μ. After 6 consistent tests, we find that a minimum of 5th order polynomial can fit the result well:

$$\mu = -52{F_n}^5 + 107{F_n}^4 - 80{F_n}^3 + 27{F_n}^2 - 4F_n + 1 \tag{4.10}$$

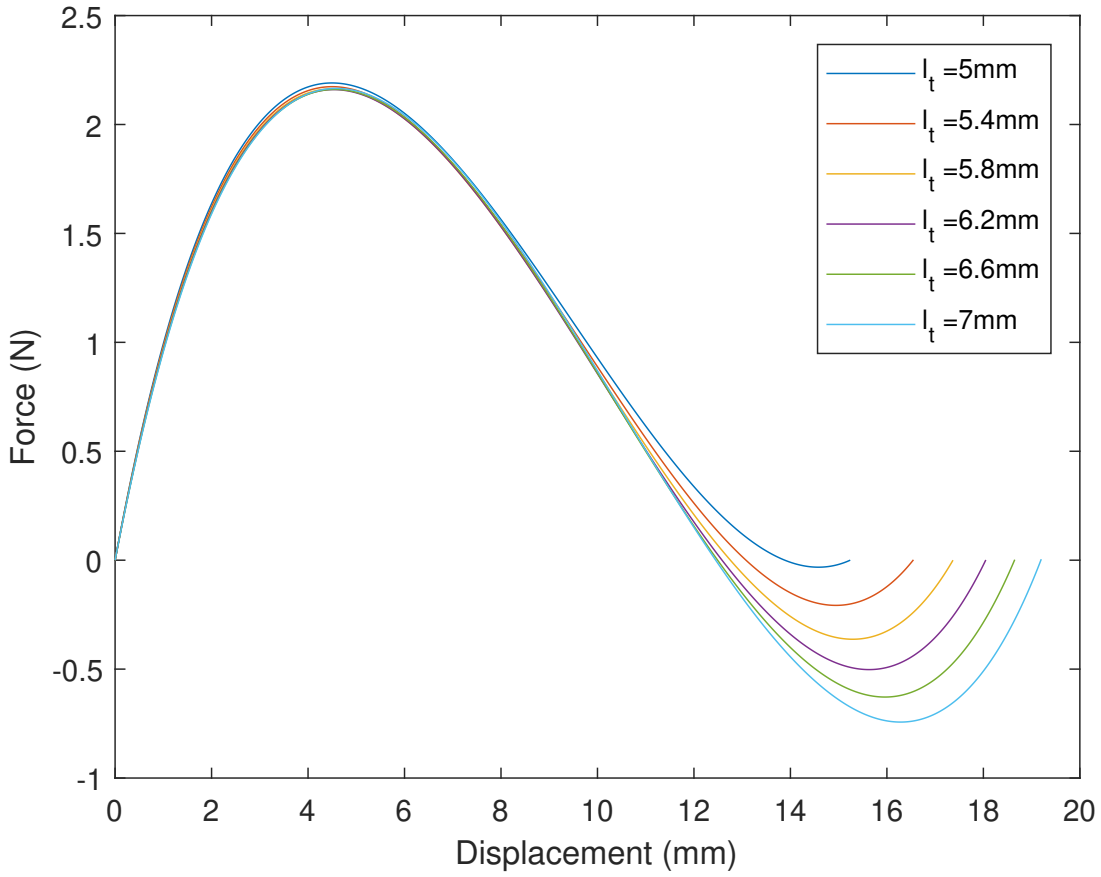


Figure 4.11: Simulations for the influence of different tube length to the force-displacement characteristic. Six different tube lengths from 5 mm to 7 mm are used. For different tube lengths, the force-displacement characteristics are almost the same before about 9 mm displacement. After 9 mm, the gripper with longer tubes tends to have a larger recover force to make the system more bistable.

With Eqn. (4.8) and Eqn. (4.10), we can calculate the range for the size of the PLA objects that will allow for successful perching. The perching object should have a width from $3.7\,\mathrm{mm}$ to $36.4\,\mathrm{mm}$.

To verify the prediction, we printed several PLA cubes with eight different sizes for boundary cases: 3, 4, 5, 6, 33, 34, 35, and 36 mm. We manually make the robot clip on the cubes and see whether it can stay or not. The results show that the robot can perch on such cubes with sizes of 5, 6, and 33 mm, which is a bit smaller than the estimation range. This error might be caused by the friction coefficient estimation.

4.4.4 Perching and grasping experiment

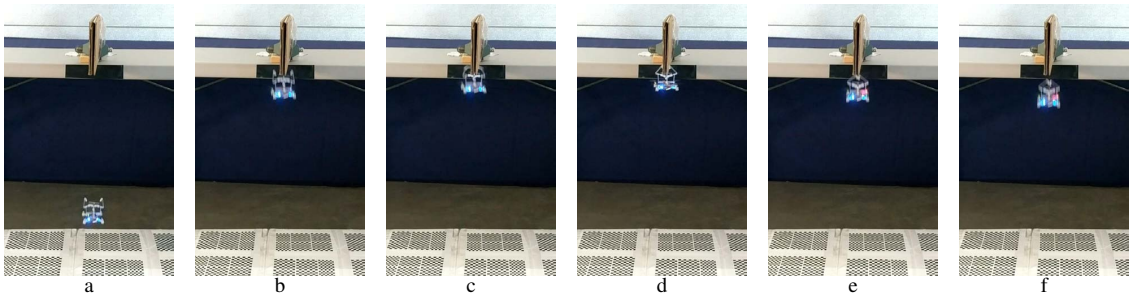
After verifying our models, we conduct various perching experiments for Perchflie in controlled and uncontrolled environments as well as the grasping experiment. For encircling perching, as long as the dimension of the perching object is smaller than the space formed by the fingers and switching pad, the Perchflie can successfully perch on it. For tall objects, the robot can use the clipping method to clip on the objects to hold the Perchflie with enough friction forces.

The perching experiments are conducted on two different objects with the two perching methods: encircling and clipping, with two typical experiments shown as image sequences in Figure 4.12. The clipping perching is conducted on a vertically placed cardboard with a width of $7\,\mathrm{mm}$. The encircling perching is conducted on a cuboid wood with a width of $31\,\mathrm{mm}$ and a height of $5\,\mathrm{mm}$. In each experiment, the Perchflie is manually controlled to take off and accelerate to the perching object. With an impact force acting on the switching pad, the gripper will close to perch with either clipping or encircling method. After perching, the motor is controlled to open the gripper. After detachment, Perchflie hovers immediately. The motor continues rotation to fully open the gripper while hovering. After the gripper is fully opened, the motor will rotate in the opposite direction to leave the lever away from the switching pad for the next perching. At last, the motor stops and Perchflie can perch again. Figure 4.12 illustrates the perching sequence for one cycle of perching and releasing for the two different objects.

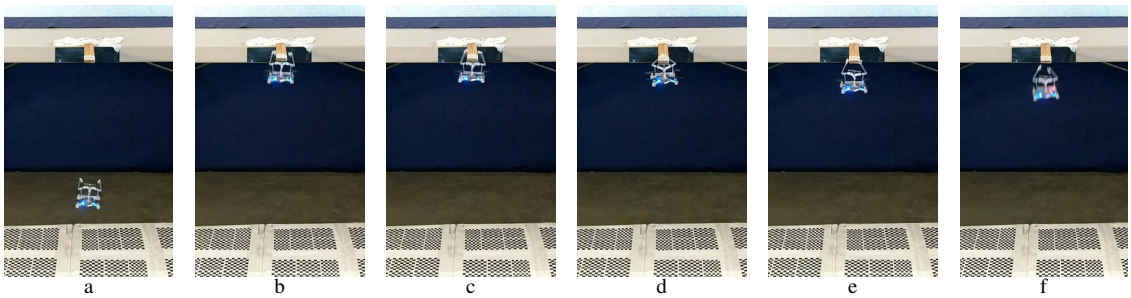
To investigate the potential of this bistable gripper, we conducted one more grasping experiment. Limited by the payload of the Crazyflie, the object is a $116\times77\times14\ \mathrm{mm}$ foam. The foam is manually put on the gripper when the Perchflie is airborne. After the Perchflie arrives at the destination. The gripper is controlled to release the object. The process is shown in Figure 4.13.

4.5 Chapter Summary

In this chapter, we design, analyze, and develop a bistable compliant gripper to enable passive perching for MAVs. The bistability makes the gripper passively trigger the perching process by the impact force. With a lever mechanism driven by a DC motor, it can repeat the perch and take-



(a) Clipping perch



(b) Encircling perch

Figure 4.12: Two perching experiments on different objects with two perching methods. The first image sequence shows the clipping method on cardboard. The second image sequence shows the encircling method. In each experiment, the Perchflie undergoes a) taking off, b-c) perching, d) staying on the object, e-f) releasing. A detailed view of the perching state for both clipping and encircling is shown in Figure 4.1 (b) and (c).

off cycle. It also provides two different perching methods, clipping and encircling, to expand the objects that it can perch. We investigate the cause of the bistability to provide a design guideline by defining a bistability index. We establish a model to predict the force-displacement characteristics for the gripper and experimentally verify the proposed model. Various perching experiments are conducted to show the feasibility of this gripper for MAV perching.

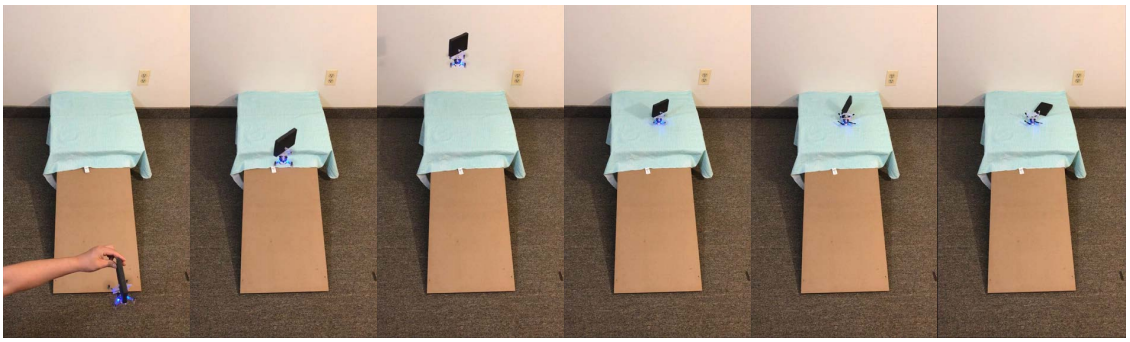


Figure 4.13: Image sequence for aerial grasping. A foam is manually put on the gripper when the robot is airborne. After the Perchflie arrives at the destination, it lands and opens the gripper to release the foam.

Chapter 5

Integration of Computational and Mechanical Intelligence

In previous chapters, computational intelligence (both state estimation and trajectory planning) and mechanical intelligence are implemented individually. To combine all these techniques on one robot, a customized MAV which contains an onboard computer, a camera, and a larger gripper is developed. In this chapter, the MAV is controlled to autonomously perch on a horizontal rod with both CTDTS and IPTS. The TTC is estimated from position and velocity feedback from the motion tracking system. Both experiments show the customized MAV system is able to combine computational and mechanical intelligence to realize robust autonomous perching.

5.1 MAV platform

Although the Crazyflie used in chapters 3 and 4 performs well in experiments, due to its limited computational power, it is impossible to implement the estimation and control algorithm onboard, even without considering the extra payload from both the camera and gripper. Thus, a new customized MAV with high computational power and a large payload is needed to combine both computational intelligence and mechanical intelligence to realize the autonomous perching.

The customized MAV platform is shown in Figure 5.1. It consists of a $250\,\mathrm{mm}$ carbon fiber frame (Lumenier QAV250 from flightclub), a Raspberry Pi 3 B+ as the onboard computer, a Raspberry Pi camera as the TTC estimation sensor (for future work), an autopilot (Mindracer from Mindpx) which controls the motors speed through a 4 in 1 ESC (Spedix from Amazon), a 3D printed larger bistable gripper, a releasing motor (from Pololu), a motor driver (DRV8838 from Pololu), 5 reflective markers for the motion tracking system, and other necessary parts for a quadcopter, i.e., brushless motors, propellers, etc. The whole MAV system is about $498\,\mathrm{g}$ and can hover in the air for about 4 minutes.

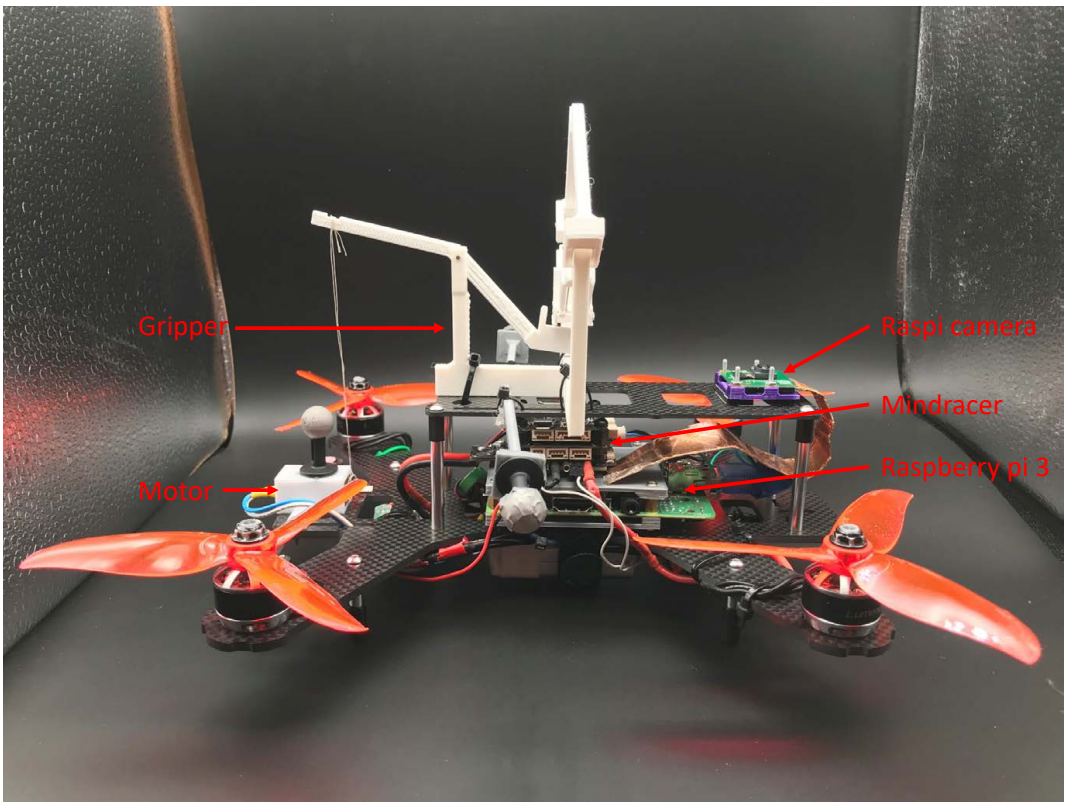


Figure 5.1: The customized MAV platform. The Raspberry Pi is used as the onboard computer, receiving the local position data from the motion tracking system, processing images, and sending attitudes, and thrust setpoints. A larger bistable gripper is installed on top of the drone, and a motor is used to release the gripper.

In this chapter, section 5.2 details the new gripper design and the design necessity. Section 5.3 introduces the trajectory planning in this perching task and the perching experiment results using CTDTS and IPTS. Section 5.4 concludes this chapter.

5.2 New gripper design

Since the new MAV platform is much heavier than the Crazyflie, a larger bistable gripper (The name larger gripper will be used to refer to the new one in this chapter.) which can hold a larger weight is designed as shown in Figure 5.2.

Similar to the bistable gripper in Chapter 4, the larger gripper consists of 4 main parts: the beam and base, the lever (the motor is not designed to be installed on the gripper.), the switching pad, and the fingers (upper finger and lower finger). There are two main differences between the gripper in Chapter 4 and the larger gripper. First is that the connection parts between the switching pad and

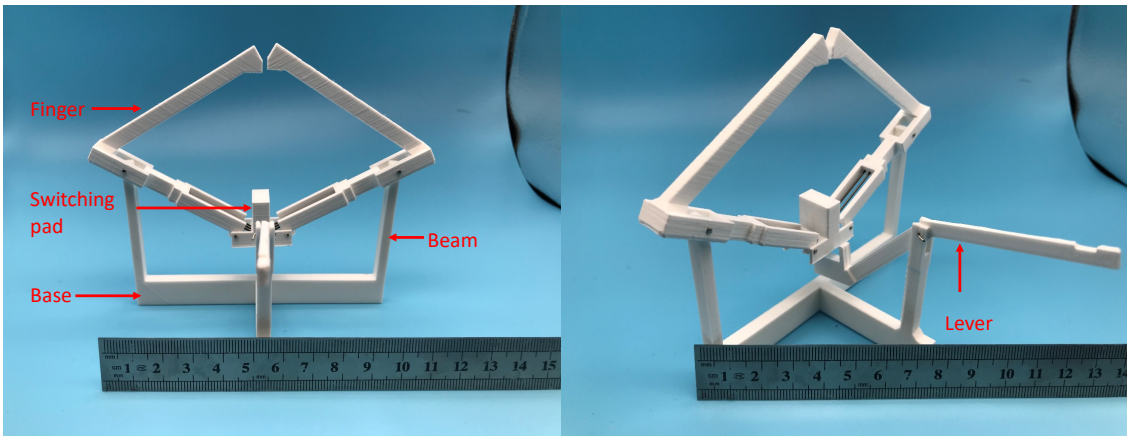


Figure 5.2: The larger gripper. It consists of 4 main parts: the beam and base, the lever (the motor is not designed to be installed on the gripper.), the switching pad, and the fingers (upper finger and lower finger). It is designed only for encircling perching.

the fingers are replaced with shape memory alloy (SMA). For the bistable gripper in Chapter 4, a silicone tube is used as the compliant connection. However, during preliminary designs, two drawbacks of the silicone tubes presented: First, the tube tends to bend out of the cross-section plane of the tube, which makes the gripper asymmetric in the open state. Second, as talked about in Chapter 4, the torsional stiffness of the tube influence the bistability. Compared to the stiffness of the beams, it is difficult to find a strong tube which can provide large enough torsional resistance, which makes $BI = -\frac{F_{min}}{F_{max}} \approx 1$. when $F_{min} \approx F_{max}$, it means we can not achieve the easy to close, but stable to hold requirement for the gripper. However, the silicone tube is only served as a compliant connection. Thus, any compliant material can be potentially used to replace the tube. Finally, the shape memory alloy (SMA) turns out to be a good substitute as shown in Figure 5.3. And three parallel SMAs work well for generating the appropriate opening and closing forces. The SMA is designed to be inserted into the holes on the switching pad and the lower finger. To avoid the possible bending-out direction movement, the lower finger is extended as a shell for the SMA and connected to the switching pad with a shaft. The second difference is that the contact feet are removed since only encircling perching is considered. The tip of the top finger is designed to be flat in the horizontal direction to make it hold on to the flat surfaces more firmly.

Table 5.1: Design parameters of the larger gripper

$\alpha(^\circ)$	$\theta_0(^\circ)$	$h_b(\mathrm{mm})$	$l_b(\mathrm{mm})$	$l_f(\mathrm{mm})$	$l_s(\mathrm{mm})$	$l_h(\mathrm{mm})$
60	30	12	50	25	29.5	5.5

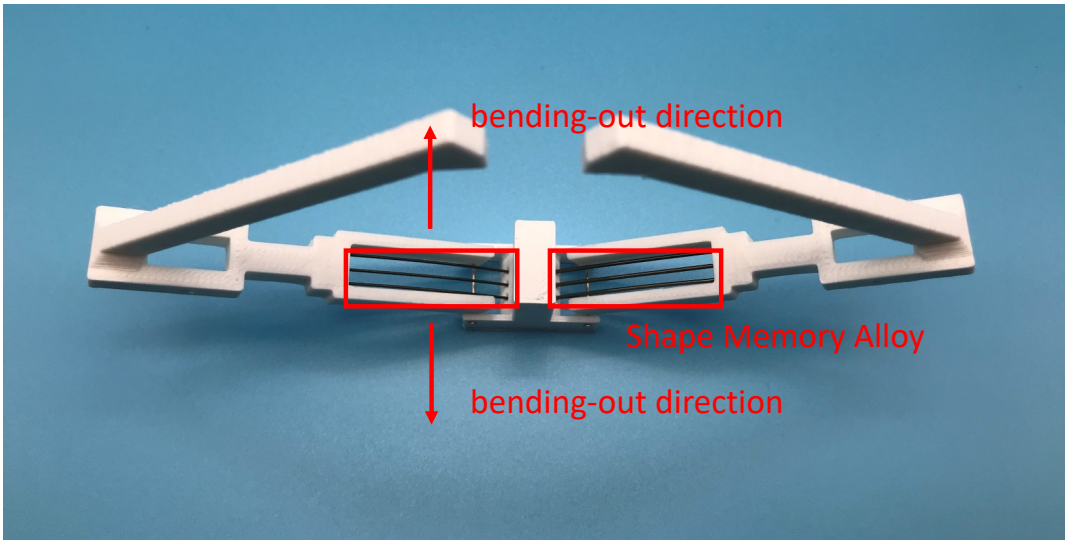


Figure 5.3: Three parallel SMAs are used as the compliant part to connect the switching pad and the fingers. To avoid the possible bending-out movement shown above, the lower finger is extended to be a shell for the SMA and connected to the switching pad with a shaft.

All the parts of the larger gripper are 3D printed with PLA material using a Prusa printer except the shafts and SMAs. The total weight is $24.47\,\mathrm{g}$ (without the motor) while the gripper in Chapter 4 is $8\,\mathrm{g}$ (with the motor). The design parameters are shown in Table 5.1. Since the upper finger dimension is only related to the clipping perching, some parameters are not listed. l_f for the larger gripper is the distance from the pivot of the finger to the surface where the SMA hole is designed. And l_s is the SMA length which corresponds to the tube length l_t in Chapter 4. And other parameters are referred to the same part as described in Section 4.3.3.

5.3 Trajectory planning and experiment

To validate the combination of computational intelligence and mechanical intelligence, the new MAV is controlled to autonomously perch on a horizontal rod with CTDTS or IPTS based on both motion tracking system and feedback. In this section, both CTDTS and IPTS trajectories are designed. And both the simulation and experiments are carried out.

Table 5.2: Planning and experiment initial conditions comparison

	X_0	$\dot{X}_0$	V_l	V_u	V_{max}	a_{max}	a_{min}	τ_s
Old simulation	-3	1.5	0.7	1	2.5	1.4	-1.4	-0.5
New simulation	-1.65	0.8	0.1	0.3	1	5	-9.8	-0.5

5.3.1 Trajectory planning

Before conducting the perching experiment, we first simulate the CTDTS and IPTS to see whether the simulation can achieve good results. As discussed in Chapter 4, the gripper will be installed on top of the MAV. We changed the TTC control direction from flying forward to flying upward. However, the experimental space is limited by the height of the roof. We have to change the initial conditions under the height limitation. The initial conditions for IPTS in this chapter (referred to as new simulation) are listed in Table 5.2 together with the ones in Chapter 3 (referred to as old simulation) for comparison. In this chapter, the initial conditions of CTDTS and IPTS are the same. The only difference in the simulation constraints is that for CTDTS, $V_l = V_u = 0.3\,\mathrm{m/s}$ while IPTS has different V_l and V_u. Note that the maximum velocity of the new MAV can be larger than $2\,\mathrm{m/s}$, but it is set to $1\,\mathrm{m/s}$ for safety reasons.

With the new initial conditions and constraints, the switching time for CTDTS is $t_s = 2.64\,\mathrm{s}$, and the reference is as Equation 5.1. Similar to Chapter 3, a third order IPTS is used, the switch time $t_s = 1.91\,\mathrm{s}$, and the reference is as Equation 5.2.

$$\tau_{ref_{CTDTS}}(t) = \begin{cases} 0.591t - 2.06, & \text{if } t < 2.64 \\ t - 3.143, & \text{if } t \geq 2.64 \end{cases} \tag{5.1}$$

$$\tau_{ref_{IPTS}} = \begin{cases} \dfrac{1}{0.3106t^3 - 0.8824t^2 - 0.24t - 0.4848}, & \text{if } t < 1.92 \\ t - 2.42, & \text{if } t \geq 1.92 \end{cases} \tag{5.2}$$

The planned trajectory is shown in Figure 5.4 for both CTDTS and IPTS. The CTDTS takes about $3.14\,\mathrm{s}$ while the IPTS takes about $2.42\,\mathrm{s}$ to finish the perching. Since the CTDS has been

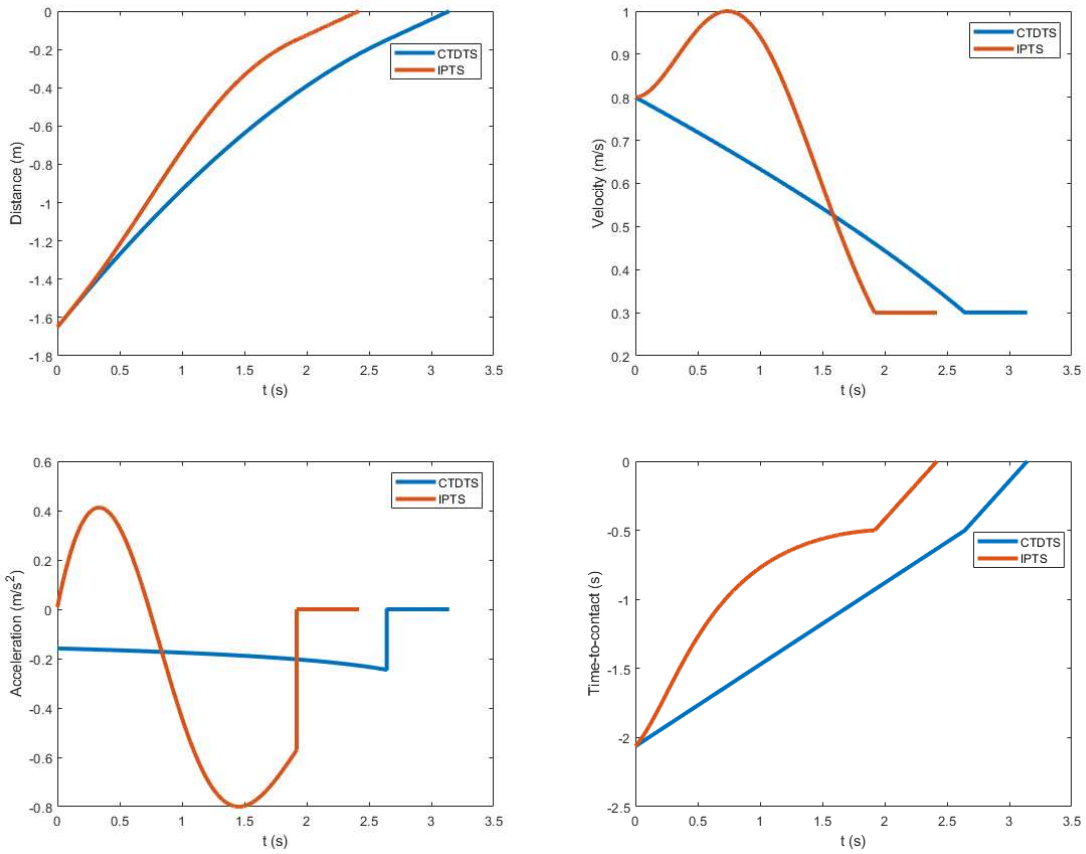


Figure 5.4: Planned trajectories for the new initial conditions and constraints. The blue line shows the result of the CTDTS and the orange line shows the IPTS. From top to bottom, left to right, are the distance, velocity, acceleration, and TTC trajectories.

shown not appropriate for perching, it is not simulated here. Similar to the previous simulation, the IPTS will increase velocity first and decrease the velocity later, which is faster than the CTDTS.

5.3.2 Perching Experiment

In this section, the new CTDTS and IPTS trajectories are used to control the MAV to perch onto a horizontal rod based on the TTC estimated from position and velocity feedback from the motion tracking system. The experiment setup is introduced first. Then the perching results are presented.

Experiment setup

The perching scenario is simplified to control the MAV to autonomously perch on a horizontal rod with encircling perching, as shown in Figure 5.5. In this scenario, the main direction we want to apply TTC control is the Z direction. The X and Y direction positions are controlled with position feedback from the motion tracking system. The larger bistable gripper is installed on top of the MAV. To make the gripper perfectly grasp the rod, the Raspberry Pi camera is also used to detect the lines and estimate the orientation of the rod. Finally, the Raspberry Pi will send roll, pitch, yaw, and thrust commands to control the motion of the MAV.

The control flow chart is shown in Figure 5.6. The Raspberry Pi 3 B+ is used as the onboard computer. It receives the drone position from the motion tracking system through wifi. Then it sends the control commands, local position information, etc, to the autopilot (mindracer). On the other hand, the autopilot will send the sensor output and firmware estimator (extended Kalman Filter) data back to the Raspberry Pi. With the local position information from Raspberry Pi, the autopilot is able to switch to the offboard board, in which it can directly receive the control command from the Raspberry Pi. Finally, the autopilot will send speed commands to the ESCs based on the control command received. When releasing is needed, the Raspberry Pi will send the PWM signal to the releasing motor and control the drone to safely land on the ground. Raspberry Pi is also used to estimate the rod orientation and the TTC through the camera.

Perching with motion tracking system feedback

As a preliminary step, the TTC is estimated from the position and velocity feedback from the motion tracking system. In both CTDTS and IPTS experiment, the MAV is first controlled to hover under the rod. Then the camera will continuously capture the images of the rod on top of it. The Raspberry Pi will estimate the rod orientation by detecting the most confident lines with Hough-LinesP function in OpenCV with Kalman Filter. The estimated rod orientation is transformed to goal yaw angle based on the geometric relationship. And finally, the goal yaw angle is sent to the autopilot to make the gripper align with the rod using the controller implemented in the firmware. Next, the MAV will be controlled to accelerate. When both the distance and velocity are in a

reasonable range with respect to the initial conditions in Section 5.3.1, the TTC control (CTDTS or IPTS) is initiated. After switching time t_s, the MAV will be controlled to fly upward with a constant speed based on TTC feedback until perching is realized. When needed, the Raspberry Pi will control the releasing motor to open the gripper and safely land the MAV.

For X and Y direction, PD and PID controllers using position feedback are designed, the output is the desired pitch and roll angle respectively. The corresponding PD parameters for X direction are: $k_{px} = 2.2$, $k_{dx} = 6$ and PID parameters for Y direction are: $k_{py} = 4$, $k_{dy} = 10$, $k_{iy} = 0.4$. The output of the two controllers are the desired pitch and roll angle, respectively. The Z direction is controlled with a tau controller as shown in Eqn 3.17 but with only a P term. For CTDTS, $k_{pz} = 10$ while $k_{pz} = 11.5$ in IPTS.

The perching experiment results are shown in Figure 5.7. The left column shows the results for the CTDTS while the right column shows the IPTS. From top to bottom are distance, velocity, and TTC. Five experiments are carried out for each strategy. The first stage of both strategies starts at $t = 0\,\mathrm{s}$. For CTDTS, the average switch time (thick vertical blue line) for 5 experiments is about $2.78\,\mathrm{s}$ compared to $2.64\,\mathrm{s}$ as planned. For IPTS, the average switch time for 5 experiments is $1.74\,\mathrm{s}$ compared to $1.91\,\mathrm{s}$ as planned. As planned, the CTDTS will decrease velocity until it reaches the switch time. The IPTS will first increase velocity then decrease the velocity until the switch time. After switch time, the MAV tries to fly upward with a constant velocity. The dashed green lines in the TTC figures are the reference TTC trajectory for each strategy. Overall, the perching performance is acceptable. However, the second stage, which controls the velocity to be constant based on TTC feedback, is not performing well. More research needs to be done to solve this problem.

5.4 Chapter Summary

In this chapter, a customized MAV platform is used to combine computational intelligence and mechanical intelligence. To make the new MAV perch on a horizontal rod, a larger bistable gripper using SMA is designed. Both CTDTS and IPTS using motion tracking system based TTC

estimation perching are experimentally carried out. The experiment shows that the computational intelligence and mechanical intelligence propose in this dissertation can be used for aerial robot perching.

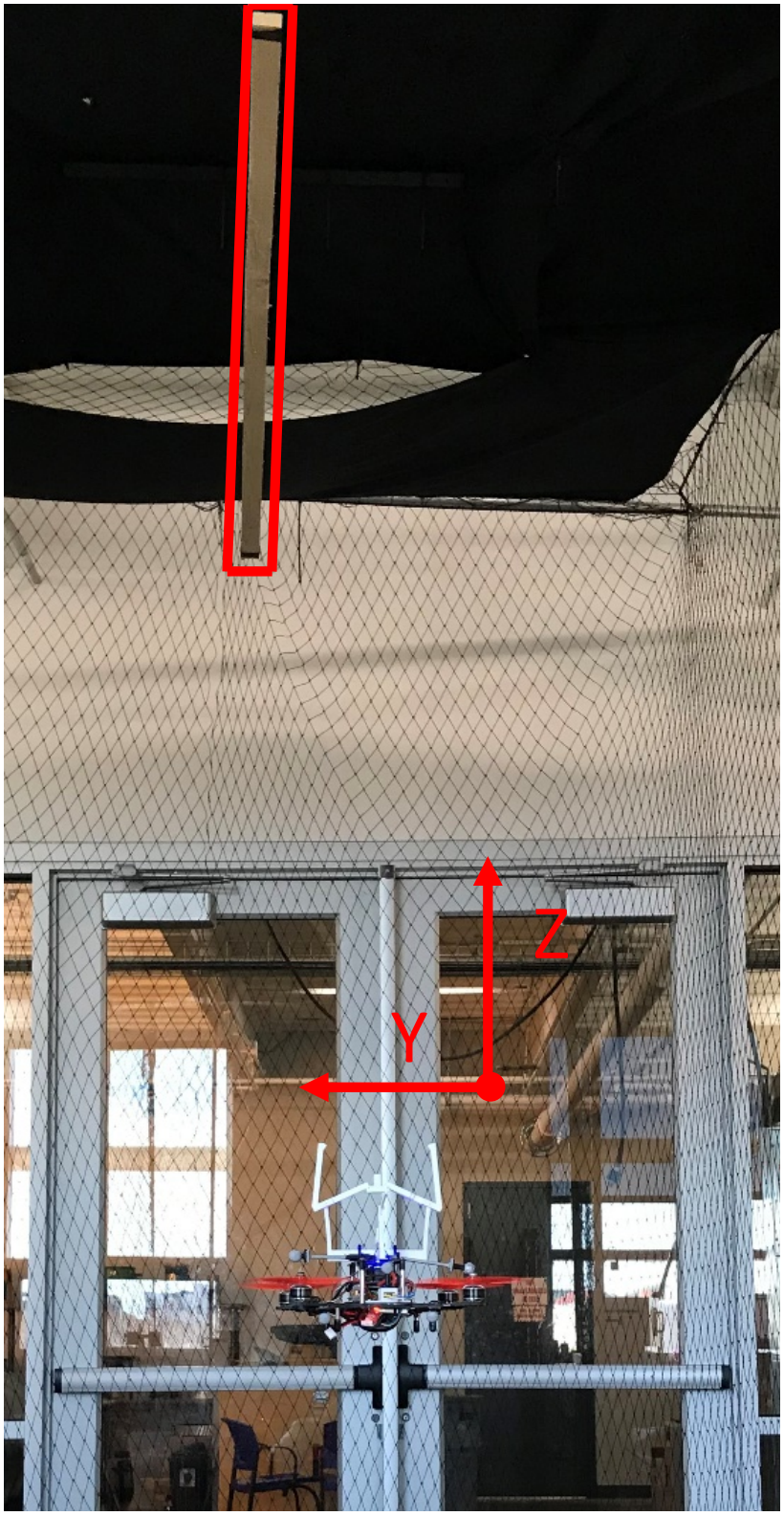


Figure 5.5: The perching scenario of the experiment. The MAV is controlled to perch on the horizontal rod. The X and Y directions are controlled with position feedback while the Z direction of the MAV is controlled with TTC feedback. Both CTDTS and IPTS are implemented on the perching experiments.

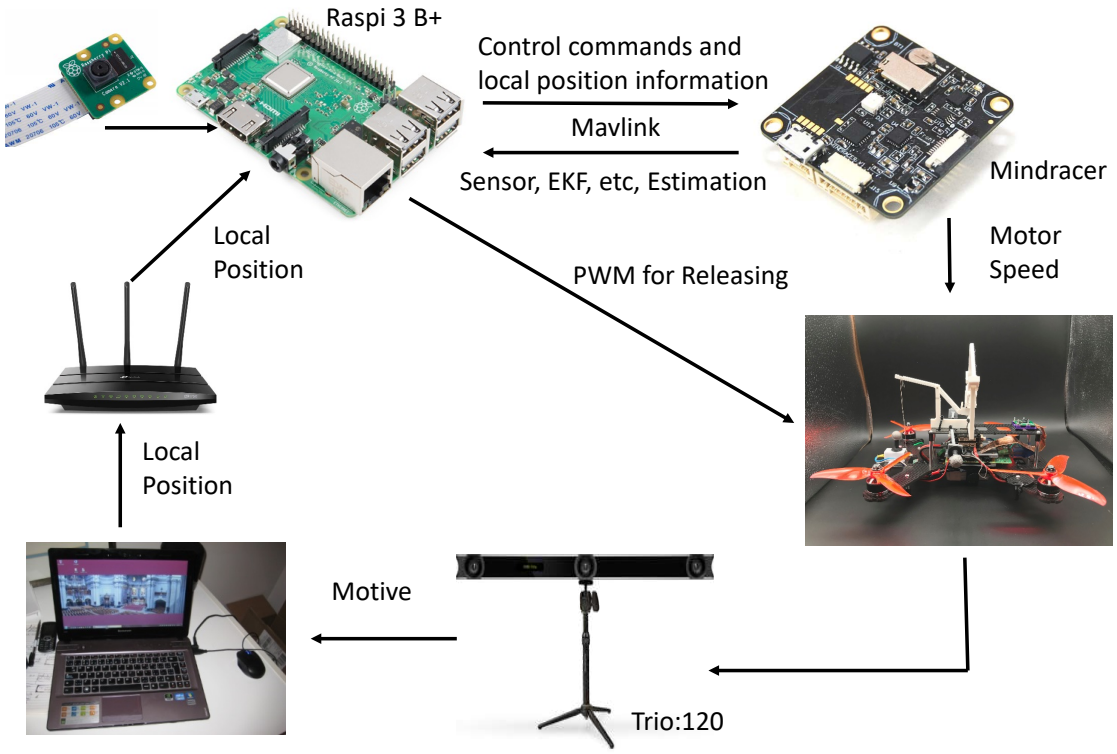


Figure 5.6: The control flow chart of the experiment. The Raspberry Pi 3B+ is used as the onboard computer. It sends the desired setpoints and local position information to the autopilot. With the local position information received, the autopilot can directly receive the Raspberry Pi control command and control the MAV motion.

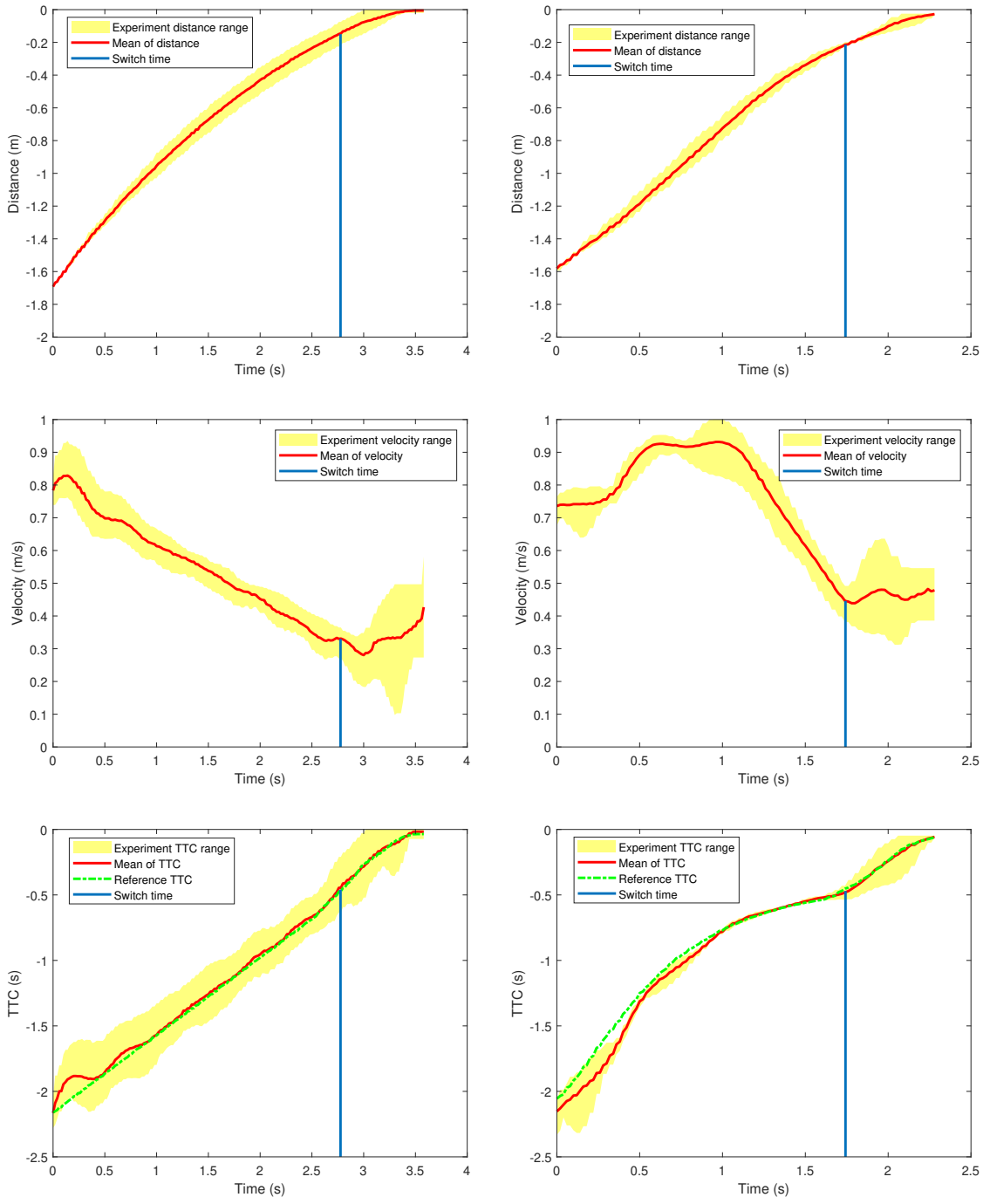


Figure 5.7: Experiment results for CTDTS (left column) and IPTS (right column). From the top to the bottom are the distance, velocity, and tau for CTDTS and IPTS, respectively. For each strategy, the yellow area represents the range of 5 experiments at a specific time. The second stage starts at the blue vertical line (2.78 s for CTDTS and 1.74 s for IPTS). The reference TTC is plotted in dashed green lines in the TTC figures. The CTDTS and IPTS can realize the nonzero contact velocity, and IPTS requires a shorter time for perching.

Chapter 6

Conclusions and future work

6.1 Conclusions

In this dissertation, a customized MAV platform is designed to combine computational intelligence and mechanical intelligence for aerial robot perching. The goal of this research is to enable MAVs with perching capability to potentially extend their flight time. However, there are two common challenges: computational challenge and mechanical challenge.

The computational challenge in perching tasks consists of flight state estimation, perching trajectory planning, and control. For the state estimation challenge, a visual information, time-to-contact (TTC), which is inspired by the landing process of the birds and flies, is used as the feedback. To estimate the TTC, a featureless method is used and extended by considering angular velocities. And preliminary experiments using the new estimation algorithm are carried out to control the mobile robot to autonomously brake before colliding with the objects in front. However, the widely used TTC trajectory will generate zero or very small contact velocity. This is not ideal when the perching mechanism needs a triggering force. To solve this challenge, two reference trajectories, CTDTS and IPTS, are proposed to realize non-zero contact velocity based on TTC feedback. Following CTDTS, the MAV can decelerate to the desired contact velocity and maintain the velocity until perching finishes. Alternatively, the MAV can accelerate first, then decelerate to the desired contact velocity and maintain the velocity until perching following the IPTS. Both the preliminary experiments using Crazyflie and the final perching experiment with the customized MAV show the feasibility of CTDTS and IPTS for non-zero contact velocity perching. And IPTS is shown to be faster while can satisfy more perching constraints.

On the other hand, the mechanical challenge requires the gripper to be swiftly closed at contact and stable enough to hold the weight of the MAV. To solve this, a bistable gripper which has two stable states is designed. The two stable states of the bistable gripper can switch back and

forth with large enough triggering forces. With the stored strain energy in the bent beams and bent compliant parts (tubes or SMAs), the state transition is fast and does not require additional force once it passes the critical point. In addition, a design guideline is provided to help design an appropriate gripper to achieve desired closing and opening forces. The perching experiments using both Crazyflie and the customized MAV show the capability of the gripper in perching tasks.

In the end, a customized MAV is used to combine both computational intelligence and mechanical intelligence to realize the autonomous aerial robot perching. A Raspberry Pi 3B+ is used as the onboard computer to do the state estimation and control. The successful perching experiments show the feasibility of the proposed computational intelligence and mechanical intelligence in the autonomous perching application.

6.2 Future work

Although this dissertation shows that the computational intelligence and mechanical intelligence can be combined together to realize robust MAV perching, there are still several directions worth to be investigated.

First, in the state estimation part, for birds and flies, TTC to a specific object in their eyes can be estimated fast and accurately. However, the TTC estimation method used in this dissertation assumes the perching object will occupy the whole image. This assumption does not always hold especially when the perching object is only a rod or a power line. One possible solution is to use the size-based estimation method. However, it requires more computational power to do feature extraction and matching. To solve this problem, a Raspberry Pi 4B+ with a better cooling system can be potentially used.

As the second part of the computational intelligence, the trajectory planning based on TTC only considers non-zero initial velocity cases. However, a more natural perching scenario would be the MAV first locates the perching object while hovering, then initiates the perching process. In this case, the initial velocity $V_0 = 0$, and $\tau_0 = \infty$. To consider this more generally, a trajectory should be planned with $V_0 = 0$ based on divergence, the reciprocal of TTC. In addition, the trajectory

planning requires a good estimation of both X_0 and V_0, which is not considered in this dissertation. Implementation of initial condition estimation as talked in [63] could be adopted.

Finally, for the bistable gripper, there are two potential improvements can be done. First, the current releasing mechanism (the motor-driven lever system) occupies a large 3D space, which can be improved using some in-plane mechanisms (e.g., a Sarrus linkage). Second, a more accurate method, beam constraint model (BCM) can be used to derive the force-displacement characteristics of the gripper, since BCM considers the varying loading conditions, boundary conditions, and initial beam curvature compared to the Pseudo-Rigid-Body-Model.

With the above potential improvements, a more intelligent MAV platform can be used to combine the computational intelligence and mechanical intelligence for autonomous perching.

Bibliography

[1] Hendrik Tennekes. *The simple science of flight: from insects to jumbo jets*. MIT press, 2009.

[2] Gordon J Leishman. *Principles of helicopter aerodynamics with CD extra*. Cambridge university press, 2006.

[3] Eastman N Jacobs and Albert Sherman. Airfoil section characteristics as affected by variations of the reynolds number. *NACA report*, 586:227–264, 1937.

[4] James I Hileman, Russell W Stratton, and Pearl E Donohoo. Energy content and alternative jet fuel viability. *Journal of propulsion and Power*, 26(6):1184–1196, 2010.

[5] William RT Roderick, Mark R Cutkosky, and David Lentink. Touchdown to take-off: at the interface of flight and surface locomotion. *Interface Focus*, 7(1):20160094, 2017.

[6] Courtney E Doyle, Justin J Bird, Taylor A Isom, Jason C Kallman, Daman F Bareiss, David J Dunlop, Raymond J King, Jake J Abbott, and Mark A Minor. An avian-inspired passive mechanism for quadrotor perching. *IEEE/ASME Transactions on Mechatronics*, 18(2):506–517, 2013.

[7] Ludovic Daler, Adam Klaptocz, Adrien Briod, Metin Sitti, and Dario Floreano. A perching mechanism for flying robots using a fibre-based adhesive. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4433–4438. IEEE, 2013.

[8] Morgan T Pope, Christopher W Kimes, Hao Jiang, Elliot W Hawkes, Matt A Estrada, Capella F Kerst, William RT Roderick, Amy K Han, David L Christensen, and Mark R Cutkosky. A multimodal robot for perching and climbing on vertical outdoor surfaces. *IEEE Transactions on Robotics*, 33(1):38–48, 2017.

[9] MA Graule, P Chirarattananon, SB Fuller, NT Jafferis, KY Ma, M Spenko, R Kornbluh, and RJ Wood. Perching and takeoff of a robotic insect on overhangs using switchable electrostatic adhesion. *Science*, 352(6288):978–982, 2016.

[10] Mirko Kovač, Jürg Germann, Christoph Hürzeler, Roland Y Siegwart, and Dario Floreano. A perching mechanism for micro aerial vehicles. *Journal of Micro-Nano Mechatronics*, 5(3-4):77–91, 2009.

[11] Joseph Moore and Russ Tedrake. Powerline perching with a fixed-wing uav. In *Proceedings of the AIAA Infotech@ Aerospace Conference, Seattle, WA*, 2009.

[12] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory generation and control for precise aggressive maneuvers with quadrotors. *The International Journal of Robotics Research*, 31(5):664–674, 2012.

[13] Justin Thomas, Morgan Pope, Giuseppe Loianno, Elliot W Hawkes, Matthew A Estrada, Hao Jiang, Mark R Cutkosky, and Vijay Kumar. Aggressive flight with quadrotors for perching on inclined surfaces. *Journal of Mechanisms and Robotics*, 8(5):051007, 2016.

[14] Kartik Mohta, Vijay Kumar, and Kostas Daniilidis. Vision-based control of a quadrotor for perching on lines. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3130–3136. IEEE, 2014.

[15] Alexis Lussier Desbiens, Alan T Asbeck, and Mark R Cutkosky. Landing, perching and taking off from vertical surfaces. *The International Journal of Robotics Research*, 30(3):355–370, 2011.

[16] Dino Mehanovic, John Bass, Thomas Courteau, David Rancourt, and Alexis Lussier Desbiens. Autonomous thrust-assisted perching of a fixed-wing uav on vertical surfaces. In *Conference on Biomimetic and Biohybrid Systems*, pages 302–314. Springer, 2017.

[17] Emily Baird, Norbert Boeddeker, Michael R Ibbotson, and Mandyam V Srinivasan. A universal strategy for visually guided landing. *Proceedings of the National Academy of Sciences*, 110(46):18686–18691, 2013.

[18] Vicki Bruce, Patrick R Green, and Mark A Georgeson. *Visual perception: Physiology, psychology, & ecology*. Psychology Press, 2003.

[19] Davis N Lee and Paul E Reddish. Plummeting gannets: a paradigm of ecological optics. *Nature*, 1981.

[20] Jean-Christophe Zufferey and Dario Floreano. Fly-inspired visual steering of an ultralight indoor aircraft. *IEEE Transactions on Robotics*, 22(1):137–146, 2006.

[21] David N Lee, Mark NO Davies, Patrick R Green, et al. Visual control of velocity of approach by pigeons when landing. *Journal of Experimental Biology*, 180(1):85–104, 1993.

[22] Floris Van Breugel and Michael H Dickinson. The visual control of landing and obstacle avoidance in the fruit fly drosophila melanogaster. *The Journal of Experimental Biology*, 215(11):1783–1798, 2012.

[23] David N Lee. A theory of visual control of braking based on information about time-to-collision. *Perception*, 5(4):437–459, 1976.

[24] James J Gibson. Visually controlled locomotion and visual orientation in animals. *British journal of psychology*, 49(3):182–194, 1958.

[25] David N Lee. Guiding movement by coupling taus. *Ecological psychology*, 10(3-4):221–250, 1998.

[26] David N Lee. General tau theory: evolution to date. *Perception*, 38(6):837, 2009.

[27] David N Lee. Tau in action in development. In *Action As An Organizer of Learning and Development: Volume 33 in the Minnesota Symposium on Child Psychology Series*, volume 33, page 1. Psychology Press, 2016.

[28] Kahlouche Souhila and Achour Karim. Optical flow based robot obstacle avoidance. *International Journal of Advanced Robotic Systems*, 4(1):13–16, 2007.

[29] Yukimasa Kaneta, Yuki Hagisaka, and Kazuyuki Ito. Determination of time to contact and application to timing control of mobile robot. In *IEEE Robotics and Biomimetics (ROBIO)*, pages 161–166, 2010.

[30] Chris McCarthy, Nick Barnes, and Robert Mahony. A robust docking strategy for a mobile robot using flow field divergence. *IEEE Transactions on Robotics*, 24(4):832–842, 2008.

[31] Farid Kendoul. Four-dimensional guidance and control of movement using time-to-contact: Application to automated docking and landing of unmanned rotorcraft systems. *The International Journal of Robotics Research*, 33(2):237–267, 2014.

[32] Dario Izzo and Guido De Croon. Landing with time-to-contact and ventral optic flow estimates. *Journal of Guidance, Control, and Dynamics*, 35(4):1362–1367, 2012.

[33] GCHE De Croon, H Ho, C De Wagter, E Van Kampen, B Remes, and Q Chu. Optic-flow based slope estimation for autonomous landing. *International Journal of Micro Air Vehicles*, 5(4):287–298, 2013.

[34] GCHE de Croon, D Izzo, and G Schiavone. Time-to-contact estimation in landing scenarios using feature scales. *Acta Futura*, 5:73–82, 2012.

[35] Herbert Heuer. Estimates of time to contact based on changing size and changing target vergence. *Perception*, 22(5):549–563, 1993.

[36] Dennis Müller, Josef Pauli, Christian Nunn, Steffen Görmer, and Stefan Müller-Schneiders. Time to contact estimation using interest points. In *Intelligent Transportation Systems, 2009. ITSC'09. 12th International IEEE Conference on*, pages 1–6. IEEE, 2009.

[37] Mauro Di Marco, Andrea Garulli, Domenico Prattichizzo, and Antonio Vicino. A set theoretic approach for time-to-contact estimation in dynamic vision. *Automatica*, 39(6):1037–1044, 2003.

[38] Ted Camus. Calculating time-to-contact using real-time quantized optical flow. *National Institute of Standards and Technology NISTIR*, 5609:51, 1995.

[39] Roberto Cipolla and Andrew Blake. Surface orientation and time to contact from image divergence and deformation. In *Computer Vision—ECCV'92*, pages 187–202. Springer, 1992.

[40] Massimo Tistarelli and Giulio Sandini. On the advantages of polar and log-polar mapping for direct estimation of time-to-impact from optical flow. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(4):401–410, 1993.

[41] Yuichi Kawai and Kazuyuki Ito. Estimation method for time to contact from visual information—a simple approach that requires no recognition of objects. In *Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference on*, pages 469–474. IEEE, 2014.

[42] Yukitada Takanashi and Kazuyuki Ito. Estimation of time to contact from blurred images. In *International Conference on Autonomic and Autonomous System (ICAS 2015)*, pages 20–23, 2015.

[43] Berthold KP Horn, Yajun Fang, and Ichiro Masaki. Hierarchical framework for direct gradient-based time-to-contact estimation. In *IEEE Intelligent Vehicles Symposium*, pages 1394–1400, 2009.

[44] Berthold KP Horn, Yajun Fang, and Ichiro Masaki. Time to contact relative to a planar surface. In *IEEE intelligent vehicles symposium*, pages 68–74, 2007.

[45] Jianguo Zhao, Jing Xu, Bingtuan Gao, Ning Xi, Fernando J Cintron, Matt W Mutka, and Li Xiao. Msu jumper: A single-motor-actuated miniature steerable jumping robot. *Robotics, IEEE Transactions on*, 29(3):602–614, 2013.

[46] Jianguo Zhao, Tianyu Zhao, Ning Xi, Matt W Mutka, and Li Xiao. Msu tailbot: Controlling aerial maneuver of a miniature-tailed jumping robot. *Mechatronics, IEEE/ASME Transactions on*, 20(6):2903–2914, 2015.

[47] Berthold K Horn and Brian G Schunck. Determining optical flow. In *1981 Technical symposium east*, pages 319–331. International Society for Optics and Photonics, 1981.

[48] Berthold KP Horn and EJ Weldon Jr. Direct methods for recovering motion. *International Journal of Computer Vision*, 2(1):51–76, 1988.

[49] Haijie Zhang and Jianguo Zhao. Biologically inspired vision based control using featureless time-to-contact estimations. In *Advanced Intelligent Mechatronics (AIM), 2016 IEEE International Conference on*, pages 1133–1138. IEEE, 2016.

[50] Greg Welch and Gary Bishop. An introduction to the kalman filter. department of computer science, university of north carolina, 2006.

[51] Mandyam V Srinivasan, Shao-Wu Zhang, Javaan S Chahl, Erhardt Barth, and Svetha Venkatesh. How honeybees make grazing landings on flat surfaces. *Biological cybernetics*, 83(3):171–183, 2000.

[52] Guido CHE de Croon. Monocular distance estimation with optical flow maneuvers and efference copies: a stability-based strategy. *Bioinspiration & biomimetics*, 11(1):016004, 2016.

[53] Haijie Zhang, Bo Cheng, and Jianguo Zhao. Extended tau theory for robot motion control. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 5321–5326. IEEE, 2017.

[54] Joseph Moore, Rick Cory, and Russ Tedrake. Robust post-stall perching with a simple fixed-wing glider using lqr-trees. *Bioinspiration & biomimetics*, 9(2):025013, 2014.

[55] Haijie Zhang and Jianguo Zhao. Bio-inspired vision based robot control using featureless estimations of time-to-contact. *Bioinspiration & Biomimetics*, 12(2):025001, 2017.

[56] Hao Jiang, Morgan T Pope, Elliot W Hawkes, David L Christensen, Matthew A Estrada, Andrew Parlier, Richie Tran, and Mark R Cutkosky. Modeling the dynamics of perching with opposed-grip mechanisms. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 3102–3108. IEEE, 2014.

[57] David N Lee, Paul E Reddish, and DT Rand. Aerial docking by hummingbirds. *Naturwissenschaften*, 78(11):526–527, 1991.

[58] David N Lee, James A Simmons, Prestor A Saillant, and F Bouffard. Steering by echolocation: a paradigm of ecological acoustics. *Journal of Comparative Physiology A*, 176(3):347–354, 1995.

[59] Hermann Wagner. Flow-field variables trigger landing in flies. *Nature*, 297(5862):147–148, 1982.

[60] MNO Davies and PR Green. Optic flow-field variables trigger landing in hawk but not in pigeons. *Naturwissenschaften*, 77(3):142–144, 1990.

[61] Pakpong Chirarattananon. A direct optic flow-based strategy for inverse flight altitude estimation with monocular vision and imu measurements. *Bioinspiration & biomimetics*, 13(3):036004, 2018.

[62] Floris Van Breugel, Kristi Morgansen, and Michael H Dickinson. Monocular distance estimation from optic flow during active landing maneuvers. *Bioinspiration & biomimetics*, 9(2):025002, 2014.

[63] Hann Woei Ho, Guido CHE de Croon, and Qiping Chu. Distance and velocity estimation using optical flow from a monocular camera. *International Journal of Micro Air Vehicles*, 9(3):198–208, 2017.

[64] Teppo Luukkonen. Modelling and control of quadcopter. *Independent research project in applied mathematics, Espoo*, 2011.

[65] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[66] James A. Preiss, Wolfgang Hönig, Gaurav S. Sukhatme, and Nora Ayanian. Crazyswarm: A large nano-quadcopter swarm. In *Proc. IEEE International Conference on Robotics and Automation*, 2017.

[67] Wolfgang Hoenig, Christina Milanes, Lisa Scaria, Thai Phan, Mark Bolas, and Nora Ayanian. Mixed reality for robotics. In *IEEE/RSJ Intl Conf. Intelligent Robots and Systems*, pages 5382 – 5387, Hamburg, Germany, Sept 2015.

[68] Farid Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 29(2):315–378, 2012.

[69] Zongyu Zuo. Trajectory tracking control design with command-filtered compensation for a quadrotor. *IET control theory & applications*, 4(11):2343–2355, 2010.

[70] Farid Kendoul, Zhenyu Yu, and Kenzo Nonami. Guidance and nonlinear control system for autonomous flight of minirotorcraft unmanned aerial vehicles. *Journal of Field Robotics*, 27(3):311–334, 2010.

[71] Robert Mahony, Vijay Kumar, and Peter Corke. Multirotor aerial vehicles. *IEEE Robotics and Automation magazine*, 20(32), 2012.

[72] Haijie Zhang and Jianguo Zhao. An integrated unmanned aerial vehicle system for vision based control. In *ASME 2017 Dynamic Systems and Control Conference*, pages V003T39A011–V003T39A011. American Society of Mechanical Engineers, 2017.

[73] Mirko Kovac. Learning from nature how to land aerial robots. *Science*, 352(6288):895–896, 2016.

[74] Michael Anderson. The sticky-pad plane and other innovative concepts for perching uavs. In *47th AIAA Aerospace Sciences Meeting Including The New Horizons Forum and Aerospace Exposition*, page 40, 2009.

[75] L. Daler, A. Klaptocz, A. Briod, M. Sitti, and D. Floreano. A perching mechanism for flying robots using a fibre-based adhesive. In *2013 IEEE International Conference on Robotics and Automation*, pages 4433–4438, May 2013.

[76] K. Zhang, P. Chermprayong, T. M. Alhinai, R. Siddall, and M. Kovac. Spidermav: Perching and stabilizing micro aerial vehicles with bio-inspired tensile anchoring systems. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6849–6854, Sep. 2017.

[77] C. E. Doyle, J. J. Bird, T. A. Isom, J. C. Kallman, D. F. Bareiss, D. J. Dunlop, R. J. King, J. J. Abbott, and M. A. Minor. An avian-inspired passive mechanism for quadrotor perching. *IEEE/ASME Transactions on Mechatronics*, 18(2):506–517, April 2013.

[78] H. Nguyen, R. Siddall, B. Stephens, A. Navarro-Rubio, and M. Kovač. A passively adaptive microspine grapple for robust, controllable perching. In *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 80–87, April 2019.

[79] Kaiyu Hang, Ximin Lyu, Haoran Song, Johannes A Stork, Aaron M Dollar, Danica Kragic, and Fu Zhang. Perching and resting—a paradigm for uav maneuvering with modularized landing gears. *Science Robotics*, 4(28):eaau6637, 2019.

[80] Larry L Howell. *Compliant mechanisms*. John Wiley & Sons, 2001.

[81] Thang-An Nguyen and Dung-An Wang. A gripper based on a compliant bitable mechanism for gripping and active release of objects. In *Manipulation, Automation and Robotics at Small Scales (MARSS), International Conference on*, pages 1–4. IEEE, 2016.

[82] Thomas George Thuruthel, Syed Haider Abidi, Matteo Cianchetti, Cecilia Laschi, and Egidio Falotico. A bistable soft gripper with mechanically embedded sensing and actuation for fast closed-loop grasping. *arXiv preprint arXiv:1902.04896*, 2019.

[83] Tyge Schioler and Sergio Pellegrino. Space frames with multiple stable configurations. *AIAA journal*, 45(7):1740–1747, 2007.

[84] Tian Chen, Jochen Mueller, and Kristina Shea. Integrated design and simulation of tunable, multi-state structures fabricated monolithically with multi-material 3d printing. *Scientific reports*, 7:45671, 2017.

[85] Minkyun Noh, Seung-Won Kim, Sungmin An, Je-Sung Koh, and Kyu-Jin Cho. Flea-inspired catapult mechanism for miniature jumping robots. *IEEE Transactions on Robotics*, 28(5):1007–1018, 2012.

[86] Je-Sung Koh, Eunjin Yang, Gwang-Pil Jung, Sun-Pill Jung, Jae Hak Son, Sang-Im Lee, Piotr G Jablonski, Robert J Wood, Ho-Young Kim, and Kyu-Jin Cho. Jumping on water: Surface tension–dominated jumping of water striders and robotic insects. *Science*, 349(6247):517–521, 2015.

[87] Tian Chen, Osama R Bilal, Kristina Shea, and Chiara Daraio. Harnessing bistability for directional propulsion of soft, untethered robots. *Proceedings of the National Academy of Sciences*, page 201800386, 2018.

[88] Jakob A Faber, Andres F Arrieta, and André R Studart. Bioinspired spring origami. *Science*, 359(6382):1386–1391, 2018.

[89] Stefano Mintchev, Jun Shintake, and Dario Floreano. Bioinspired dual-stiffness origami. *Science Robotics*, 3(20):eaau0275, 2018.

[90] Philipp Rothemund, Alar Ainla, Lee Belding, Daniel J Preston, Sarah Kurihara, Zhigang Suo, and George M Whitesides. A soft, bistable valve for autonomous control of soft actuators. *Science Robotics*, 3(16):eaar7986, 2018.

[91] Ahmad Alqasimi, Craig Lusk, and Jairo Chimento. Design of a linear bistable compliant crank–slider mechanism. *Journal of Mechanisms and Robotics*, 8(5):051009, 2016.

[92] Jesse L Silverberg, Jun-Hee Na, Arthur A Evans, Bin Liu, Thomas C Hull, Christian D Santangelo, Robert J Lang, Ryan C Hayward, and Itai Cohen. Origami structures with a critical transition to bistability arising from hidden degrees of freedom. *Nature materials*, 14(4):389, 2015.

[93] Babak Haghpanah, Ladan Salari-Sharif, Peyman Pourrajab, Jonathan Hopkins, and Lorenzo Valdevit. Multistable shape-reconfigurable architected materials. *Advanced Materials*, 28(36):7915–7920, 2016.

[94] Haijie Zhang, Jiefeng Sun, and Jianguo Zhao. Compliant bistable gripper for aerial perching and grasping. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1248–1253. IEEE, 2019.

[95] H. Zhang, E. Lerner, B. Cheng, and J. Zhao. Compliant bistable grippers enable passive perching for micro aerial vehicles. *IEEE/ASME Transactions on Mechatronics*, pages 1–1, 2020.

[96] Jiefeng Sun and Jianguo Zhao. An adaptive walking robot with reconfigurable mechanisms using shape morphing joints. *IEEE Robotics and Automation Letters*, 4(2):724–731, 2019.

[97] Anthony DeMario and Jianguo Zhao. Development and Analysis of a Three-Dimensional Printed Miniature Walking Robot With Soft Joints and Links. *Journal of Mechanisms and Robotics*, 10(4), 04 2018.

[98] Shorya Awtar and Shiladitya Sen. A Generalized Constraint Model for Two-Dimensional Beam Flexures: Nonlinear Load-Displacement Formulation. *Journal of Mechanical Design*, 132(8), 08 2010. 081008.