DESIGN OF AN AUTONOMOUS UNMANNED AERIAL VEHICLE FOR PHYSICAL INTERACTION WITH THE ENVIRONMENT

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Daniel R. McArthur

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

August 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF DISSERTATION APPROVAL

Dr. David J. Cappelleri, Chair

School of Mechanical Engineering

Dr. George T. Chiu

School of Mechanical Engineering

Dr. Xinyan Deng

School of Mechanical Engineering

Dr. Richard M. Voyles

School of Engineering Technology

Approved by:

Dr. Jay Gore

Head of the School of Mechanical Engineering Graduate Program

To my wonderful wife Sondra, and my four amazing daughters: Madilyn, Emma, Kate and Lacey—whose constant support, encouragement, sacrifices, and smiles made all of this possible.

ACKNOWLEDGMENTS

I would like to sincerely express my gratitude to Prof. David J. Cappelleri for his invaluable support, guidance and encouragement throughout the course of this work, and for always making time to give me guidance and direction when I faced obstacles in my studies or my research. I greatly appreciate the opportunities he gave me to expand my engineering knowledge and experience, and the tools, resources, facilities and funding he provided so I could carry out this work.

I would also like to extend my sincere thanks to Prof. George Chiu, Prof. Xinyan Deng and Prof. Richard Voyles for their insights, and for serving on my advisory committee.

I am grateful to the National Science Foundation, Purdue's School of Mechanical Engineering, and the donors of the Laura Winkelman Davidson Fellowship and the Helen and John Lozar Assistantship, for providing generous funding to support my graduate studies.

In addition, I am very appreciative of Prof. Patricia Davies and the faculty, students and staff at Herrick Laboratories for providing a supportive and inspiring workplace filled with opportunities to share my work with and learn from others.

I would also like to express my gratitude to all of my fellow students working in the Multi-Scale Robotics and Automation Lab. I would especially like to thank Arindam Bhanja Chowdhury, to whom I am deeply indebted, for his valuable support—spending countless hours working with me in the lab, discussing new techniques and algorithms, offering constructive criticism, encouraging me when I was struggling, and celebrating with me in my successes. I would also like to thank Tyler Landers for his help with building initial prototypes, and Ze An for his considerable support, particularly in the final year of my research.

Finally, I would like to thank my family: my parents and my siblings for their continual encouragement, and my immensely supportive wife and daughters, whose unwaivering positivity and encouragement kept me motivated to work hard, and reminded me to be happy and smile along the way.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

ABSTRACT

McArthur, Daniel R. Ph.D., Purdue University, August 2019. Design of an Autonomous Unmanned Aerial Vehicle for Physical Interaction with the Environment. Major Professor: David J. Cappelleri, School of Mechanical Engineering.

Unmanned aerial vehicles (UAVs), when paired with an onboard camera, have proven to be useful tools in many applications, including aerial photography, precision agriculture, and search and rescue operations. Likewise, UAVs capable of physically interacting with the environment have shown great potential to help people perform dangerous, or time-consuming tasks more safely and efficiently than they could on their own. However, due to onboard computation and battery life limitations and complex flight dynamics, using UAVs to physically interact with the environment is still a developing area of research. Considering these limitations, the primary goals of this work are to (1) develop a new UAV platform for aerial manipulation, (2) develop modular hardware and software for the platform to enable specific tasks to be performed autonomously, and (3) develop a visual target tracking method to enable robust performance of autonomous aerial manipulation tasks in unstructured, real-world environments. To that end, the design of the Interacting-BoomCopter UAV (I-BC) is presented here as a new platform for aerial manipulation. With a simple tricopter frame, a single additional actuator for generating horizontal forces, and lightweight, modular end-effectors, the I-BC aims to balance efficiency and functionality in performing aerial manipulation tasks, and is able to perform various tasks such as mounting sensors in hard-to-reach places, and opening small doors or panels. An onboard camera, force and distance sensors, and a powerful single board computer (SBC) enable the I-BC to operate autonomously in unstructured environments, with potential applications in areas such as large-scale infrastructure inspection, industrial inspection and maintenance, and nuclear decontamination efforts.

1. INTRODUCTION

Unmanned aerial vehicles (UAVs), when paired with an onboard camera, have proven to be useful tools in many applications, including aerial photography, surveillance, inspection, precision farming, and search and rescue operations. In particular, rotary-wing UAVs (as opposed to airplanes) are ideally-suited for operations that require the UAV to remain relatively still (such as up-close inspection and surveillance operations), since they are capable of vertical take-off and landing (VTOL), and can hover with zero translational velocity.

Researchers have been actively developing new designs and control strategies for rotary-wing UAVs (particularly quadrotors) for the past few decades. These efforts have produced robust attitude stabilization and position controllers (see [1–4]), but, until recently, complex flight dynamics and limitations in onboard computation power and battery life have limited small UAVs' ability to physically interact with the environment (e.g. to manipulate objects or transport payloads).

1.1 Motivation

Recent technological advancements, such as the development of powerful, lightweight single board computers (SBCs), lighter, more energy-dense batteries, and 3D-printing techniques for rapid prototyping, have helped enable the development of new UAV platforms that are capable of both observing and physically interacting with the environment through aerial manipulation. These manipulations may involve pushing, pulling, grasping, sliding, rolling, twisting, etc., and are generally performed to achieve some overarching goal: placement, assembly or disassembly of components, transportation of cargo, etc. With applications in search and rescue [5], construction [6, 7], infrastructure inspection and maintenance [8–10], and many more, UAVs capable of performing aerial manipulation tasks have shown great promise as tools that can take people out of harm's way and help people perform dangerous or time-consuming tasks more safely and efficiently than they could on their own. However, in order to realize the full potential of these interactive UAVs on a large scale and in real-world scenarios, much research remains to be done to develop their safety, efficiency, and autonomy. In the following sections, we will review recent research efforts in the literature concerning two fundamental challenges associated with autonomous aerial manipulation. First, we will consider several issues related to the design of the vehicle itself, and second, we will consider issues related to the localization of UAVs in unstructured environments and the localization of target objects for aerial manipulation.

1.2 Literature Review: Vehicle Design for Aerial Manipulation

To date, a large portion of the research in the field of aerial manipulation has made use of traditional rotary-wing UAV platforms (e.g. quadrotors) which benefit from mechanical simplicity and stable flight performance, and can be produced at a low cost. However, most of these vehicles are underactuated (either because they have fewer actuators than degrees of freedom (DOF), or because they have several actuators mounted and oriented in the same plane), and are thus unable to independently control forces and torques in all 6 DOF. For example, in order for a quadrotor to move forward in the horizontal plane, it must first change its orientation by tilting so that the thrust from its main rotors has a component in the forward direction, thus resulting in a forward acceleration. This coupling between position and attitude limits the pose trajectories attainable by underactuated UAVs, and, as a consequence, limits their ability to interact with the environment since physical interaction tasks may require the ability to instantaneously resist or apply arbitrary forces and torques in any of the 6 DOF. In order to perform aerial manipulation tasks, the limitations of underactuated UAVs need to be resolved. Three key ways to address these limitations will be considered in the following sections: 1) adding to or changing the configuration of the primary actuators of a UAV to increase the number of independently controllable DOF, 2) adding a robotic arm with one or more DOF to augment a UAV's total number of independently controllable DOF, and 3) considering physical interaction tasks that only require the application of forces or torques in a limited number of DOF.

1.2.1 Novel Actuator Configurations

Various actuator configurations have been investigated recently to improve upon the performance of traditional underactuated rotary-wing UAV platforms. As discussed above, these improvements have the potential to enhance or enable various aerial manipulation capabilities.

Reconfigured UAV Platforms

Consider, for example, a typical hexrotor, which has six main rotors mounted in a single horizontal plane with each rotor rotating about a vertical axis. In [11], each of the primary rotors of a hexrotor were mounted to its frame with a fixed rotation ($20°$) about the central axis of their radial mounting arms (see Fig. 1.1). Since the rotor thrust vectors are no longer strictly vertical, the combined thrust vectors of all six rotors span the space of Cartesian forces and torques, thus making the vehicle (dubbed the Dexterous Hexrotor) fully actuated. With this actuator configuration, it was shown that the vehicle can hover in one place with a tilted orientation ($-10°$ roll angle), and translate while maintaining a level attitude. Later work determined optimal hexrotor tilt angles based on a variety of objectives, including maximizing force exertion capacity and flight efficiency [12], maximizing the vehicle's dynamic manipulability measure [13], and minimizing the control effort required for a desired flight trajectory [14]. The control of a hexrotor with tilted variable-pitch propellers has also been proposed in [15], with simulations demonstrating its ability to track a position trajectory and rotate in place.

The advantages of dynamically-changeable rotor tilt angles have also been considered on tricopters [16], quadrotors [17–19], and hexrotors [20] with the addition of actuators that allow the primary rotors to tilt during flight. In [16], the two front rotors of a tricopter tilt forward to generate large longitudinal pushing forces ($22\ N$) approximately equal to the full

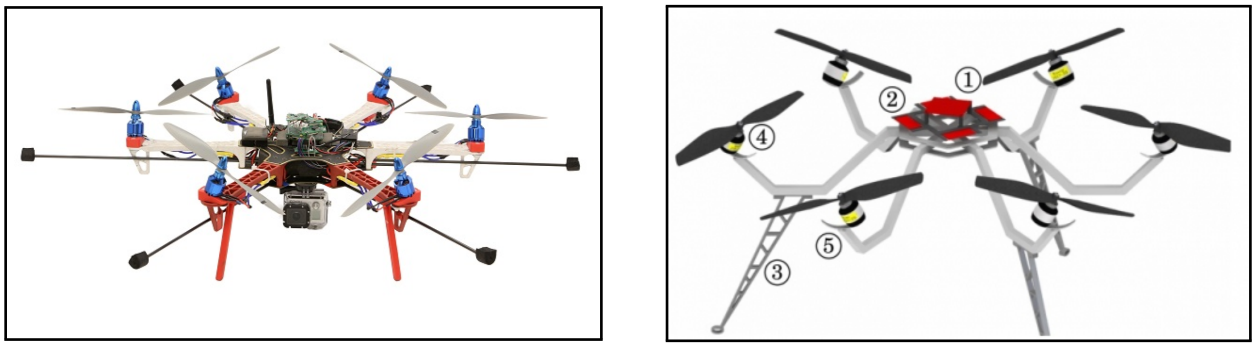


Figure 1.1. : Fully-actuated hexrotor platforms. Left: Dexterous Hexrotor [11]; right: CAD rendering from [14].

weight of the vehicle. In [17], dual axis tilting propellers are used to increase the agility and fault tolerance of a quadrotor, but the vehicle's full actuation for complex pose control is not analyzed. However, [18] presents the Holocopter quadrotor with tilting propellers that can hover in place with high precision (average error $\approx 1.7\ cm$), follow an eight-shaped trajectory while simultaneously tracking a sinusoidal pitch rotation of the vehicle's body, and remain still in space while rotating up to $\approx \pm 55°$ around the roll and/or pitch axes. Similarly, [19] presents the quad tilt rotor UAV, which has four main rotors mounted on servo-actuated arms. This setup allows the vehicle to tilt up to 260° from level, and experimental results demonstrate a stable transition between level (0° roll/pitch) and perpendicular (90° pitch) hovering. This capability provides the possibility of flying in narrow spaces or potentially interacting with a vertical surface.

New UAV Platforms

In addition to reconfiguring traditional UAV platforms, several research groups have also developed novel UAV configurations that are fully or overactuated. Similar to the tilt-rotor UAVs described above, the overactuated Omnicopter UAV [21] uses three variable-angle ducted fans to control its roll and pitch angles and to provide lateral forces, however, its primary thrust and yaw control are generated by two coaxial counter-rotating propellers at the center of the vehicle frame (see Fig. 1.2). Thus, the Omnicopter is able to achieve horizontal translation while maintaining a level attitude.

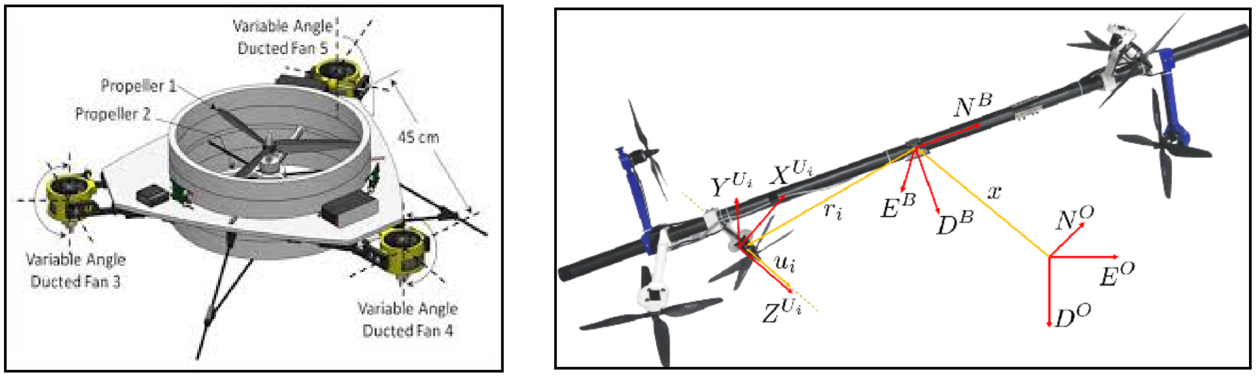


Figure 1.2. : Left: Fully actuated Omnicopter UAV [22]; right: ODAR (omni-directional aerial robot) [23].

More recently, optimization techniques have been used to design unique UAV configurations [23–25]. In [23], pairs of symmetrically-attached (with respect to the main vehicle frame) reversible rotors were considered as basic actuator elements, and the optimal attachment location of each rotor pair on a single carbon-fiber tube (the main vehicle body) was determined with the goal of maximizing the control wrench generation. Experiments performed with the resulting ODAR (omni-directional aerial robot) platform (see Fig. 1.2) demonstrated its ability to simultaneously track a circular position trajectory and a sinusoidal pitch angle trajectory, perform vertical pitch-flipping (virtual pivoting around one of its ends), and exert a large downward force of $\approx 30\ N$. Unlike [23], [25] preselected the number and orientation of rotors (eight rotors fixed to the vertices of a cube) to minimize the dependence of the vehicle's dynamical characteristics on its orientation, and then determined the rotor orientations by maximizing the vehicle's agility (measured by the 2-norm of the maximum attainable force-torque output in any direction). While the resulting vehicle is capable of accelerating up to 24 m/s^2 in any direction (not considering gravity), and can simultaneously track position and orientation trajectories, its controller (using a simplified force and torque model) was not able to fully decouple the vehicle's translational and rotational dynamics, thus resulting in fairly large position and angle tracking errors (up to $\approx 0.1\ m$ and $\approx 5^\circ$, respectively).

1.2.2 UAVs Augmented with Robotic Arms

Another approach to resolving the problem of underactuation in aerial manipulation research is to endow a traditional UAV platform with one or more robotic arms, thus increasing its total number of controllable DOF.

Some of the early efforts in this area used ducted-fan UAVs or quadrotors along with a multiple-DOF manipulator attached to the side of the airframe to make contact with vertical surfaces. [26] presents the design of a manipulation system composed of a 3-DOF Delta manipulator and a novel 4-DOF end-effector for performing inspection-by-contact tasks, e.g. non-destructive testing (NDT) with an ultrasonic sensor. This manipulator can perform Cartesian movement to compensate for UAV dynamics to help place sensors and objects at desired locations. In [27], the manipulator was attached to a commercially-available AscTec Pelican quadrotor, and an impedance controller was designed to allow the system to contact a vertical surface. Experimental results confirmed that the system is able to remain stable while applying a force to a wall. In a similar approach, [28] also demonstrated a stable docking capability between a ducted-fan UAV and a vertical surface.

Considering more complex aerial manipulation scenarios, [29] determined that a multiple-DOF robotic arm could be mounted underneath a helicopter to perform aerial manipulation tasks, however, it was noted that 1) the control system models are complicated due to the dynamic coupling between the manipulator and helicopter, and 2) even small interaction forces between the manipulator and helicopter can lead to low-frequency instabilities and oscillations if the manipulator motion is not restricted to a single dimension or else compensated for in the helicopter flight controller. In a related work, [30] demonstrates the first use of a fully actuated redundant robot arm mounted underneath a helicopter (see Fig. 1.3) to perform an aerial manipulation task (grasping a pole and pulling it out of a fixture).

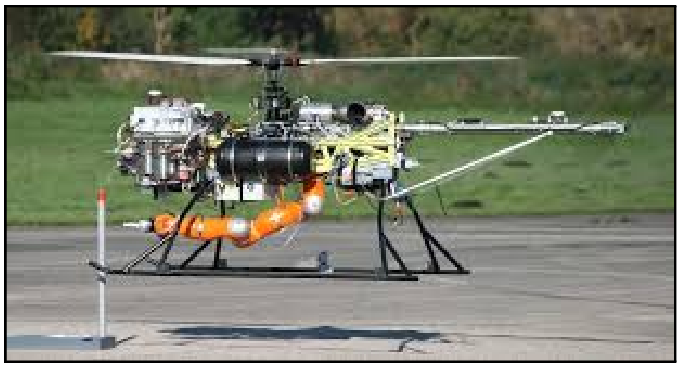
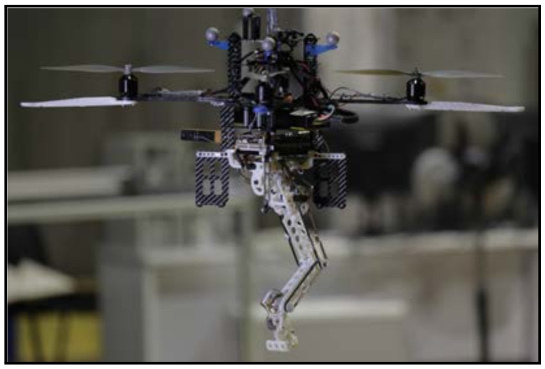



Figure 1.3. : Left: Helicopter with fully actuated robotic arm [30]; right: PUL5AR 5-DOF robotic arm mounted under AscTec Pelican UAV [31].

Several other research groups have focused specifically on the design of robot arms for aerial manipulation. For instance, [32] presents a 3-DOF parallel manipulator (mounted beneath an AscTech Neo hexrotor) that provides a large planar workspace in front of the vehicle, [31] presents the 5-DOF PUL5AR robot arm (see Fig. 1.3) that minimizes kinematic coupling between its motion and the UAV attitude, and [33] presents a 3-DOF compliant robot arm with a compliant finger and torque estimation for applications that involve contact forces between the robotic arm and the environment.

1.2.3 Limited-DOF Aerial Manipulation Tasks

Having considered various UAV and manipulator configurations that enable or enhance physical interaction capabilities for UAVs, let us now consider the interaction tasks themselves. Specifically, we will consider three subsets of aerial manipulation tasks that do not require full actuation in all 6 DOF, namely object transport and assembly, inspection by contact, and lightweight maintenance and perching.

Object Transport and Assembly

The first instance of unstructured object retrieval with an aircraft-mounted gripper was performed by the Yale Aerial Manipulator [34], which consists of a 4 kg helicopter equipped with a custom, compliant gripper that balances loads across each of its four fingers (see Fig. 1.4). The object retrieval was performed by manually flying the vehicle above the object, switching to an autonomous position hold mode as the gripper was closed on the object, and then increasing the thrust to lift the object off the ground. Successful object retrieval was demonstrated 18 times with objects including a softball (160 g, 89 mm), PVC cylinder (900 g, 390 mm), wood block (700 g, 265 mm), and a weighted tool case (1.45 kg, 335 mm).

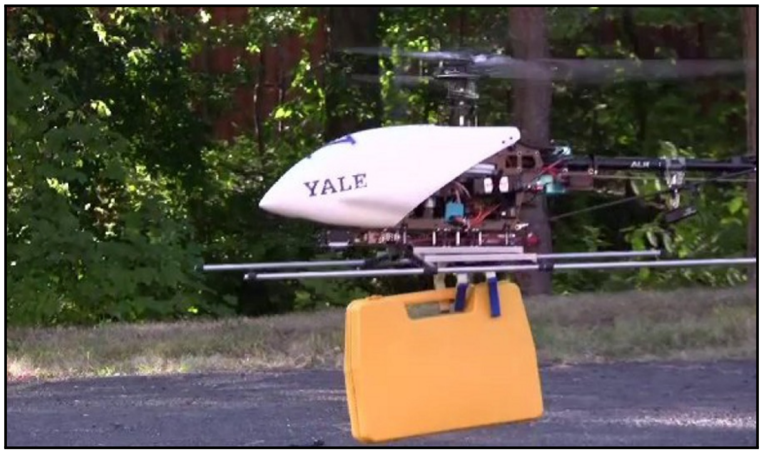


Figure 1.4. : Yale Aerial Manipulator carrying a toolcase [35].

In a similar effort, [34, 36–38] each examine the task of grasping and/or manipulating an object from above with a quadrotor. [36] utilizes both impactive grippers (which use clamping motions and frictional forces) and ingressive grippers (which penetrate into surfaces with metal hooks) to grasp objects at predefined locations, and develops methods for identifying important flight parameter changes (e.g. moments of inertia, mass, and center of mass offset) that occur when carrying an object. Experiments that included these parameter changes in the controller design showed significant improvements in position-tracking performance.

In contrast to the above experiments, which have objects at predefined locations, [37] uses a custom compliant gripper and an IR camera (with an IR LED marker placed next to the object) to locate and grasp an object (150 g stuffed toy), and [38] uses a custom electro-permanent magnetic gripper and downward facing RGB camera to detect and grasp both static and moving objects. In the latter work, dozens of experiments showed that the proposed visual-servoing method enabled the UAV to autonomously detect, grip, and transport a static object to a drop-off location with a success rate $> 95\%$, even in the presence of wind disturbance up to 15 m/s. In the case of objects moving linearly in an arbitrary direction with a velocity of 0.1 m/s, the success rate dropped to 78%.

Exploiting a quadrotor's ability to independently control its 3D position and heading (yaw angle), [6] explores the task of assembling three-dimensional cubic structures composed of beams and columns with magnetic ends. A simple 1-DOF gripper (19 g) with a pair of fingers attached beneath a quadrotor (AscTec Hummingbird) is used to pick up each component from a bin and then assemble it into a cubic structure (by placing it at a desired position from above). Error detection is achieved without the need of additional sensors in two ways: first, when picking up a component, the thrust-to-hover is compared to the nominal thrust required to hover with the weight of the payload, and second, while still grasping a component that has been placed on the structure, the UAV executes an open-loop yaw moment and, if the vehicle's yaw angle remains at the setpoint, then the part is considered to have been fixed in place.

Inspection By Contact

Much of the research in the area of inspection by contact is motivated by potential industrial applications, such as large-scale infrastructure inspection, and inspection and maintenance of power plants, wind turbines, etc. [28, 39, 40]. Similar to the first example in section 1.2.2, [41, 42] consider tasks that require physical contact with a vertical surface. In [41], impact-control strategies are considered for a ducted-fan miniature UAV (DFMAV) that enable docking with and sliding against a wall. Similarly, [42] presents the design of the AIRobots CoaXial-rotor prototype (ACX) which is able to achieve stable docking and controlled sliding on walls. The ACX has a pentagon-shaped frame and its docking mechanism uses compliant materials with integrated force sensing resistors (FSRs) to provide force feedback during the docking and sliding maneuvers. Experiments demonstrated the ACX's ability to slide horizontally while maintaining contact against a wall.

Considering the challenge of handling impacts during highly dynamic physical interaction tasks, [43] developed a manipulator system composed of a rod and a locking mechanism that converts the kinetic energy of a collision into potential energy (stored in the manipulator). Experiments performed with the manipulator attached above a quadrotor showed that, for impacts at $\approx 2\ m/s$, the peak force exerted on the target was reduced by 78% (from 54.2 N to 11.8 N) when the locking mechanism was used. This allowed the vehicle to maintain stable contact with the wall after impact, whereas a collision without the locking mechanism destabilized the flight controller and led to a crash.

Lightweight Maintenance and Perching

In addition to inspection-by-contact manipulation scenarios, several lightweight maintenance and perching tasks have been considered that cover a wider range of applications. For example, [44,45] consider the application of horizontal forces that could be used for inspection tasks or to perform cleaning or grinding tasks. In [44], an extra propeller is added below a quadrotor to provide horizontal thrust (see Fig. 1.5). It was shown through experimentation that the vehicle (with a brush attached on the front) could maintain flight stability while applying a desired force (up to 5 N) to a surface for an extended period of time (50 s in the experiment). In more recent work, [45] developed a controller for a quadrotor that enables large force exertion against a vertical surface (with a simple rod end-effector attached above and extending in front of the vehicle). With this controller, it was shown that a contact force equal to the weight of the UAV (15 N) can be sustained against a surface for several minutes.

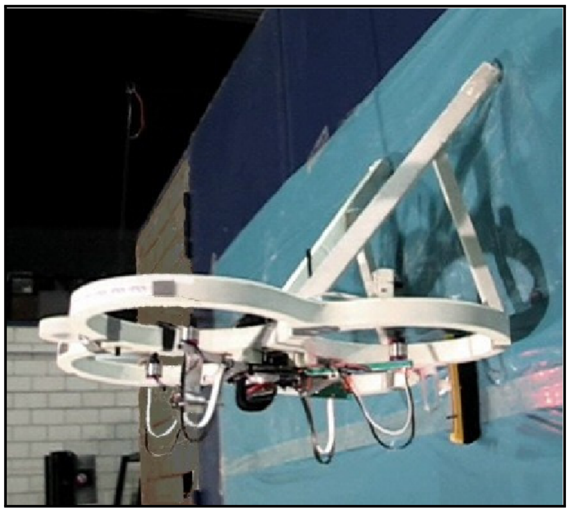
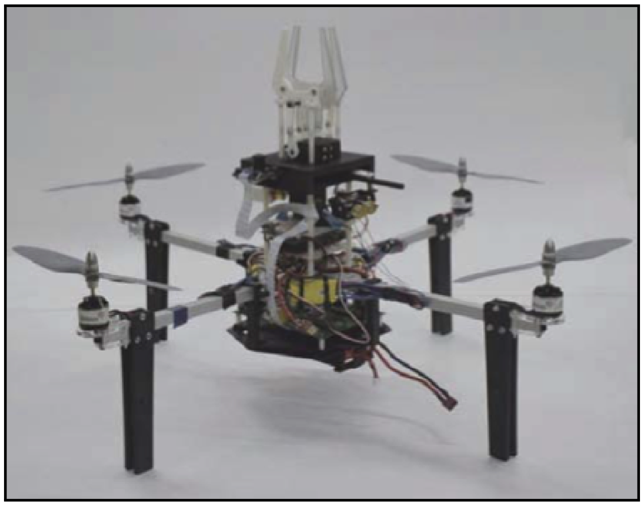



Figure 1.5. : Left: A semi-autonomous flying robot applying force to a wall [44]; right: quadrotor with gripper for manipulation above the airframe [46].

For use cases where the workspace above the UAV is of particular interest, such as under-bridge or in-tunnel inspection and maintenance, [46] presents the design of a quadrotor with an upward-facing hand (see Fig. 1.5). The custom hand has 1-DOF for opening and closing, can slide in one dimension, and is controlled autonomously with vision feedback from an upward-facing camera. To demonstrate the autonomous gripping capabilities of the system, the quadrotor was flown manually beneath a 3 cm bar placed 1.5 m above the vehicle, and, as the vehicle approached the bar, the hand moved autonomously along a slider to align with the bar, and gripped the bar once it was within reach. At this point, the main rotors were stopped to further demonstrate that the gripper is able to support the weight of the vehicle to enable perching by hanging. This capability could be used to increase the operation time of UAVs while they perform surveillance or inspection operations.

Other research groups have also considered UAV perching in the context of aerial manipulation, including [47] and [48]. [47] demonstrates the ability to reliably and reversibly perch on a vertical surface, using a manipulator with a vacuum-cup. The manipulator has two carbon-fiber legs below the main arm with the vacuum-cup to help support the weight of the vehicle, and the main arm uses the mechanism presented in [43] (discussed above) to absorb the impact force generated during collision with the wall. Experiments have verified that the UAV can successfully perch on a wall, power down its main rotors, and then takeoff again.

In contrast to the perching methods proposed above, which are intended to increase flight time, [48] considers a door-opening task which requires the force from all of the main rotors. In a test flight, vacuum cups are used during flight to attach the front of a quadrotor to a door and then the vehicle rotates until it is parallel to the door and a second pair of vacuum cups secures the vehicle to the door. Once perched on the door, a custom, inflatable arm is used to push a lever-style door handle to unlatch the door, and then the main rotors use their maximum thrust to push the door open to $\approx 60^\circ$.

1.2.4 Summary of Vehicle Design for Aerial Manipulation

In summary, the vehicles described in section 1.2.1 have many desirable attributes for performing aerial manipulation, including the ability to apply forces and torques in any direction, which provides simultaneous control over position and orientation in all 6 DOF, and (in some cases) the ability to apply large forces relative to the weight of the vehicle. However, these benefits come at the cost of reduced flight efficiency due to the weight of additional actuation mechanisms, and the thrust lost to internal forces in the case of UAVs with tilted propellers.

As discussed in section 1.2.2, several successful experiments have demonstrated the feasibility of performing aerial manipulation tasks using traditional underactuated UAV platforms endowed with robotic arms. In particular, the use of these augmented UAVs shows great promise for aerial manipulation scenarios that require high dexterity, cover a large workspace, or involve complex motions. However, the added weight of the robotic arm will reduce the payload capacity and flight time of the UAV, and, as mentioned above, the motion of the arm can add significant complexity to the control design and potentially destabilize the UAV.

Finally, section 1.2.3 highlighted several tasks that do not require full actuation in 6 DOF, such as object transport and assembly, inspection by contact, and lightweight maintenance and perching. Many of these tasks could have direct applications to infrastructure and industrial inspection and maintenance. However, in some cases, the UAVs designed to perform these tasks are highly specialized, that is, they are designed for only one task, thus potentially limiting their viability for adoption at a large scale.

1.3 Literature Review: UAV Localization for Aerial Manipulation

Having considered several challenges associated with the design of a UAV platform capable of performing aerial manipulation tasks, we will now consider recent research efforts in the area of UAV localization and target tracking for aerial manipulation.

Precise mobile robot navigation (on the ground or in the air) is a complex and challenging task that has received significant attention in the literature, and continues as an active area of research in the robotics community. In the case of aerial robots performing manipulation tasks, an even higher level of accuracy is required during navigation to ensure that contact between the UAV and objects in the environment is properly controlled. Furthermore, UAVs have additional limitations that make precise navigation more challenging, including weight and computational constraints, additional noise in sensor data and camera images induced by the constant motion and vibration of the rotors, and more susceptibility to external disturbances such as wind. These challenges have been addressed in the literature in a variety of different ways. In the following sections, we will review research efforts related to two crucial aspects of UAV navigation that are prerequisites for autonomous execution of aerial manipulation tasks in real-world scenarios: 1) UAV localization in unstructured environments, and 2) UAV motion control relative to a target object.

1.3.1 UAV Localization in Unstructured, GPS-denied Environments

Siegwart et al. [49] described the following four subsystems as the building blocks of navigation: perception (extracting useful information from sensor data), localization (determining the robot's position relative to its environment), cognition (deciding appropriate actions to take to accomplish tasks), and motion control (controlling motor outputs to achieve a desired trajectory). Successful navigation requires careful and accurate integration of each of these subsystems. Typically, solutions for the issues of perception, cognition and motion control of UAVs are developed for a particular vehicle type or use case, whereas the issue of UAV localization is applicable to a much broader range of scenarios, and has thus received substantial attention in the literature in the past decade.

UAV localization is most often approached as part of the broader simultaneous localization and mapping (SLAM) problem [50, 51], or through the implementation of visual odometry (VO) [52–55]. Several examples from the literature that apply SLAM, or VO algorithms on UAV platforms are included below, following a brief overview of SLAM and VO.

Overview of SLAM

SLAM consists of using onboard sensors to estimate a robot's state (position, orientation, velocity, etc. over time), while simultaneously creating a map of the objects of interest in the robot's environment (e.g. obstacles, landmarks, etc.) [56]. Given the robot's pose (position and orientation) within the map, other tasks like path planning and obstacle avoidance can then be performed. The map also serves as a way to limit the accumulated error (drift) in the state estimation, by resetting or adjusting the state estimate when a unique landmark has been revisited (this resetting process is referred to as *loop closure*). However, there are many instances where it is infeasible or unnecessary to produce a complete map of the environment. Some examples include industrial settings such as warehouses, factory floors, etc. where beacons may be used to manually create a map of important landmarks, or in outdoor settings with access to GPS. In these instances, full SLAM may not be necessary, so long as the robot can localize sufficiently well relative to the known landmarks.

Many SLAM implementations make use of laser scanners as a robust and reliable input source [57] (particularly for 2-D SLAM), however, the weight and power consumption of these devices limits their use on lightweight UAVs [58]. In recent years, the light weight, low cost, and ubiquity of camera sensors, combined with significant improvements in computing hardware has led to the development of many vision-based SLAM algorithms (VSLAM) which use cameras as their primary sensor input [59].

Overview of VO

VO uses a sequence of images to estimate the translational and rotational motion of a camera relative to a reference frame. VO is closely related to VSLAM, but focuses primarily on the local consistency of the robot trajectory estimate from pose to pose, rather than creating a globally-consistent estimate of both the robot pose and a map of the environment [55]. And, in fact, many VSLAM algorithms use VO for their localization component. VO approaches employ two general techniques to calculate the velocity or *flow* of an image: feature matching/tracking (where visual features are matched over a number of frames or in adjacent frames), and optical flow (OF) techniques (where spatio-temporal variations in pixel intensity are measured between sequential images) [60]. Typically, the feature matching/tracking methods are more accurate than optical flow techniques, but are more complex and require higher resolution images and more computing power. On the other hand, optical flow techniques perform better in scenes without recognizable features, such as those with repetitive patterns, blurry textures, or poor lighting conditions [61].

Similar to VO, visual-inertial odometry (VIO) [62, 63] combines data from an IMU to improve the accuracy and robustness of the VO estimation process.

UAV Localization with SLAM and VO

The first work demonstrating the ability of a UAV to navigate through an unknown environment (without the aid of external positioning sources such as GPS or beacons) was performed by Blösch, et al. in 2010. In this work, a VSLAM algorithm (PTAM [64]) was implemented with a downward-facing camera on a quadrotor UAV. Due to onboard computational constraints, the SLAM and position controllers were both implemented on a ground station computer. Using this setup, autonomous hovering and rectangular path following in an indoor lab setting were achieved with absolute RMS errors of 4.61 *cm* and

13.1 *cm*, respectively. However, there are a few important limitations worth noting: 1) the map scale must be adjusted manually, and, since a single camera is used to build the map, instabilities leading to a crash may arise if the scale and orientation of the map drift, 2) it is possible for the localization to fail due to a lack of visual features in the camera's field of view, or varying illumination conditions, and 3) if the map grows too large, increased computational cost will cause the SLAM algorithm to stop updating.

Using the same PTAM algorithm for position estimation of the UAV, [37] (described in Sec. 1.2 above) overcame the issue of scale drift by pre-mapping a small area near the takeoff location in order to initialize the map scale.

In later work, several other groups have used a similar approach for position control of UAVs. For example, the European project SFLY [58] fused the output of PTAM with IMU data on a hexrotor UAV to engage in autonomous navigation, 3D mapping, and optimal-coverage flights in GPS-denied environments. The increased payload capacity of the hexrotor platform allowed for a larger onboard computer (Intel Core 2 Duo) capable of running the algorithm for local navigation, and an offboard computer performed the global navigation (expressing the poses of multiple coordinating UAVs in a common coordinate frame). The total position drift of the estimator was reported as 1.5 *m* for a 350 *m* trajectory (0.43% drift), but the plotted X-Y-Z position of the UAV compared to gound-truth GPS measurements from the flight indicates occasional positioning errors on the order of a few meters.

In a similar effort, [65] achieved autonomous UAV navigation by estimating the so-called 2.5D pose (X-Y position and yaw orientation) of a quadrotor using a scanning laser range sensor as its primary source of position/yaw estimation. Using the laser scans, 2D SLAM maps were generated while flying across multiple floors of a building. An additional forward-facing camera was also added to the UAV to help aid in loop closure and to distinguish between floors. The output of the onboard estimator was compared to ground truth (a Vicon motion capture system [66]) while hovering, and while traveling along a specified trajectory. The standard deviation of the errors were reported for each flight, respectively, as follows: $\{\sigma_x, \sigma_y, \sigma_z\} = \{8.49, 9.11, 5.54\}$, and $\{\sigma_x, \sigma_y, \sigma_z\} = \{2.47, 3.23, 0.70\}$ (units in cm).

These efforts have demonstrated that full SLAM approaches can be applied in the autonomous control of UAVs, but the computational complexity of various SLAM methods, especially in 3D, often requires some portion of the algorithm (typically the mapping/loop closure) to be offloaded to a ground station computer. However, considering that a full map of the environment is not necessary in many applications, many recent research efforts have focused on using just VO for UAV localization.

In an early work, Kendoul et al. [67] developed an OF-based approach to VO, which enabled a quadrotor UAV to autonomously takeoff, hover in place, and follow a given trajectory in both indoor and outdoor settings using only a single, downward-facing camera and an IMU as inputs. The UAV attitude and position controllers ran onboard, while the OF calculations were performed on a ground computer based on images streamed wirelessly from the onboard camera. Multiple flight tests were performed to compare the position estimation of the OF-based position/velocity controller with a low-cost GPS ($\pm 2\ m$ horizontal position accuracy) as ground truth. Flight tests demonstrated that OF-based velocity estimates were at least as accurate as the GPS velocity estimates, and that the vision-based horizontal position estimates were more accurate than the GPS position measurements, although GPS measurements are more robust for long-distance outdoor flights. This work also noted that the OF-based VO computation struggles in certain low-texture environments (such as a large, smooth concrete floor). However, the approach still performed well at a high altitude over a large field where the only visual features in the field of view were grass and dirt.

On a much smaller scale, [61] implemented OF-based VO on a 46 g quadrotor using five optical flow sensors typically used in high performance computer mice. Two flights in unstructured environments (outdoors near vegetation, and indoors in a cluttered office) demonstrated that the quadrotor could maintain its position while undergoing controlled vertical oscillations at 0.5 Hz. Additional tests using the same sensor suite on a larger quadrotor in a motion capture space showed that the positional drift was about 50 cm on average after 2 minutes of flight (also with controlled vertical oscillations at 0.5 Hz). This approach uses very lightweight sensors and an optical flow algorithm that can be implemented on a simple microcontroller, and thus presents a minimal-weight solution to the UAV localization problem. However, while this work provides no discussion about lighting conditions, Honegger et al. points out the need for strong lighting to provide accurate measurements as a key disadvantage of computer mouse sensors [68].

In an effort to provide a more robust OF sensor for mobile robot navigation, [68] presents the open source and open hardware PX4Flow module. This module incorporates a CMOS sensor (with automatic exposure control), an ultrasonic range sensor, an onboard gyroscope, and an ARM Cortex M4 CPU to compute optical flow at a rate of 250 FPS. The accuracy of the PX4Flow sensor was assessed by integrating the OF X and Y velocities over a square path (starting and stopping at the same point) traversed in an indoor, carpeted office environment. Compared to ground truth (Vicon), the total positional drift over a 28.44 m trajectory was 0.25 m (0.88% drift). In a manual, outdoor flight test traversing 192.16 m, the PX4FLOW showed similar position estimation results to a computer mouse sensor, but no ground truth measurements were provided for this test.

Recently, Delmerico and Scaramuzza provided a benchmark comparison of several publicly-available Visual-Intertial Odometry algorithms for flying robots [69]. The performance of each algorithm was evaluated using the EuRoC datasets [70], which consist of 11 stereo image sequences, synchronized with IMU data, captured from a front-down facing camera on a hexrotor UAV at 20 Hz, along with ground-truth measurements from a Leica Multistation and a Vicon motion capture system [70, 71]. The datasets include slow flights in two small, cluttered rooms, and more dynamic flights in a large industrial machine hall. Some of the algorithms included in the comparison are SVO [72], OKVIS [73], ROVIO [62], VINS-Mono [74], etc. In the comparison, each of the algorithms were implemented on various computing platforms, including SBCs typically used on UAVs, and a laptop computer. The results were reported using four metrics: absolute translational error (RMSE), CPU usage, memory usage, and per-frame processing time. A summary of the CPU usage versus RMSE reported for each algorithm is shown in Fig. 1.6. This figure highlights the primary conclusion of the paper: the accuracy and robustness of visual state estimation can be improved with additional computation, but for systems with limited computational resources, it is difficult to find the right balance between estimator performance and computational load.

In the context of aerial manipulation applications, where there are more strict requirements on positioning accuracy, it is important to note from Fig. 1.6 that the best performance (on average) that can be expected across all algorithms implemented on an SBC is $\approx 10\ cm$ RMSE in X-Y-Z position estimates. These results are consistent with the results shown in Labbé and Michaud's recent, more comprehensive comparison of several VSLAM and VO methods [75]. In that work, the performance of several different VO and RGB-D SLAM methods were compared across multiple datasets, including the EuRoC dataset used in [69], with computations performed on a desktop Intel Core i7-3770 processor. Averaging the results across all of the EuRoC datasets, the translational RMSE for the VO methods ranged from $\approx 7\ cm$ to $28\ cm$ and the range for the VSLAM methods was $\approx 4\ cm$ to $8\ cm$.

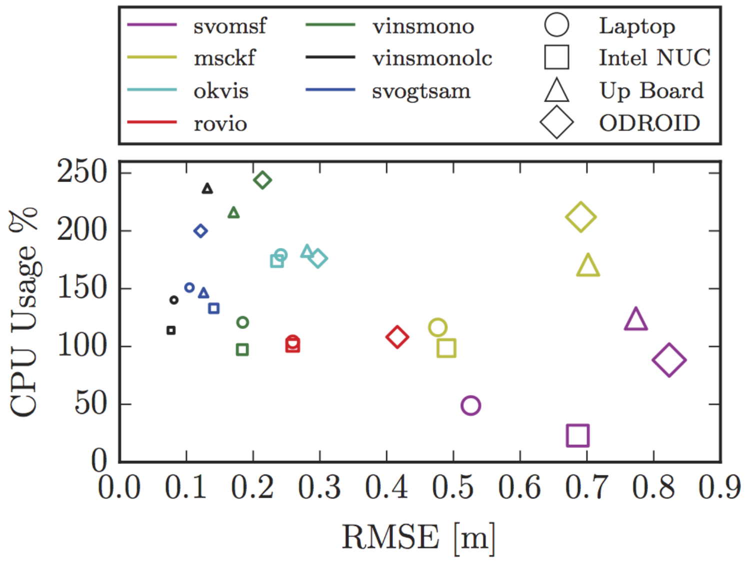


Figure 1.6. : CPU Usage vs. RMSE for several VIO algorithms (adapted from [69]).

1.3.2 UAV Motion Control Relative to a Manipulation Target

Having discussed the issues associated with precise UAV localization, we will now consider the challenge of positioning and orienting the UAV relative to a manipulation target object. Similar to the topic of visual robot localization, visual object tracking is a well-established field of research, and recent efforts continue to improve the accuracy, robustness, and computational efficiency of the algorithms for use in real-world applications on UAVs [76–79]. However, since visual tracking algorithms are dependent on so many different variables (e.g. image lighting conditions and background, input sensor type (RGB, depth, etc.), characteristics of the objects to be tracked, expected amount and type of camera motion, etc.) no single tracking algorithm exists that can be used in all applications [80]. Thus, here we will focus mainly on how the UAV is to be controlled relative to a target object after it has been detected.

Overview of Visual Servo Control

The concept of using computer vision to control the motion of a robotic system is referred to as visual servo control, or visual servoing, and is a well-established approach to autonomous robotic manipulation [81,82]. The visual servoing problem is typically approached using one of two schemes: image-based visual servo control (IBVS), or position-based visual servo control (PBVS). In IBVS, a set $\mathbf{s}$ of one or more important visual features (such as the centroid of an object) is selected using image-plane coordinates (pixel location), and the motion of the camera/robot is controlled so that the error between $\mathbf{s}$ and a desired final state $\mathbf{s}^*$ is minimized (e.g. align the centroid of the object in the center of the image). PBVS, on the other hand, defines the set $\mathbf{s}$ as the 3D pose of the visual feature(s) with respect to the camera, and the motion of the robot is controlled to minimize the error between $\mathbf{s}$ and $\mathbf{s}^*$ with respect to a 3D reference frame.

Many of the methods used for autonomous control in aerial manipulation tasks to date are based on relatively simple object tracking methods, and have focused primarily on vertical manipulation tasks (such as payload collection/transport) with artificial markers near the target. For example, in [37], an IR LED is co-located with an object to be grasped, and a Nintendo WiiMote camera is used to detect the IR blob. A PBVS-type control scheme is used in which a UAV's desired position setpoint is updated based on the estimated 3D location of the IR blob in the global frame. Using this controller in an indoor flight test, the UAV successfully aligned with the manipulation target, and descended to grasp the object, although the initial UAV position was < 3 *cm* away from the object in the horizontal plane (and 50 *cm* above the object).

Using a similar approach, [83] considered the more complex task of automatic aerial retrieval of a mobile robot. For object detection, an ARToolkit visual marker (9 *cm* square) was placed on the mobile robot, and a downward-facing camera connected to an onboard computer was used to detect the marker and provide a position and orientation estimate of the marker relative to the camera [84]. After calibration, the standard deviation of the estimated object location (X-Y-Z) was reported as: $\sigma = [2.2\ cm, 1.7\ cm, 4.2\ cm]$ (5.0 *cm* total). With a success rate of $\approx 63\%$, several outdoor flight tests were performed where the UAV flew to a given GPS coordinate above the mobile robot, and then used PBVS control based on the detected marker location to descend and retrieve the mobile robot with an electromagnet. The magnetic retrieval mechanism used a funnel that required a horizontal positioning accuracy of 7 *cm* relative to the mobile robot. These results demonstrate that aerial manipulation tasks can be performed autonomously without the need of a global map of obstacles/features in the environment (only GPS coordinates were used).

In a more recent work, [85] (described in Sec. 1.2 above) used an IBVS control scheme to grasp an object using a robotic arm attached to a quadrotor. A blue, rectangular LED panel was mounted near the object to aid in detection (see Fig. 1.7). A guidance command generated from the IBVS control scheme was fed into an algorithm that maps the command into joint velocities of the manipulator and motion of the UAV as a whole. A 180° fisheye camera lens was used to prevent loss of the target image in the field of view, especially near the target where small movements of the UAV cause large changes in the image. While the IBVS technique was implemented onboard (with an Odroid XU SBC), the UAV position was estimated offboard (with Vicon). This work demonstrates the feasibility of using onboard computation to control both the UAV motion, and the motion of an attached robotic arm using an IBVS control scheme.

Other work has proposed IBVS and PBVS strategies for more complex scenarios such as collaborative aerial manipulation [86] (although this was only tested in numerical simulations), high-speed aerial grasping from above [87] (performed indoors, using IBVS in 2D,

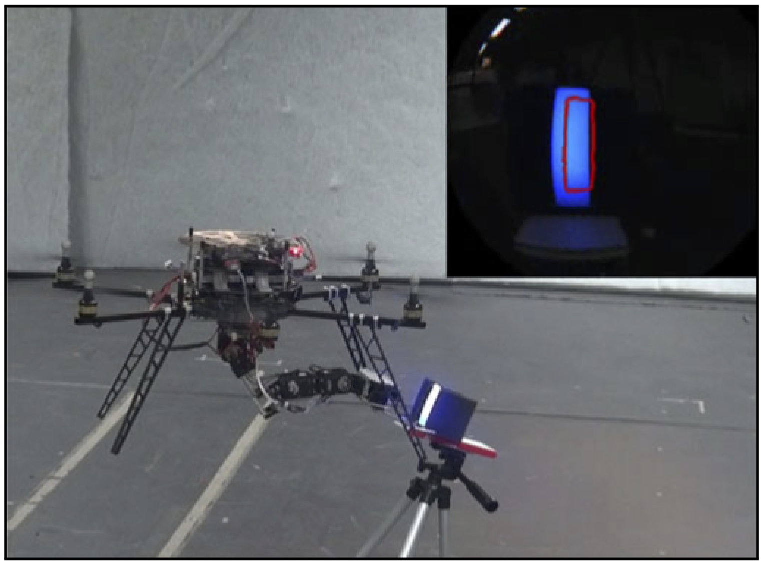


Figure 1.7. : Aerial manipulator grabbing an object with a blue LED attached [85].

and Vicon for lateral positioning and yaw control), and real-time tracking of user-selected objects [88] (using correlation filters for target tracking and motion capture for position control).

Finally, some works have considered more complex object detection scenarios in outdoor settings [39, 89, 90]. In [89], machine learning (running onboard) was used to classify and detect the pose of a variety of objects, however, the UAV did not perform any autonomous actions based on the detected target information, and the pose estimation errors were not reported. And in [90], an Intel Aero drone with downward- and forward-facing cameras used GPS to hover above a balcony, descend until a visual marker was detected on a wall, and then fly forward toward the marker while using the depth camera to avoid obstacles (although the UAV did not carry any payload or perform any manipulations in the flight test).

1.3.3 Summary of UAV Localization for Aerial Manipulation

In review, the research efforts described in section 1.3.1, demonstrate that various VS-LAM techniques can be used for onboard UAV localization, but come with a high computational cost, and additional work remains to be done to make them more robust for use in environments with few visual features, changing/poor lighting conditions, and rapid camera movements (as experienced on UAVs). Alternatively, several works have shown promising results using OF-based VO for lightweight, onboard UAV localization in both indoor and outdoor scenarios, with pose estimation performance at least on par with GPS, even in environments with few visual features (as long as there is at least some texture in the field of view). And, in a recent comparison of several monocular VO-based UAV localization methods implemented on SBCs, the state-of-the-art achieved in translational position estimation (measured as RMSE) was shown to be $\approx 10\ cm$, given sufficient computational resources.

As discussed in section 1.3.2, aerial manipulation efforts to this point have typically either assumed that the position and orientation of the object to be manipulated are known, or else they have added simple visual markers on or near the object for easy detection/tracking. For example, several researchers have considered tasks such as object transport using simple tracking methods with artificial markers (e.g. IR blob detection, cylinder detection, custom visual markers [91], etc.) in indoor environments (typically using external position estimation systems for UAV localization/position control). Other works have demonstrated that autonomous operations can be performed using both IBVS and PBVS techniques, even without a global map of obstacles/features in the environment. However, relatively few efforts have considered the complete autonomous aerial manipulation problem where both the UAV localization and visual target tracking are performed onboard in unstructured environments.

1.4 Contributions

In this work, the design of a new platform for aerial manipulation is presented: the Interacting-BoomCopter UAV (I-BC). With a simple tricopter frame, a single additional actuator for generating horizontal forces, and lightweight, modular end-effectors, the design of the I-BC aims to balance efficiency and functionality in performing aerial manipulation tasks. Onboard cameras, force and distance sensors and a powerful single board computer (SBC) enable the I-BC to perform autonomous aerial interactions with the environment. In addition, a featureless visual target tracking method is presented that allows the I-BC to operate in unstructured indoor and outdoor environments.

Considering the limitations, and building upon the strengths of prior work in the field, the main contributions of this work are:

- The design of a novel UAV platform for aerial manipulation
- The design of modular end-effectors with integrated force and distance sensing to enable both momentary and sustained contact with the environment
- A strategy for tracking the position and orientation of an aerial manipulation target without the use of artificial markers
- Realization of autonomous aerial manipulation capabilities in unstructured environments, using only onboard computation for both UAV localization and visual target tracking

The following chapters are organized as follows. In chapter 2, the design and prototyping of the I-BC is covered in detail. Then, chapters 3-4 consider two representative aerial manipulation task implementations, along with custom end-effector designs and high-level autonomous control strategies for the I-BC. Next, chapter 5 presents a strategy for localizing the I-BC relative to a manipulation target that enables the I-BC to perform aerial manipulation tasks autonomously in unstructured environments. Chapter 5 also presents experimental results from several indoor and outdoor flight tests that validate the proposed target localization strategy and demonstrate the I-BC's ability to perform autonomous aerial manipulation tasks in real-world scenarios. Finally, chapter 6 provides a summary of the contributions of this work, along with conclusions and recommendations for future work.

2. VEHICLE DESIGN & PROTOTYPING

The primary limiting factor in the productiveness of many aerial robot systems is their power source (typically Lithium-polymer (LiPo) batteries). From the time of takeoff to the time of landing, an aerial vehicle must constantly expend power to offset its own weight plus the weight of any payload, in addition to the power consumed by all of its onboard sensors, computers, and robotic actuators. As such, there is a tradeoff between improved performance (e.g. payload capacity) and overall efficiency (e.g. maximum flight time). Thus, every component of the I-BC was selected and designed to minimize the overall weight and power consumption of the vehicle, while still providing sufficient payload capacity and onboard sensing and computational power to perform the desired aerial manipulation tasks.

2.1 Airframe and Actuator Configuration

The I-BC is a multirotor UAV platform that employs three primary rotors for lift and general maneuvering [92] (see Fig. 2.1). The primary rotors are arranged in a Y-shaped configuration and are mounted in a horizontal plane. In order to reduce unbalanced reaction torques, the two front propellers rotate in opposite directions, and the tail rotor is mounted on a servo-actuated tilting platform that allows it to rotate about the tail arm. These three primary rotors and tail servo provide the I-BC with four independent degrees of freedom: X-Y-Z position, and heading (yaw angle).

To move forward, the thrust from the front rotors is decreased and the tail-rotor thrust is increased to induce a forward rotation (pitch). This rotation generates a net force in the forward direction causing the vehicle to translate. Similarly, to move right, the thrust from the right rotor is decreased and the thrust from the left rotor is increased to induce a rotation (roll) and thus a translation to the right. These roll and pitch angles are reversed to produce motion in the opposite directions, backward and left, respectively. Finally, the

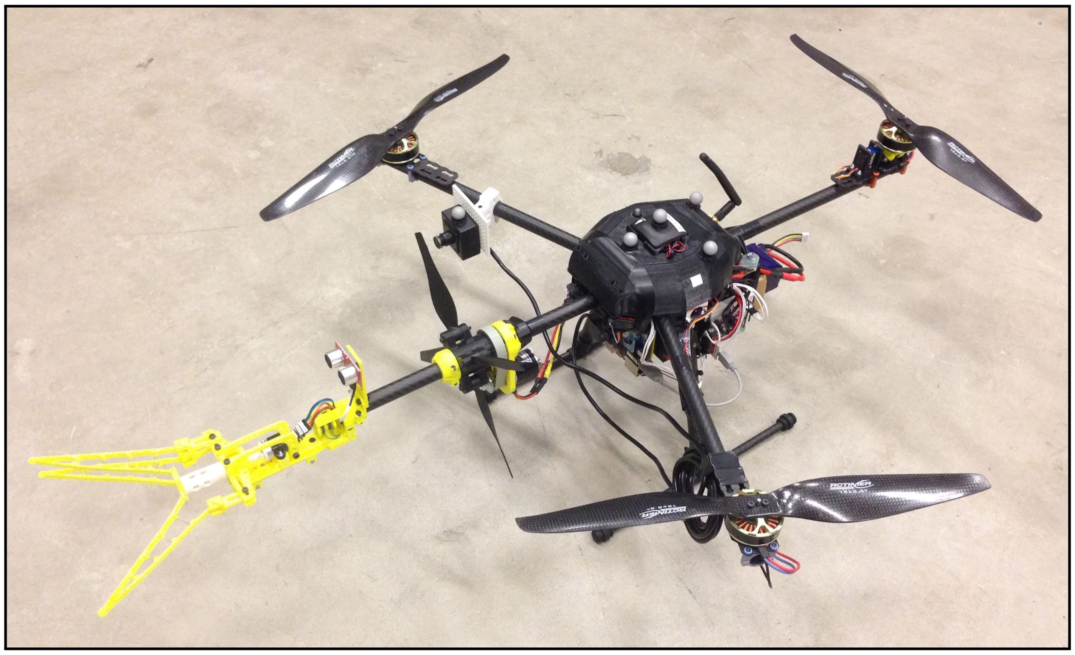


Figure 2.1. : Interacting-BoomCopter (I-BC) prototype.

vehicle orientation or heading angle (yaw) is adjusted by tilting the tail rotor. Tilting the tail rotor toward the right or left of the vehicle generates an unbalanced torque and causes the vehicle to rotate counter-clockwise or clockwise, respectively.

An additional propeller, mounted in the vertical plane on a boom extending out from the front of the I-BC, provides transverse thrust in the forward and reverse directions. Since the front-mounted propeller (boom-prop) rotates around the front boom, its thrust is applied very close to the I-BC's center of gravity (COG), and thus allows the I-BC to move forward and backward without pitching. This decoupling of forward and reverse motion from the vehicle's pitch angle is advantageous when performing aerial manipulation tasks that require the UAV to maintain a steady attitude relative to the manipulated object.

Several lightweight, modular end-effectors have been designed to enable the I-BC to perform specific aerial interaction tasks. The I-BC's end-effectors are attached at the end of its front arm, and have integrated force and distance sensors that allow the I-BC to interact autonomously with the environment. Because of their modular designs, these end-effectors can be switched out as needed to perform different manipulation tasks. Specific design details for these end-effectors will be provided in later chapters.

Table 2.1 lists the I-BC's major components. Several custom parts, including the boom-prop mechanism, tail servo mount, battery compartment, camera mount, and protective shell were 3D-printed on a Makerbot Replicator 2X using Acrylonitrile Butadiene Styrene (ABS) plastic with 75% infill.

Table 2.1. : Primary components of the I-BC.

Component	**Details**
Carbon-Fiber Frame	Tarot FY650 (modified to tricopter)
Main Motors	Turnigy Multistar 4108-480 Kv
Boom-prop Motor	NTM Prop Drive 28-36 1400 Kv (560 W)
Main Propellers	RCTimer 15x5.5" Carbon Fiber
Boom-prop Propellers	Gemfan 8x4.5 3D Propeller (modified)
Primary ESCs	Afro Slim 20A (SimonK Firmware)
Boom-prop ESC	Afro 30A (SimonK Firmware)
Tail Servo	Hex-tronik MG14 (Metal Gear)
Flight Controller	Pixhawk (with 3D acc/gyro/mag/baro)
Companion Computer	Odroid XU4 running Linux and ROS
GPS Module	3DR uBlox GPS with Compass
Force Sensors	Force-Sensing Resistor (Interlink Electronics 402)
Batteries	Turnigy nano-tech, 6000 mAh, 25-50C

2.2 Dynamic Model

A free-body diagram of the I-BC is shown in Fig. 2.2. The right-handed inertial frame (E-frame) is denoted by the x-y-z axes and a right-handed body frame (B-frame) is denoted by x_b, y_b, z_b axes. The positive x_b-axis points along the I-BC's front boom, the positive y_b-axis points to the I-BC's left and the positive z_b-axis is directed upwards. The roll (ϕ), pitch (θ), and yaw (ψ) angles are defined by a right handed rotation about the positive x, y, and z axes, respectively.

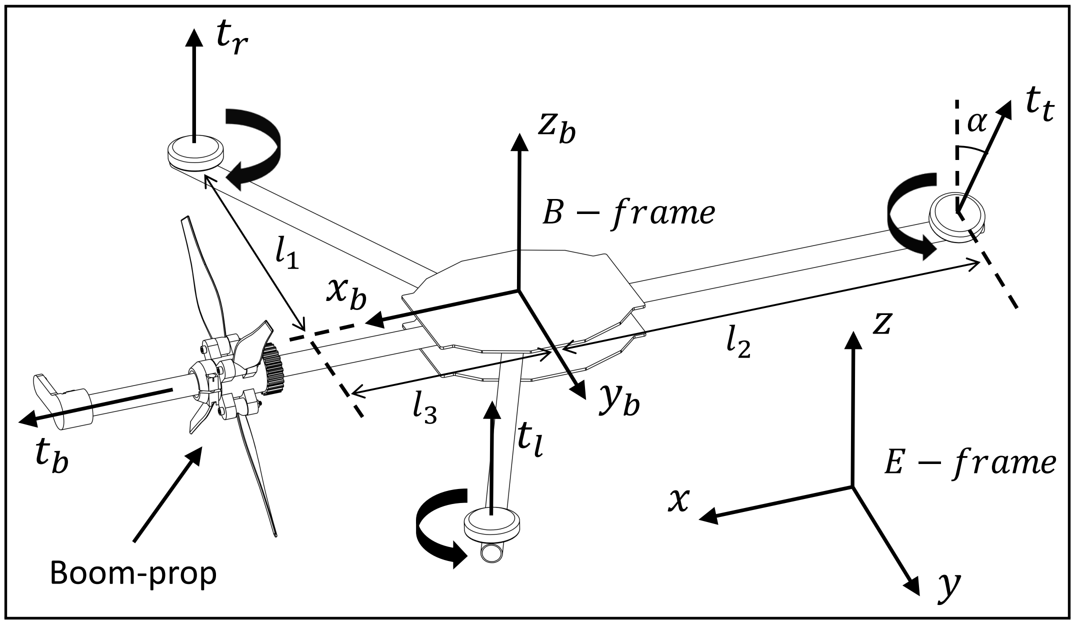


Figure 2.2. : I-BC free-body diagram.

The complete mathematical model of the system is given as:

$$\dot{\mathbf{X}} = \mathbf{V} \tag{2.1}$$

$$\dot{\mathbf{\Theta}} = \mathbf{W_{xyz}\Omega} \tag{2.2}$$

$$\dot{\mathbf{V}} = \frac{1}{m}\mathbf{R_{xyz}F_b} + \mathbf{G} \tag{2.3}$$

$$\dot{\mathbf{\Omega}} = \mathbf{I^{-1}}(\mathbf{T_b} - \mathbf{\Omega} \times \mathbf{I\Omega}) \tag{2.4}$$

where $\mathbf{X} = [x\ y\ z]^T$, $\mathbf{V} = [\dot{x}\ \dot{y}\ \dot{z}]^T$, and $\mathbf{\Theta} = [\phi\ \theta\ \psi]^T$ are the position, velocity, and roll-pitch-yaw angles in the E-frame, respectively. $\mathbf{\Omega} = [p\ q\ r]^T$ denotes the angular velocities about the x_b, y_b and z_b-axis in the B-frame. The $\mathbf{W_{xyz}}$ matrix is given as:

$$\begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & \frac{s_\phi}{c_\theta} & \frac{c_\phi}{c_\theta} \end{bmatrix}$$

m is the mass of the vehicle; $F_b = [F_{xb}\ 0\ F_{zb}]^T$ is the force vector along the x_b, y_b and z_b-axis and $G = [0\ 0\ \text{-}g]^T$ (g is the acceleration of gravity), both in the E-frame. $\mathbf{R_{xyz}}$ is the rotation matrix given as:

$$\begin{bmatrix} c_\psi c_\theta & -s_\psi c_\phi + c_\psi s_\theta s_\phi & s_\psi s_\phi + c_\psi s_\theta c_\phi \\ s_\psi c_\theta & c_\psi c_\phi + s_\psi s_\theta s_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

Note: s_θ, c_θ, t_θ are the abbreviated forms of $sin(\theta)$, $cos(\theta)$ and $tan(\theta)$ respectively in the above matrices.

I is the moment of inertia matrix: $I = [\ [I_{xx}\ 0\ 0]^T\ [0\ I_{yy}\ 0]^T\ [0\ 0\ I_{zz}]^T\ ]$ where I_{xx}, I_{yy} and I_{zz} are the moments of inertia about x_b, y_b and z_b axes, respectively. The torque about the x_b, y_b and z_b-axis in the B-frame is denoted as $T_b = [\tau_{xb}\ \tau_{yb}\ \tau_{zb}]$.

As shown in the Fig. 2.2, t_l, t_r, t_t and t_b represent the thrusts from the left, right, tail, and boom-prop rotors, respectively, and α is the tilt angle of the tail-rotor measured from the vertical. Therefore, the T_b vector can be represented as:

$$\begin{bmatrix} \tau_{xb} \\ \tau_{yb} \\ \tau_{zb} \end{bmatrix} = \begin{bmatrix} l_1(t_l - t_r) \\ l_2 t_t c\alpha - l_3(t_l + t_r) - \tau_t s\alpha \\ l_2 t_t s\alpha + \tau_t c\alpha - \tau_l + \tau_r \end{bmatrix} \tag{2.5}$$

where τ_l, τ_r and τ_t are reverse torques of the left, right and tail rotors, respectively. The force vector component $F_{zb} = t_l + t_r + t_t cos(\alpha)$.

With the decoupled translational motion enabled by the addition of the boom-prop, the I-BC can operate in two modes: tricopter mode and boom-prop mode. In tricopter mode, the I-BC moves like a conventional tricopter (tilting the airframe to move forward and backward) and thus, the F_{xb} component is zero. However, when the I-BC is operating in the boom-prop mode, the airframe remains level while the boom-prop is engaged to produce forward or reverse thrust, and $F_{xb} = t_b$. The boom-prop reaction torque τ_b will be added to (or subtracted from) the τ_{xb} when the boom-prop is engaged in a clockwise (or counter-clockwise) sense.

2.3 Prototype Design and Testing

An initial BoomCopter (BC) prototype (I-BC without an end-effector) was built to characterize the general flight performance of the vehicle, and to determine the relative efficiency of using a front-mounted propeller for forward thrust rather than the traditional method of pitching the airframe forward. The BC airframe is based on a 650-class carbon-fiber quadrotor frame (the 650-class designation indicates the nominal distance from motor to motor is 650 mm). The frame was modified so that the three primary rotor arms are oriented in a Y-shaped configuration, and a fourth arm extends out the front of the vehicle. In this modified configuration, the total width and length of the airframe are 60 cm and 70 cm, respectively. To provide forward thrust, a 650 Kv motor (T-Motor: MT3506-25) with a 12 in diameter propeller (APC 12x3.8P) was mounted at the end of the front arm. The BC prototype and the I-BC both use a Pixhawk [93] flight control unit (FCU) to stabilize their attitude (driving the roll and pitch angles to zero with PID control) and record critical parameters to a microSD card during flight (including data from the GPS, IMU and power system monitoring sensors).

2.3.1 Comparison Between Tricopter and Boom-prop Mode

In the hobby community [94], the boom-prop was used on a tricopter UAV as a way to achieve high-speeds during forward flight without pitching forward, but no efficiency comparison was made between these flights and traditional forward flights. In order to compare the forward-flight efficiency of the BC's two flight modes, the forward acceleration achieved while flying in both tricopter and boom-prop mode were simulated based on the dynamic model described above. For tricopter mode, we calculated the accelerations achieved at various forward pitch angles, and for boom-prop mode, we calculated the acceleration achieved with the boom-prop at full throttle. A comparison between the simulated accelerations in the two modes is shown in Fig. 2.3, with tricopter mode accelerations in blue and the boom-prop mode acceleration in red.

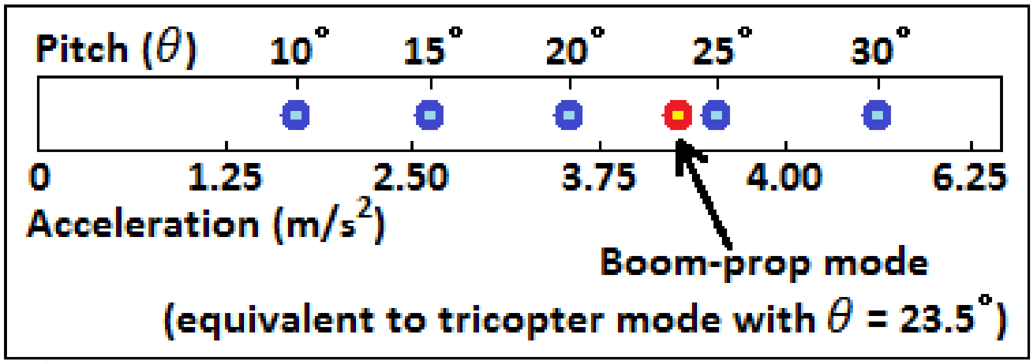


Figure 2.3. : Simulated BC accelerations in tricopter mode at different pitch angles (θ), and boom-prop mode at full throttle.

These simulations show that the tricopter mode acceleration exceeds the boom-prop mode (0° pitch) acceleration for forward pitch angles (θ) larger than 23.5°. This is expected since the tricopter mode uses three larger propellers for forward thrust while the boom-prop mode uses only one (typically smaller) propeller for forward thrust.

Several outdoor flight tests were performed in mild weather conditions (with minimal wind disturbance) to validate the results of the above simulations. Each test consisted of flying the BC straight forward for approximately 75 m at a fixed altitude in both the boom-prop and the triopter mode. A switch on the radio controller was used to activate the boom-prop mode by ramping up the boom-prop motor to full throttle in a 2-second period, and then ramping the throttle back down to zero when the switch was deactivated.

When flying in tricopter mode, the BC pitched forward $30^\circ - 40^\circ$ and reached an average top speed of $\approx 15.5\ m/s$ (as indicated by the GPS module), while using $\approx 285\ W$ of power. In boom-prop mode, the BC achieved an average top speed of $\approx 9\ m/s$ while using $\approx 415\ W$ of power (an increase of $\approx 45\%$ power usage). These tests confirm that flying the BC in boom-prop mode results in slower top-speeds and higher energy consumption compared to tricopter mode.

2.3.2 Boom-prop Hardware Design

An exploded view of the I-BC's boom-prop mechanism is shown in Fig. 2.4 below. The boom-prop design found in the hobby community [94] consists of several 3D-printed components (made with ABS plastic), and a few commercial off-the-shelf components: motor, pulley, timing belt and bearings. However, the boom-prop's thrust capacity is primarily affected by: 1) the number of propeller blades (restricted by the blade spinner), 2) the shape and diameter of the propeller blades, 3) the motor, and 4) the gear ratio between the motor and the blade spinner (established by the size of the pulley). Thus, we focused on optimizing these four components to maximize the thrust generated by the boom-prop. Following is a discussion of design considerations for each of the primary boom-prop components, and experimental results demonstrating the performance of the boom-prop.

Static Thrust Test Setup

In order to compare the thrust output of various boom-prop designs, several bench tests were performed in which the input voltage and current, propeller rpm, and thrust output of the boom-prop were measured for each configuration. The propeller rpm was measured with a digital photo tachometer and a $1\ cm \times 1\ cm$ reflective IR patch attached to the propeller. An RC Electronics Watt's Up meter was connected in series with a Tenma 72-630 DC power supply to measure the input voltage and current to the boom-prop motor during each test, and the power consumption was calculated as $P = I * V$. The thrust generated by the boom-prop was measured by mounting the boom-prop mechanism to a static thrust test rig.

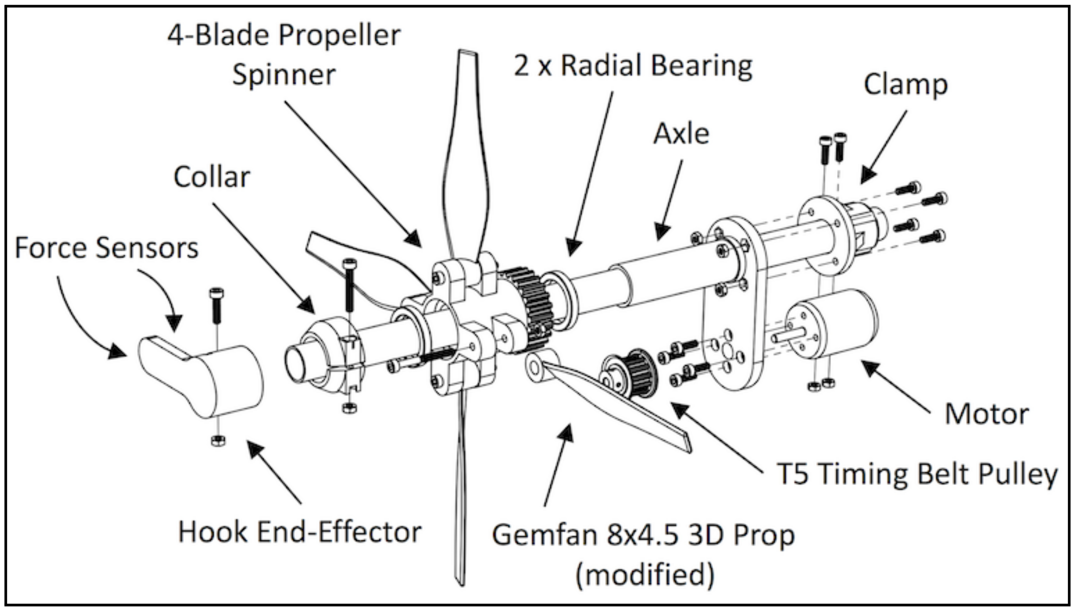


Figure 2.4. : Exploded view of the I-BC boom-prop and end-effector assembly.

The test rig uses an L-shaped one-to-one lever arm to transmit the horizontal boom-prop thrust to a weight scale where the force is observed on an LCD screen and manually recorded. The boom-prop was driven by an Afro 30A electronic speed controller (ESC), which received pulse-width modulated (PWM) throttle commands in increments of 10% (e.g. 10%, 20%, etc.) from an Arduino Uno (ATMega328P) microcontroller with a UART serial connection to a laptop.

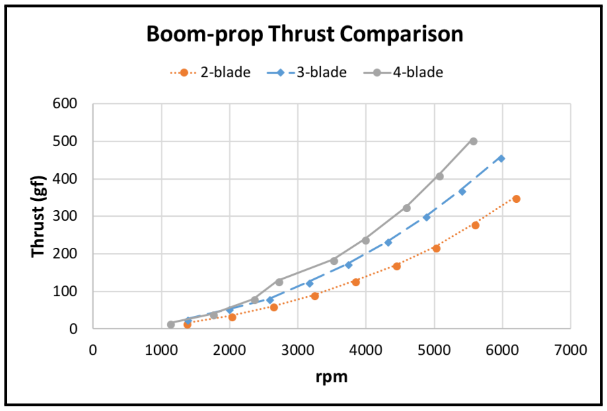
Shape and Diameter of Propeller Blades

Most propeller blade profiles are optimized to generate thrust when rotated in one direction only (indicated with CCW for counter-clockwise or CW for clockwise), and the output thrust is significantly reduced when the blade is spun in the reverse direction. This is not desirable for the I-BC where equal forward and reverse thrust capability is desired to maximize the flexibility of the I-BC in performing aerial manipulation tasks. So-called *3D propellers* with symmetric blade profiles do exist for use with reversible motors on some UAV platforms. These 3D propellers generate the same thrust when rotated in either direction, but this thrust is typically lower than the thrust produced by a standard propeller of the same diameter. However, since the thrust of a 3D propeller is still greater than the thrust produced by a standard propeller of the same diameter spinning in reverse of its indicated direction (e.g. a CCW propeller spinning CW), 3D propeller blades were used for the I-BC's boom-prop so that the forward and reverse thrusts are the same. At the time of writing, folding 3D propeller blades are not in production, so custom folding 3D propeller blades were manufactured in our lab by removing half of a standard 3D propeller blade and sanding down the center hub of the blade until it was smooth and round so that it could be attached to the blade spinner as shown in Fig. 2.4. The diameter of the propeller blade also has a direct effect on the thrust generated, with larger diameter propellers producing more thrust per rpm than smaller diameter propellers within the scale of propellers that will fit on the I-BC (assuming a similar propeller width ratio). Thus, the largest available 3D propellers (8 *in* diameter) were used for the boom-prop.

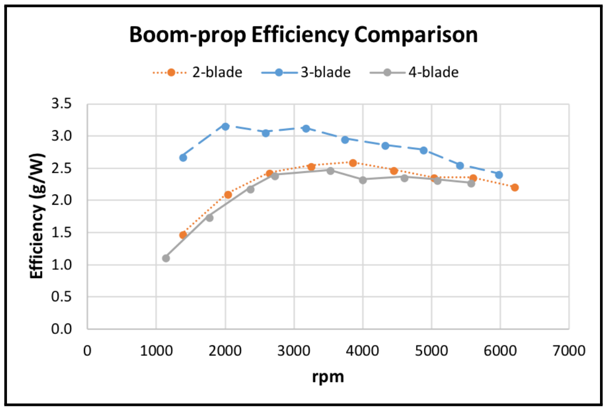
Number of Propeller Blades

Static thrust tests were performed to compare the thrusts generated by the boom-prop with different numbers of propeller blades (using variants of the blade spinner that can accommodate two, three, or four folding propeller blades) while keeping the propeller blades, motor, pulley, and power source constant for each test. Fig. 2.5 shows comparisons of the boom-prop thrust generated with two, three and four propeller blades. As expected, the total thrust increases with an increased number of propeller blades (see Fig. 2.5a), but the increase is less pronounced as the total number of blades increases.

Fig. 2.5b shows a substantial increase ($\approx 25\%$) in efficiency for the three-blade boom-prop configuration compared to the two- and four-blade configurations, where efficiency is defined as the total thrust generated (measured in gram-force) divided by the input electrical power. However, since the operating time of the boom-prop (limited to the time spent performing aerial manipulation tasks) will be a relatively small portion of the total flight time, the benefits of increased efficiency are outweighed by the benefits of a larger maximum thrust produced by the four-blade configuration.



(a) Boom-prop thrust vs. rpm.



(b) Boom-prop efficiency vs. rpm.

Figure 2.5. : A comparison of boom-prop thrusts generated with different numbers of propeller blades.

Motor Velocity Constant and Gear Ratio

With a four-blade 8"-diameter 3D propeller configuration selected, the only remaining variables to optimize in the boom-prop design were the motor and gear ratio (pulley). As can be seen in Fig. 2.5, the boom-prop output thrust increases with rotation speed for a given propeller diameter. Thus, further static thrust tests focused on identifying the ideal combination of motor and pulley size to maximize the boom-prop rotation speed.

The motor velocity constant (K_v) is frequently used when determining the suitability of a particular brushless DC motor to drive a propeller of a certain size and pitch. K_v relates the no-load rotational speed of a brushless DC motor to the voltage applied to its coils as follows: $K_v = \frac{\omega_{no-load}}{V_{applied}}$. This value is inversely proportional to the torque constant of a motor (K_t), which relates the motor torque to the armature current as follows: $K_t = \frac{\tau}{i_a}$. Generally speaking, given two motors and a specified voltage, the motor with a higher K_v value will rotate faster, but will produce lower torque. Thus, motors with higher K_v values are often used with smaller propellers, which operate at higher speeds and require lower torques to accelerate, and motors with lower K_v values are used with larger propellers that require higher torques. Other motor parameters such as the physical size of the motor and the number of poles and windings within the motor itself can also affect the performance of a motor-propeller system, which makes the motor selection process more complicated.

Considering these various factors, we first performed several static thrust tests with the four-blade boom-prop configuration to determine the effect of changing the boom-prop gear ratio. In these tests, it was found that decreasing the gear ratio between the motor and the blade spinner (by increasing the size of the pulley) from 2:1 down to 1.5:1, increased the maximum rpm achieved and thus the maximum thrust achieved, but decreasing the gear ratio from 1.5:1 down to 1.2:1 had no noticeable effect. Also, the lower gear ratios had no noticeable effect on the efficiency of the boom-prop. Using an NTM2836, 1000 K_v motor, the maximum thrust achieved was $\approx 670\ gf$.

Finally, in an attempt to further increase the boom-prop rpm (and thus the static thrust), a motor with a slightly higher K_v value and power rating was selected. Thrust tests with this motor (NTM2836, 1400 K_v), a 2:1 gear ratio, and the four-blade propeller configuration gave the maximum overall boom-prop thrust: 878 gf. As will be shown in later sections, this force is sufficient for performing a range of aerial manipulation tasks, but it could be increased in the future if larger 3D propeller blades become available, or by using larger diameter standard propellers (if asymmetric forward/reverse force is deemed acceptable for a certain manipulation task or tasks).

2.4 Flight Performance

The BC platform specifications were characterized by hovering and flying outdoors in straight-line forward trajectories in both boom-prop and tricopter mode with various payloads attached underneath the vehicle. The battery compartment is designed to accommodate either one battery or two batteries connected in parallel. Using a second battery significantly increases the total flight time, but adds a substantial amount of weight, thus decreasing the payload capacity. Table 2.2 provides several key specifications for the BC (including variations depending on the number of batteries used). Battery life was determined based on the safe practice of using only $\approx 80\%$ of the battery's capacity, and the payload (excluding the weight of the batteries) was calculated such that the vehicle is able to hover with the throttle at 75%, allowing some headroom for modest maneuvering. The total flight time calculation was also based on the assumption that aerial interaction tasks use the boom-prop at full throttle for ≈ 5 seconds at a time and occur intermittently (about once per minute).

Table 2.2. : BoomCopter vehicle specifications.

Parameter	**Value (1 battery)**	**Value (2 batteries)**
Mass	2.31 kg	2.92 kg
Max Payload	1.86 kg	1.20 kg
Max Speed	17 m/s	Not Tested
Flight Time	13 $min.$	21.4 $min.$
Flight Range	8.8+ km	14.4+ km
Power to Hover	230 W	335 W

Note: Overall Size = 60 $cm \times 70$ $cm \times 29$ cm

2.4.1 Indoor Flight Setup

As mentioned above, the I-BC's position and attitude are controlled by a Pixhawk FCU. When flying outdoors, the Pixhawk uses a combination of GPS and barometer data to autonomously maintain the vehicle's altitude and horizontal position within the resolution of the GPS. However, when flying indoors, GPS readings are unavailable and the barometer-based altitude measurements can be unreliable due to sudden air pressure changes resulting from opening or closing a door, airflow from HVAC fans, etc. As such, the I-BC has a downward-facing laser range finder (LIDAR-Lite v3) which enables autonomous altitude control indoors. The addition of the downward-facing laser range finder allows the I-BC to enter Altitude Hold mode and maintain a steady altitude during teleoperated indoor flights.

Fig. 2.6 shows the I-BC's performance in Altitude Hold mode. During the 60 second window shown, the error in altitude was typically less than ±4 *cm* and reached up to 6 *cm* only a few times.

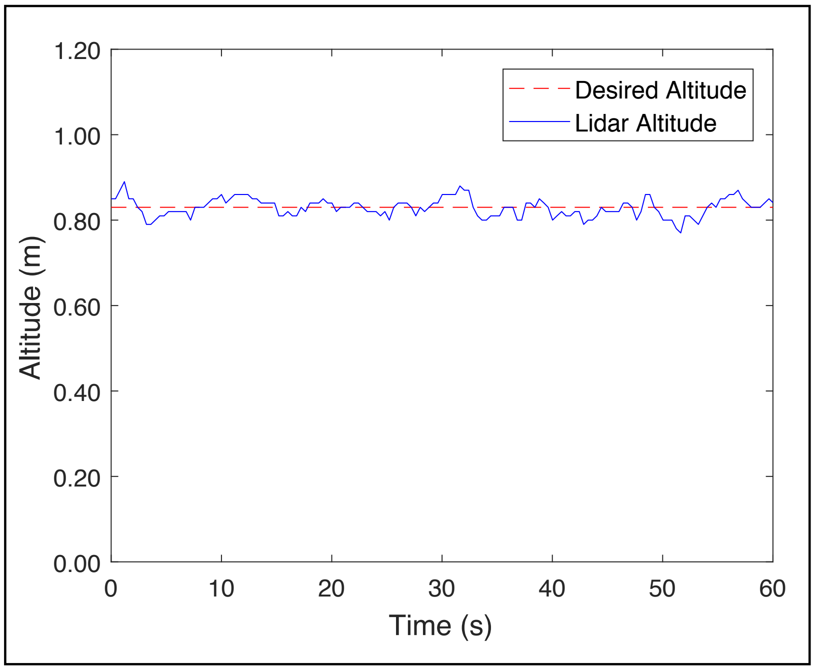


Figure 2.6. : I-BC altitude control performance in altitude hold flight mode.

The base flight software running on the Pixhawk for the outdoor flights was the open-source ArduPilot flight stack. However, when performing indoor flights, the PX4 flight stack was used since it provides better support for position estimation with motion capture systems.

3. REMOTE SENSOR MOUNTING TASK

The ability to place a sensor package (i.e. a wireless camera, vibration sensor, etc.) in remote or hard-to-reach locations with a UAV is desirable in many situations, including infrastructure inspection and maintenance, structural health monitoring, air quality assessment, construction management, surveillance operations, etc. As a representative task for these types of scenarios, the remote mounting of a wireless camera sensor package on a vertical surface is considered here (see [95, 96]). First, the sensor package design will be introduced, followed by the design of a custom end-effector that can transport and deploy a small sensor package. Then, real-time image processing techniques for identifying the target mounting location will be discussed, and finally, a high-level control scheme that enables the I-BC to perform remote sensor mounting tasks autonomously will be presented along with experimental results.

3.1 Wireless Sensor Package Design

For the remote sensor mounting task, a rectangular box housing was designed to carry a wireless camera (with antenna) and its 9V power supply (see Fig. 3.1), with the dimensions shown in Fig. 3.1a. The complete sensor package has a mass of 100 g with the contents shown in Fig. 3.1b, which is well within the payload capacity of the I-BC, and is low enough that the I-BC's COG can be maintained near the center of the vehicle by shifting the battery toward the back of the vehicle.

This sensor package form factor ($3.1\ cm \times 5.1\ cm \times 5.8\ cm$, mass $\leq 100\ g$) can feasibly house a variety of different sensors including vibration, temperature, or air quality sensors, etc. along with a wireless communication module and a power source, thus, the methods presented here can be extended to a broad range of applications.

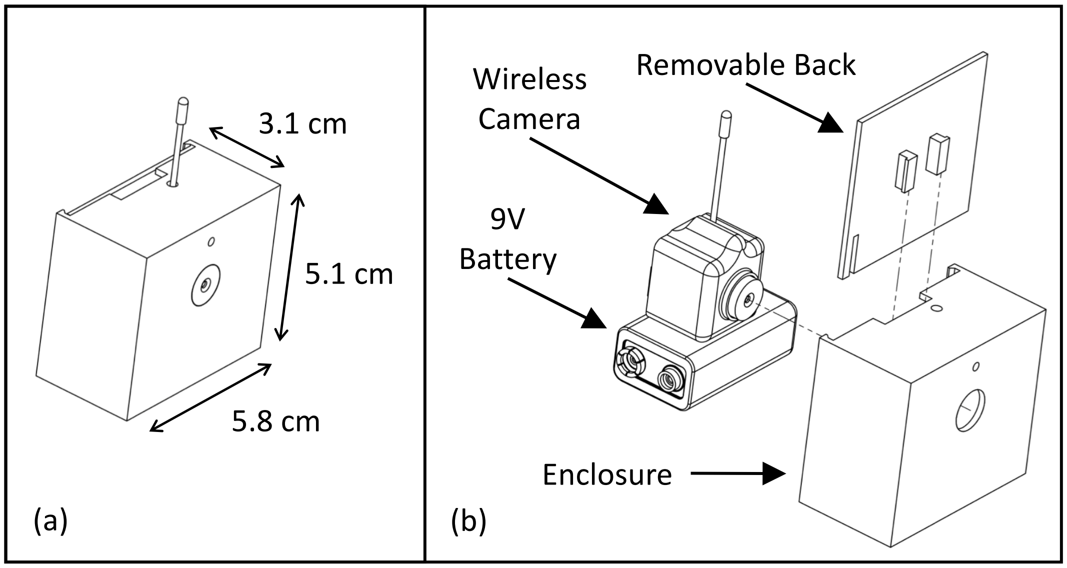
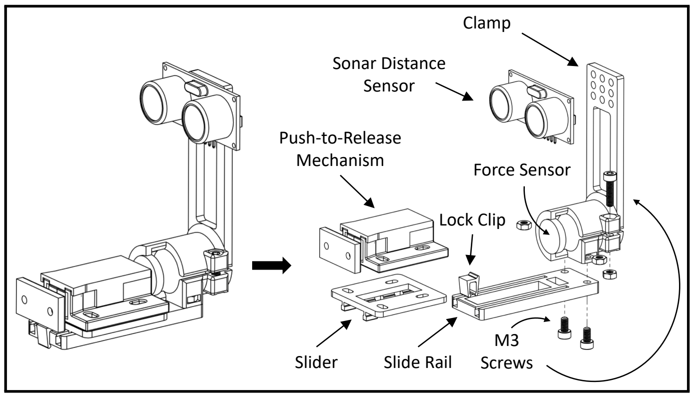


Figure 3.1. : (a) Dimensions of 3D-printed wireless camera sensor package; (b) exploded view of sensor package components.

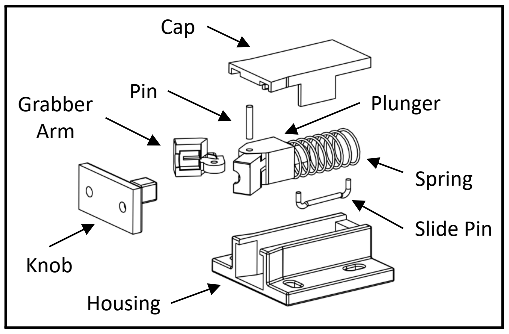
The back of the sensor package is flat so that it can be attached to a surface with two-sided tape. This method of attachment requires the mounting surface to be relatively clean and smooth, but will work on a wide variety of common structural surfaces such as metal beams, painted drywall, glass, etc., and does not require the use of any additional tools or hardware (e.g. drills, rivets, etc.). In our experiments, we determined that 3M RP45 VHB (very high bond) two-sided foam tape is a robust adhesive for this application since it provides strong adhesion in both indoor and outdoor settings and is compatible with a wide variety of mounting surfaces.

3.2 End-Effector Design

Physically mounting a sensor package on a vertical surface involves two primary operations: first, the sensor package is attached to the surface, and second, the package is released from the I-BC's end-effector so the I-BC can retreat. The end-effector shown in Fig. 3.2 was designed so that both of these operations can be accomplished through the use of the boom-prop without the need for any additional actuators. The boom-prop generates a horizontal



(a) Sensor-mounting end-effector assembly and exploded view.



(b) Push-to-release mechanism exploded view.

Figure 3.2. : Assembly and exploded views of the I-BC's push-to-release end-effector used for remote sensor mounting.

force that presses the sensor package against the mounting surface until it is firmly attached. After the sensor package has been attached to the surface, a passive push-to-release mechanism (PTRM) is used to release the sensor (see Fig. 3.2b). The clamp, slide, rail, sled and lock clip were all 3D printed with ABS plastic. The assembled end-effector weighs only 50 g and is completely passive, which reduces the overall weight and conserves battery power to extend flight times. The sonar sensor is a low-cost HC-SR04 module that provides accurate distance measurements between 2 cm and 2 m with a resolution of 3 mm.

3.2.1 Push-to-release Mechanism (PTRM)

The PTRM is based on a grab latch design that is typically used on cabinet doors, and consists of three main components: the knob, the grabber arms, and the housing (see Fig. 3.2b). The PTRM toggles between a locked state and a released state when the knob is pressed inward.

The knob consists of a base that can be attached to a sensor package with screws or an adhesive, and a small cube extended a short distance from the base by a cylindrical cantilevered arm. The plunger slides in and out of a cavity in the housing, and has two grabber arms that rotate about a common pin to open and close around the cube portion of the knob. The grabber arms are driven to a normally-open state by a torsional spring when the grabber is extended to the front of the housing cavity, however, the width of the cavity decreases along the length of the housing so that the grabber arms are forced shut as the plunger slides into the housing. If the knob is pressed against the plunger while it slides into the housing, the knob will be captured by the grabber arms and cannot be removed until the plunger slides back out of the housing. For our experiments, we purchased a PTRM and replaced the original grabber arms and knob with our own 3D printed versions that use a cube-shaped knob instead of a spherical knob. The cube shape on the knob prevents it from rotating while enclosed in the grabber arms. This allows for the orientation of the sensor package to be controlled more strictly.

The two states of the PTRM are governed by the slide pin (a hook-shaped, cantilevered pin, which is held in the center of the housing cavity by a circular, planar spring). The planar spring provides an upward restoring force proportional to the downward deflection of the slide pin, and provides left/right restoring forces proportional to any right/left deflection of the pin. Thus, the end of the slide pin is always driven back to the center point of the housing cavity. The end of the pin (which is curved upward) rests inside a 3D groove that is embedded in the bottom of the plunger. This groove is designed so that the slide pin travels between two resting positions corresponding to the open and locked states.

The force required to open the PTRM is governed primarily by the spring attached to the plunger. This force was measured to be $\approx\ 575\ gf$, which requires the boom-prop of the I-BC to be designed to produce a minimum of $\approx\ 600\ gf$ of thrust to release the sensor package.

3.2.2 Force & Forward Distance Sensor Management

In order to measure the forces applied by the end-effector, an FSR was placed in series with the PTRM. This was accomplished by mounting the PTRM on a slider which glides with minimal friction along the slide rail (see Fig. 3.2a) until it comes into contact with the

FSR. Thus, all of the force transmitted from the I-BC's front arm to the sensor package is registered by the FSR.

A sonar distance sensor was also attached to the end-effector to measure the forward distance to a vertical surface (see Fig. 3.2a). The sensor is raised sufficiently high above the PTRM to prevent false readings from the sensor package attached to the PTRM.

Both the FSR and the sonar distance sensor are connected to an Arduino Pro Mini microcontroller (16 MHz ATmega 328P), referred to as the I-BC's Device Manager. The Device Manager converts the force and distance sensor values into standard data types and formats them into a binary packet for low-latency streaming over a UART serial connection. The Robot Operating System (ROS) framework [97] was used to develop a node running on the I-BC's onboard computer (Odroid XU4) that connects to the Device Manager data stream and makes the sensor data available to the I-BC controller (described in section 3.4 below) through standard ROS topics.

3.3 Manual Flight Tests

A table turned on end (0.61 m wide $\times$ 1.68 m tall) was used as a rigid vertical mounting surface, and the desired sensor mount location was marked with a 16.5 cm concentric square target, with its center at a height of 1.2 m (see Fig. 3.3).

After the stability of the I-BC's altitude controller was confirmed through multiple test flights, the wireless camera sensor package was attached to the end-effector and several manual sensor-mounting tests were performed. During normal flight, the Pixhawk FCU on the I-BC operates in one of several flight modes (e.g. Stabilize attitude, Altitude Hold, Position Hold, etc.). The I-BC's remote controller (RC) is programmed to allow the pilot to use physical switches to change the FCU's flight mode and turn the boom-prop on or off as desired.

For each flight test, the I-BC was flown manually to the desired altitude, switched into Altitude Hold mode, and then guided by the pilot to a position $\approx$ 1 m away from the target location on the wall. At this point, the boom-prop was engaged with 12% throttle by the

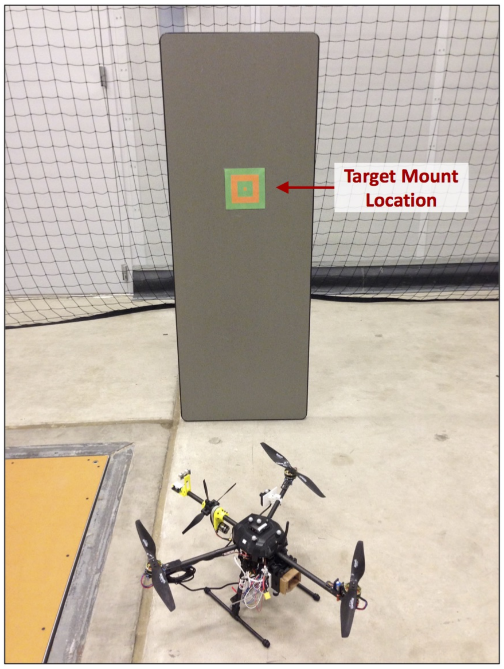


Figure 3.3. : Experimental setup for manual sensor-mounting flight tests.

pilot, which caused the I-BC to approach the wall at a slow speed. Once the sensor came into contact with the wall and was released from the end-effector's PTRM, the pilot disengaged the boom-prop, flew the vehicle back to its launch point, and landed it on the ground.

Fig. 3.4 shows snapshots from three stages of the sensor mounting operation that resulted in the minimum placement error. For reference, images from the onboard camera video feed corresponding to each stage are included to the right of each snapshot.

Fig. 3.5 shows the output of the end-effector sensors during a successful sensor mounting operation. The slope of the decreasing sonar measurements (see Fig. 3.5a) represents the I-BC's forward velocity during the approach. This remained steady at about 3.3 cm/s until impact, at which point the vehicle recoiled backward from the impact. It was then was pitched backward (after the boom-prop was disengaged) to complete the landing process. This behavior is displayed in Fig. 3.5a with an initially shallow slope in the sonar data after the impact with the wall, followed by an increasing slope as the vehicle is pitched away from the surface.

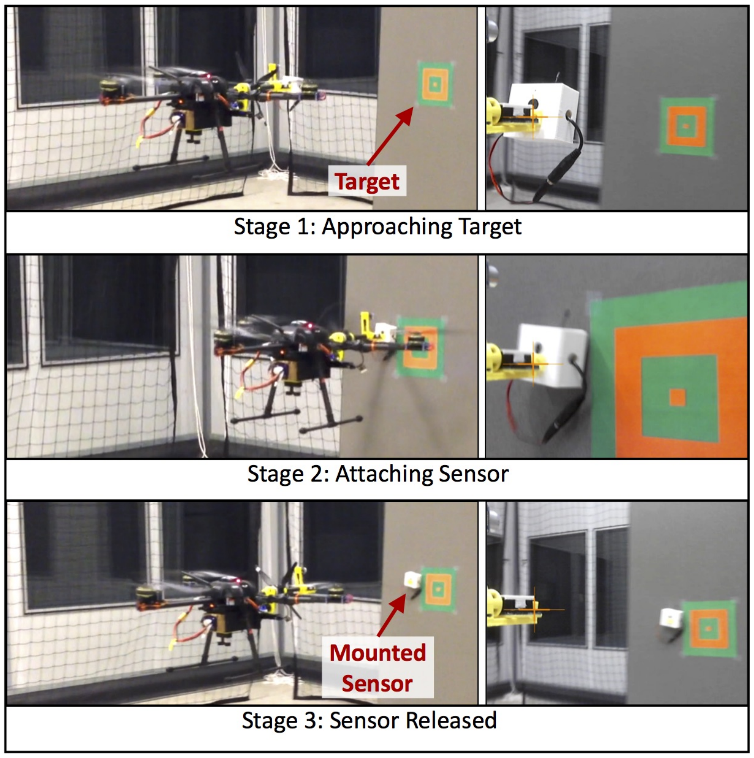
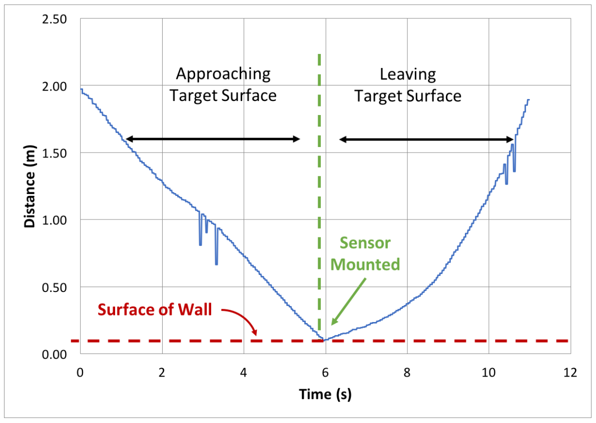
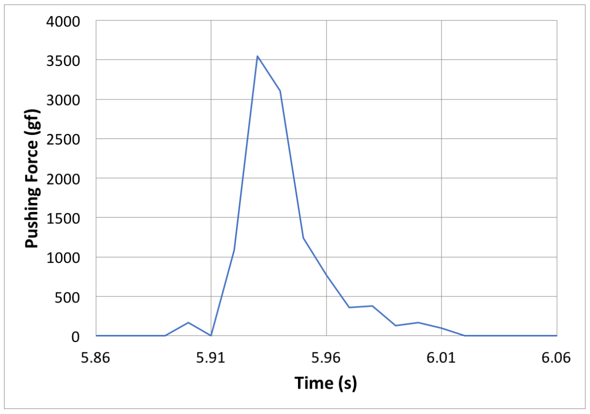


Figure 3.4. : Successful I-BC sensor mounting operation. Right column: view from onboard camera.



(a) Forward distance reported by sonar sensor.



(b) Force measured by end-effector during impact.

Figure 3.5. : End-effector sensor readings during a manual sensor mounting operation.

The impulse generated during the collision can be seen in Fig. 3.5b. The momentary peak in force occurred over a timespan of $\approx 0.1\ s$ and was higher than the force required to toggle the PTRM. Thus, the sensor was released from the end-effector immediately upon impact. This made it unnecessary to increase the boom-prop throttle to its maximum value, but other sensor delivery mechanisms, or a PTRM with a higher releasing force may require the boom-prop to engage at a higher throttle after impact.

The sensor placement error was measured for five manual sensor mounting trials where the sensor was successfully attached to the target wall. For these trials, the maximum, minimum, and average placement errors were 19 cm, 12 cm, and 16 cm, respectively. As expected, the accuracy and repeatability of this manual sensor placement are highly dependent on the pilot's skill level and the amount of attention that the pilot can safely allocate to the task while maintaining stable flight and a sufficient awareness of the vehicle's surroundings. Thus, it is desirable to automate the process with feedback from the I-BC's sensors to reduce the sensor placement error and reduce the stress on the pilot.

3.4 Autonomous Control Strategy

To increase the safety, accuracy and repeatability of the sensor mounting operation, a software package was developed to enable the I-BC to perform the sensor mounting task autonomously. The software is built in the ROS framework and uses feedback from the I-BC's onboard sensors, including end-effector pushing force, distance to the wall (sonar), and vehicle pose.

3.4.1 Extended Finite State Machine

The autonomous sensor mounting task consists of several operations performed sequentially, and is ideally suited for implementation in software as an extended finite state machine (EFSM). Each operation to be performed corresponds to a state in the EFSM, and state transitions are triggered by various events (e.g. when the I-BC arrives at a desired position). Fig. 3.6 shows the state-transition diagram for the sensor mounting task. Note that, for safety purposes, the pilot can enable or disable the autonomous controller at any time by toggling a switch on the RC, and fail-safes triggered by logical error conditions or sensor failures also return control to the pilot immediately.

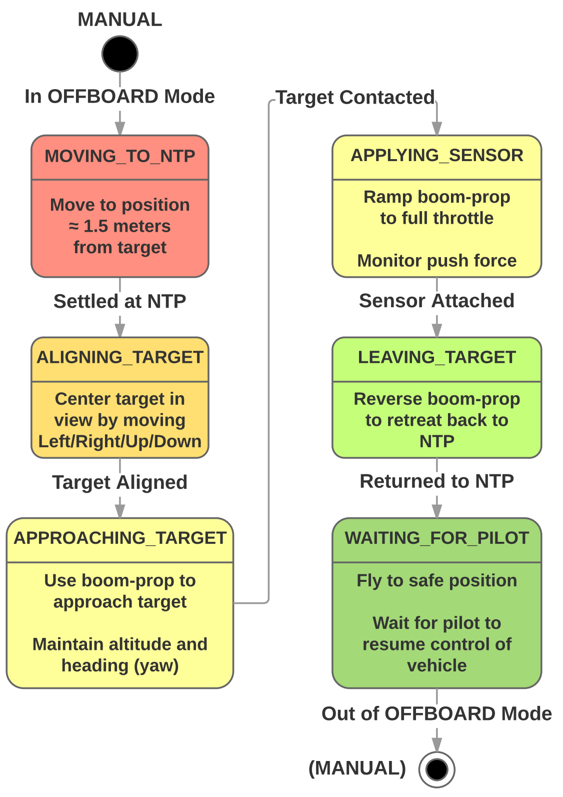

MANUAL
In OFFBOARD Mode
MOVING_TO_NTP
Move to position ≈ 1.5 meters from target
Settled at NTP
ALIGNING_TARGET
Center target in view by moving Left/Right/Up/Down
Target Aligned
APPROACHING_TARGET
Use boom-prop to approach target
Maintain altitude and heading (yaw)
Target Contacted
APPLYING_SENSOR
Ramp boom-prop to full throttle
Monitor push force
Sensor Attached
LEAVING_TARGET
Reverse boom-prop to retreat back to NTP
Returned to NTP
WAITING_FOR_PILOT
Fly to safe position
Wait for pilot to resume control of vehicle
Out of OFFBOARD Mode
(MANUAL)

Figure 3.6. : State-transition diagram for autonomous sensor mounting task [92].

3.4.2 Moving to Near-Target Position

It is assumed that the precise location where the sensor will be mounted, referred to as the target position (TP), is unknown prior to takeoff. Thus, the initial state (shown in red in Fig. 3.6) brings the vehicle to a near-target position (NTP) that is within 1-2 meters of the desired TP. Experimentally, the NTP is defined with a workspace coordinate in a Vicon motion capture volume used for testing. In application, the NTP could be specified by a GPS coordinate near the TP, or the pilot could manually fly to this position based on direct line-of-sight operation or with feedback from the onboard camera video stream. The NTP also specifies a heading angle which ensures that the vehicle's forward axis (body X axis) is perpendicular to the mounting surface. As the I-BC moves to its commanded position and heading setpoint, the EFSM compares its current estimated position with the setpoint, and triggers an event after the vehicle stays within a small envelope ($\leq 10\ cm$) in each direction of the setpoint for a predetermined amount of time. Once at the NTP, the I-BC adjusts its perpendicular distance to the wall (based on the forward-facing sonar readings) to ensure an optimal field of view for target tracking with the camera.

3.4.3 Aligning Vehicle with Target

With the vehicle stationed at a position near the mounting surface and within view of the TP, the EFSM transitions to the `ALIGNING_TARGET` state (shown in orange in Fig. 3.6). In this state, the TP is estimated by fusing the object tracking data from the camera with the sonar distance readings and the vehicle pose estimate. The error due to time delay between the camera image capture and the most recent pose estimate is minimal due to the high update rate of the pose estimate (100 Hz).

The object tracking algorithm could track either a predefined visual target, or a target specified by the user clicking in the live video feed from the I-BC's camera. More details on this algorithm are provided in section 3.5. Once an estimate of the TP is available, the I-BC's position setpoint is adjusted so that the end-effector is aligned horizontally and vertically with the TP. This position setpoint is referred to as the Webcam-Tuned NTP.

3.4.4 Mounting Sensor and Retreating

Once the vehicle has arrived at the Webcam-Tuned NTP, it is positioned directly in front of the desired mounting location and the EFSM transitions into the `APPROACHING_TARGET` state. The desired approach trajectory is computed by forming a 3D line between the Webcam-Tuned NTP and the TP. The boom-prop is then ramped up to $\approx 20\%$ throttle to move the vehicle forward along the desired trajectory. During the approach, the vehicle's position setpoint is continuously updated by projecting the current vehicle position onto the desired approach trajectory. This allows the position controller to correct horizontal and vertical deviations from external disturbances (e.g. wind) while simultaneously preventing it from commanding non-zero pitch angles, since the forward motion is generated by the boom-prop.

The force sensor in series with the sensor package is used to determine when the package comes into contact with the mounting surface. When the detected pushing force exceeds a predetermined threshold, a collision event is triggered and the EFSM transitions to the `APPLYING_SENSOR` state. At this point, the boom-prop is ramped up to full throttle while the force sensor data is monitored to ensure that the sensor package is pressed firmly in place and that the PTRM toggles to release the sensor package. After the sensor package is in place, the EFSM transitions to the `LEAVING_TARGET` state and the boom-prop throttle is reversed to $\approx -20\%$ to move the I-BC away from the wall and back to the Webcam-Tuned NTP (the initial position of the `APPROACHING_TARGET` state). At this point, the EFSM transitions to its final state, and the I-BC holds its position until the pilot resumes manual control.

3.5 Real-Time Image Processing

As stated above, the EFSM utilizes real-time image processing from the onboard camera to align the vehicle with the target. This approach is chosen since it can be adapted to both indoor and outdoor environments and there are a multitude of powerful lightweight cameras available today [5]. We use a green circular pattern (placed next to the TP on the wall) to localize the I-BC for this particular task. Our image processing algorithm is divided into two phases: pattern detection and position estimation, which are described below. A simple algorithm for the pattern detection is used in order to reduce the computational burden on the onboard computer. The algorithm uses OpenCV 3 [98] and runs at $\approx$ 20 Hz on the onboard computer.

3.5.1 Pattern Detection

When the algorithm starts up, the live camera image feed is sent from the I-BC to a laptop so that the user can select the (green) color pattern and initialize the algorithm. Once the color pattern is clicked, the algorithm running on the I-BC's onboard computer calculates the Hue-Saturation-Value (HSV) of the selected pixel and uses these values to filter the subsequent camera video frames. Therefore, the ensuing image frames only show objects with HSV color values within a small threshold of the selected pixel. Thus, the pattern is detected (in the absence of any other similarly-colored objects). A blue rectangle is then drawn around the detected pattern, as shown in Fig. 3.7.

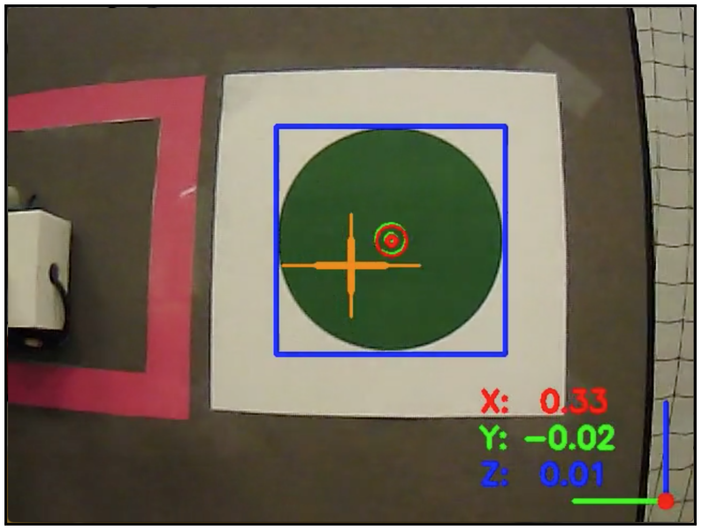


Figure 3.7. : Green pattern used to estimate the position of the I-BC. The bounding blue rectangle and red center point are used to calculate the distance from the pattern. The orange crosshairs indicate the center of the camera frame. The numbers are the x, y and z distances (in meters) from the center of the camera frame (crosshairs) to the center of the pattern (red dot). The X-axis is directed into the page, the Y-axis is to the left, and the Z-axis is directed upwards.

As the I-BC changes position, the pattern will appear bigger or smaller, depending on the distance from the target. Due to these changes and the variability in lighting conditions, the initial HSV values used as thresholds for the filter may no longer be valid. Therefore, the algorithm dynamically updates the threshold values for the color filter. Every time it detects the pattern, the HSV values of the pixels at the centroid of the patten are used as the filter thresholds for the next iteration.

3.5.2 Position Estimation

Once the pattern has been detected, the position estimate of the I-BC relative to the TP needs to be determined. The onboard attitude controller ensures that the I-BC stays perpendicular to the wall and performs only negligible yaw movements during the position estimation routine. The sonar sensor mounted to the end-effector is used to measure the distance to the TP (wall) while the position of the tracked color pattern in the image is used to determine the relative position between the I-BC and TP and compute movement commands in the Y- (left, right) and Z-axes (up, down) to align the I-BC with the target (see Fig. 3.7). The position of the tracked color pattern is initially computed in pixels relative to the center of the image frame. Therefore, the appropriate scale to convert from pixel values to a distance value in meters is needed. We make use of the known size (area) of the colored circle (10 *cm* diameter) and compare it with the corresponding pixel value in the image to make this conversion. To check the accuracy of this routine, we placed the I-BC at different distances from the wall, measured the exact distance using the Vicon motion capture cameras, and recorded the size of the pattern as seen through the camera (see Fig. 3.8). The area of the pattern decreases nonlinearly as the distance from the pattern increases. Fitting an exponential curve to this data gives us an estimate of the distance to the wall as a function of the pattern area (with an average error of about ± 4 *cm*), and the corresponding pixel-to-meter scale factor. Based on these values, the I-BC is commanded to move laterally and vertically to align the center of the image with the center of the color pattern.

The EFSM only uses the image processing algorithm to determine position estimates for control if the I-BC is within the highlighted region shown in Fig. 3.8. This ensures that the green pattern is within the camera's field of view and that the estimation errors are low.

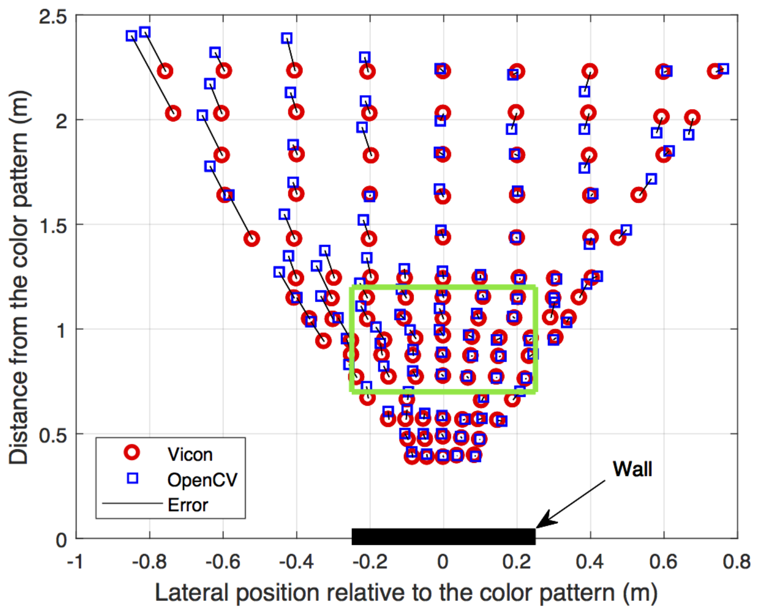


Figure 3.8. : Top view of the experimental room. Map of actual position obtained from the Vicon system and estimated position calculated with OpenCV. Estimation error increases further away from the pattern. Thus, we utilize the onboard camera for position estimates only when the I-BC is within the green rectangular zone (0.7 to 1.2 m away from the wall).

3.6 Experimental Results

Several flight tests were performed to experimentally validate the performance of the autonomous control strategy described above, and to compare the performance of autonomous versus manual flights.

3.6.1 Experiment Setup

Fig. 3.9 shows the experimental test setup for autonomous sensor mounting in the lab. The same table from section 3.3 was used as the target mounting surface. The use of this table simulates scenarios where the width of the desired mounting surface is constrained (e.g. mounting a sensor on a truss member), but we assume that the results from the experiments are also applicable to scenarios with larger mounting surfaces such as walls.

As discussed in section 3.5 above, a green circle (10 cm diameter) was used as a tracking target for the image processing algorithm. The red square outline indicates the desired

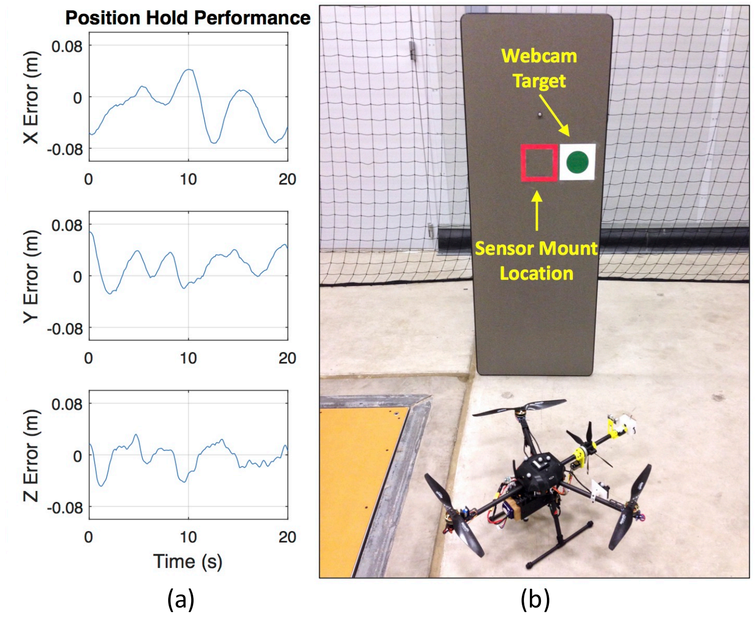


Figure 3.9. : (a) Performance of I-BC's FCU position controller. (b) Experimental setup for an autonomous sensor mounting task. The red square is centered on the desired mounting location (TP), and the width of the square (16 cm) indicates the acceptable placement error. The green circle is tracked by the camera. Table dimensions: 0.61 x 1.68 m.

sensor mounting location (TP = Target Position) and the acceptable positioning error (due to the limitations of the airframe/FCU). Any sensor placement inside or on the edge of the red outline is considered successful. In our experiments, the green target was offset to the right of the actual mounting location for convenience in measuring the sensor placement performance, and so that the sensor adhesive is attached to the actual mounting surface and not the printed paper target.

3.6.2 Sensor Placement Performance

Several successful autonomous sensor mounting test flights were performed to measure the sensor placement accuracy and repeatability. Fig. 3.10 shows the sensor mounting operation from both external and onboard viewing angles.

During the autonomous sensor mounting process, the EFSM calculates various position setpoints to maneuver the I-BC to the mounting position. Fig. 3.11 shows these setpoints

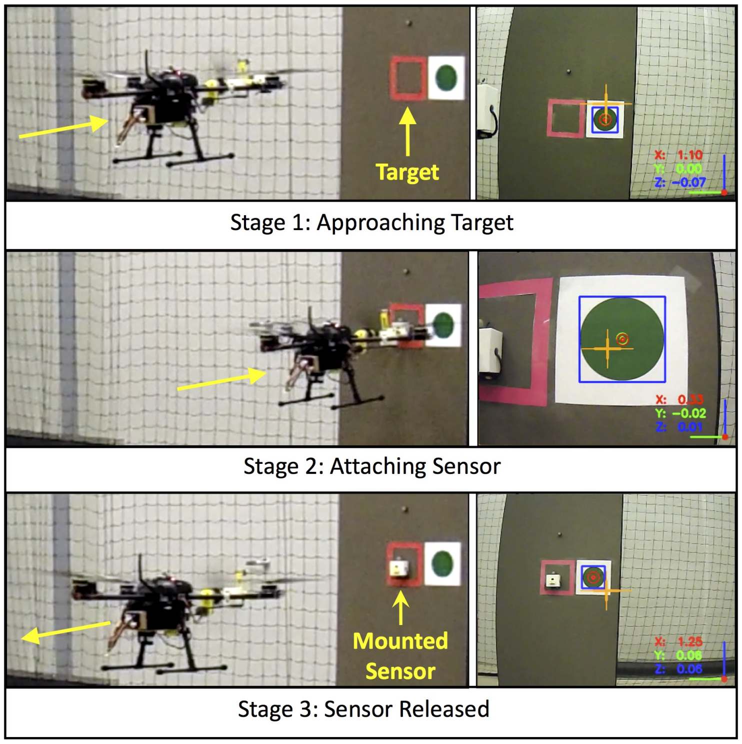


Figure 3.10. : Video screenshots from a successful autonomous sensor mounting operation. Right column: view from onboard camera.

(red squares) along with the I-BC's travel path for one of the successful autonomous sensor mounting flights. The force applied to the sensor during impact on the same flight is also included as an inset in Fig. 3.11.

When evaluating the sensor placement performance, we considered comparisons of the target mounting location (TP) against both the estimated mounting location (based on image processing) and the actual mounting location. As expected, the image processing estimation and actual sensor placement error each varied randomly between trials. Fig. 3.12 shows the absolute error in the estimated and actual sensor mounting locations for each of the 11 successful flight tests. The maximum overall placement error from the test flights was 9.3 *cm* and the average error was 6.3 *cm*.

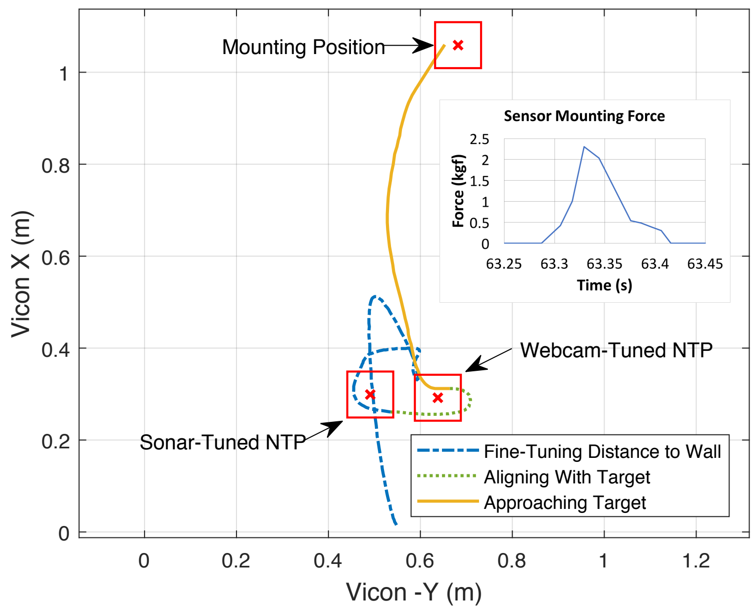
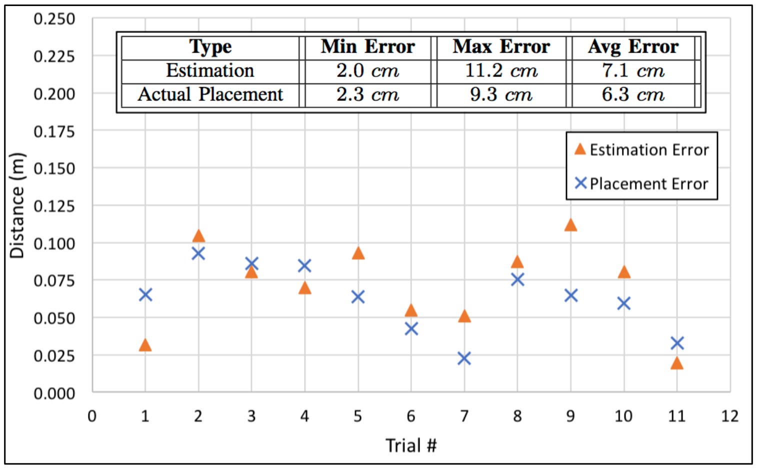


Figure 3.11. : Top view of I-BC position during sensor mounting task. Red squares indicate autonomously calculated position setpoints. Inset: Force on sensor package during impact.



Type	Min Error	Max Error	Avg Error
Estimation	2.0 cm	11.2 cm	7.1 cm
Actual Placement	2.3 cm	9.3 cm	6.3 cm

Figure 3.12. : Image processing estimate error and sensor placement error relative to ground truth (Vicon) for 11 test flights. Inset table lists minimum, maximum and average errors across all trials.

3.6.3 Comparison Between Manual & Autonomous Sensor Mounting

Compared to the manual sensor mounting flight tests described in section 3.3 above, the autonomous flights resulted in a 61% decrease in sensor placement error. In addition, the autonomous controller reduces the pilot's workload and allows the pilot to focus on monitoring the operation as a whole. These results demonstrate that the autonomous control strategy proposed in section 3.4 is effective at reducing sensor placement error, and helps improve the safety and reliability of the operation as a whole.

4. DOOR OPENING MANIPULATION TASK

Many aerial manipulation operations (e.g. infrastructure or industrial inspection and maintenance) may require prolonged contact with objects in the environment. Thus, we consider here the representative task of opening an electrical enclosure door. This operation can be extended to many applications such as opening roof-top HVAC access panels for inspection or to change filters, or opening enclosure doors to sample for trace nuclear contamination during decontamination and decommissioning of nuclear facilities as discussed in [99].

The door-opening task was first considered on the ground (with swivel casters attached to the I-BC's landing gear and a small propeller attached to the tail arm to provide yaw control, see [92]), and then in the air (see [100]). For each case, the design of a custom end-effector with embedded force and distance sensing will be presented, followed by real-time image processing techniques used for detecting and tracking the enclosure door, and a high-level control strategy that enables autonomous door-opening with the I-BC. Finally, experimental results are presented to validate the proposed control strategy and assess the performance of the system as a whole.

4.1 Door-Opening on the Ground

Fig. 4.1 shows the lab setup for opening the door on the ground. The enclosure was bolted to an L-shaped wooden stand, and the stand was held in place by a large weight placed on the lower portion of the stand extending along the ground behind the enclosure door. The dimensions of the electrical enclosure are 276 $mm \times 225$ $mm \times 98$ mm (height, width, depth), and the force required to open the door is ≈ 600 g (regulated by variable-length magnetic strips attached around the inside of the door. The I-BC was retrofitted with four 2 in (50.8 mm) diameter swivel casters (as shown in Fig. 4.1) to allow the I-BC to roll along the ground with minimal friction. A bi-directional tail fan was also added to control the I-BC's yaw angle during the door-opening operation.

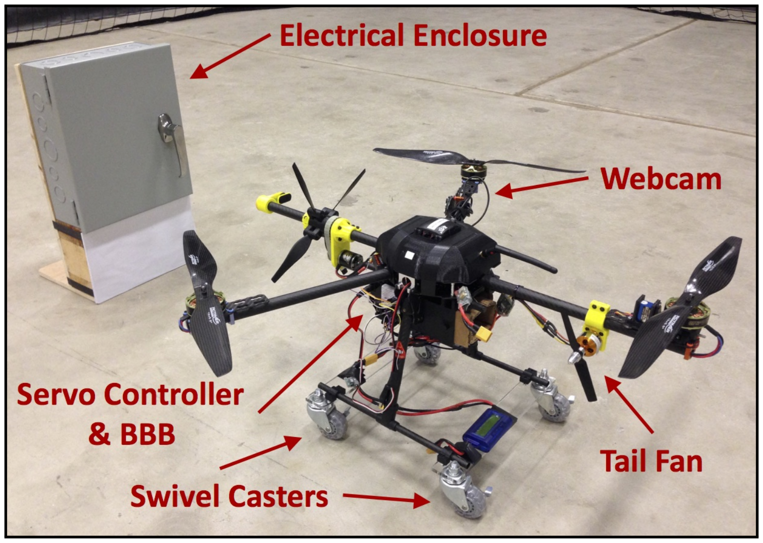


Figure 4.1. : Experimental setup for door opening on the ground.

4.1.1 End-Effector Design

Fig. 4.2 shows a simple, passive hook-shaped end-effector that was designed for the initial door-opening experiments on the ground. One FSR is attached on each side of the hook with two-sided foam tape to measure the pushing and pulling forces between the end-effector and the enclosure door/handle. A force plate was attached in series with the FSR on the pulling side of the hook to help distribute the forces evenly from the door handle to the force sensor. To prevent the end-effector from slipping while in contact with the door handle, a small strip of velcro (rough or "hook" side) was attached to the force plate (no velcro was added to the door handle). The cylindrical portion of the end-effector fits onto the end I-BC's front boom, and has a small gap on one side so that a single M3 screw can be used to clamp the end-effector in place.

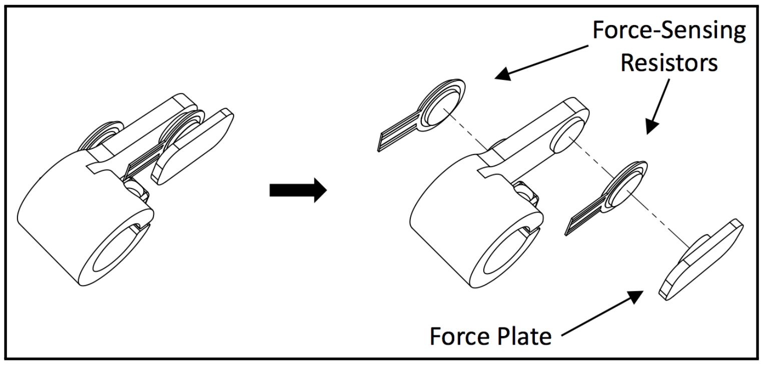


Figure 4.2. : Hook-shaped end-effector assembly and exploded view.

4.1.2 Real-Time Image Processing & Autonomous Control Strategy

To accomplish the autonomous door-opening task, ROS and the OpenCV 3.0 framework were used to develop a real-time image processing algorithm to detect and track the electrical enclosure door, and a state machine to control the I-BC via the boom-prop and tail fan. All of the software was run on an onboard BeagleBone Black (BBB) computer.

In the image processing algorithm, the size, shape, and aspect ratio of the enclosure door and door handle are predefined, and the incoming video stream from the onboard camera (Logitech C270) is converted into a stream of grayscale images. After performing an edge-detection process, the algorithm produces a closed-contour image and contours that do not match the predefined size, shape and aspect ratios of the enclosure door and/or handle are filtered out. Next, the centroids of the contours are calculated, and appropriate commands are sent to control the I-BC's motors based on the position of the centroids and the position (fixed) of the end-effector in the image.

The image processing algorithm and corresponding motor control is divided into three stages as described below. As a safety precaution, the boom-prop and tail fan are activated in short bursts to allow time for a human monitor to intervene in the event of a potential collision due to image processing errors.

Stage 1

During the initial approach to the door, the I-BC is commanded to follow an approximately straight-line trajectory until the end-effector is positioned behind the door handle (i.e. in contact or nearly in contact with the surface of the door). To achieve this, the image processing algorithm detects the centroid of the enclosure door and monitors the horizontal position of the end-effector relative to the door's centroid (Fig. 4.3). If they are aligned, a 'drive forward' command is sent to activate the boom-prop, otherwise, a 'turn left' or 'turn right' command is sent to activate the tail fan and align the end-effector with the door's centroid.

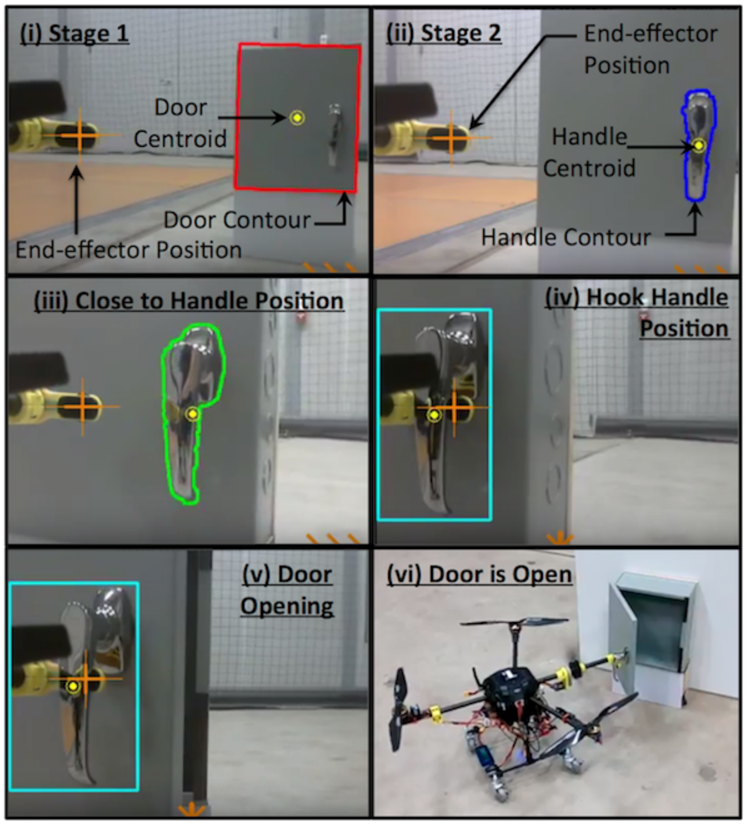


Figure 4.3. : Onboard view of autonomous door opening. (i) - (v) Images from the onboard camera at various stages and decision gates of the vision-based control scheme. (vi) I-BC successfully opening the door.

Stage 2

As the I-BC moves forward, the second stage of the algorithm is initiated when the contour of the door in the processed image becomes too large to fit within the video frames. In this stage, the algorithm focuses on the contour of the door handle and its centroid rather than the contour of the door (see Fig. 4.3(ii &iii)), and continues to issue the movement commands described above to correct the alignment of the end-effector with the door handle.

Stage 3

The third and final stage begins when the end-effector is in contact with the door or is about to contact the door. This is determined based on the size of the door handle contour. Based on the geometry of the end-effector, the I-BC is programmed to approach the door handle from the left side. Therefore, once the end-effector is close to the door, the I-BC is commanded to 'turn right' in order to hook the handle with the end-effector. In the designed algorithm, the contours of the handle and end-effector merge at this instant, signaling that the handle has been hooked (see Fig. 4.3(iv)).

Once the handle has been hooked, the I-BC is commanded to 'drive backward' to pull the door open (see Fig. 4.3(v)). The door is pulled until the pulling-force sensor values from the end-effector indicate that the door has been opened. This happens when the pulling-force values increase above a certain threshold (indicating that the handle is being pulled), and then fall rapidly below an experimentally determined threshold (indicating that the door has opened). After the door is opened, the I-BC is commanded to 'drive forward' to push the door closed. The door is then pushed until the pushing-force sensor values indicate that the door is closed (i.e. the force values rise and remain above an experimentally determined threshold for 0.25 s).

4.1.3 Experimental Results

Several experiments were performed to demonstrate the I-BC's ability to open and close the enclosure door on the ground. Manual experiments were first performed to collect data from the force sensors and record videos from the onboard camera. This information was used to determine appropriate force thresholds for detecting when the door is opened and closed, and to determine contour parameters for detecting the door and handle.

Fig. 4.3 shows the progression of the door/handle detection program as the I-BC approaches and opens the door. Fig. 4.4a and Fig. 4.4b show the force sensor output as the door was opened and closed, respectively. The initial bouncing of the pulling-force signal was ignored by a filter that uses a combination of the three most recent force values and a minimum applied force threshold to determine when the door was actually opened (seen as a sudden drop in the force at the moment that the door opens).

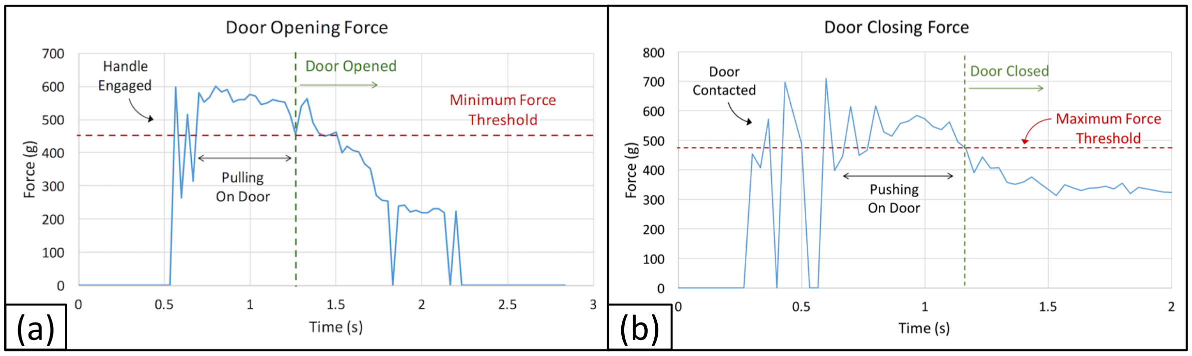


Figure 4.4. : Force sensor data during autonomous door opening (a) and door closing (b) tasks.

Discussion

Using the hook-shaped end-effector and image-processing techniques described above, the I-BC was able to autonomously open and close the enclosure door on the ground. However, due to the swiveling motion of the casters, the motion of the vehicle during the autonomous approach to the door was inconsistent. This, in combination with the fact that the forward distance to the door was unknown by the controller, led to several failed attempts where the end-effector was unable to hook onto the handle during Stage 3 of the control algorithm described above. Based on these results, new image-processing and control algorithms were designed and hardware changes were made to accomplish the door-opening task in the air (see Fig. 4.5).

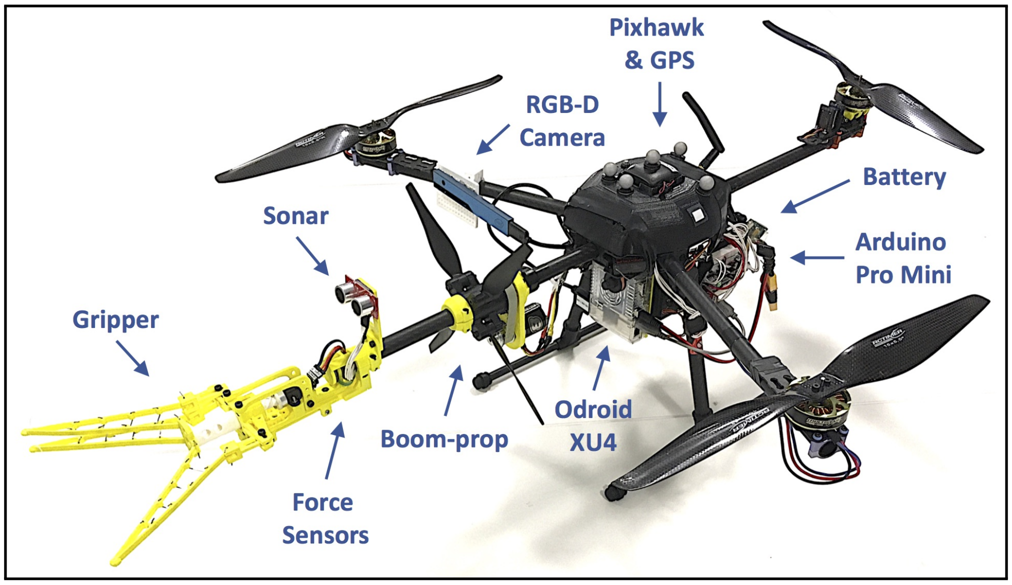


Figure 4.5. : I-BC prototype equipped with a compliant, three-finger gripper for door opening.

4.2 Door Opening in the Air

Fig. 4.6 shows the lab setup for opening the door in the air. The enclosure was bolted to the same L-shaped wooden stand used in section 4.1, and the stand was clamped from above to a table turned on end (same table as in section 3.3).

4.2.1 End-effector Design

As discussed in section 4.1.3, the I-BC was able to successfully open the enclosure door on the ground using a hook-shaped end-effector. However, the geometry of the hook end-

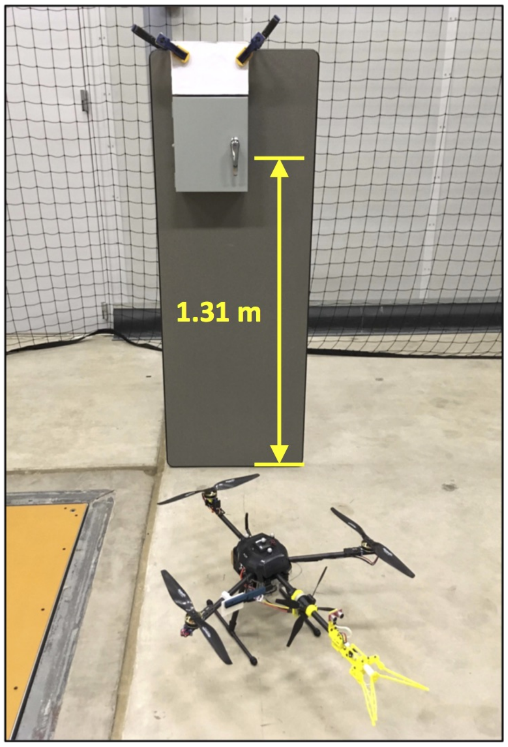


Figure 4.6. : Lab setup for an autonomous door-opening task.

effector necessitates a specific and precise trajectory to open the door. Furthermore, the positional drift measured in previous flight tests (see Fig. 3.9a) would likely make it difficult to maintain the hook end-effector in the necessary trajectory for the duration of the door-opening operation. Thus, a new end-effector was designed that is less sensitive to positional uncertainty, and is better suited to perform the door-opening task in the air.

Fig. 4.7 shows the new gripper design for autonomous door opening in the air (inspired by the Festo FinGripper [101] and open-source Tri-Max-Gripper [102]). All of the gripper components are 3D-printed from ABS plastic, with the exception of the rigid cross-members in the fingers, the drive motor and the fasteners. The following sections present key features of the gripper design.

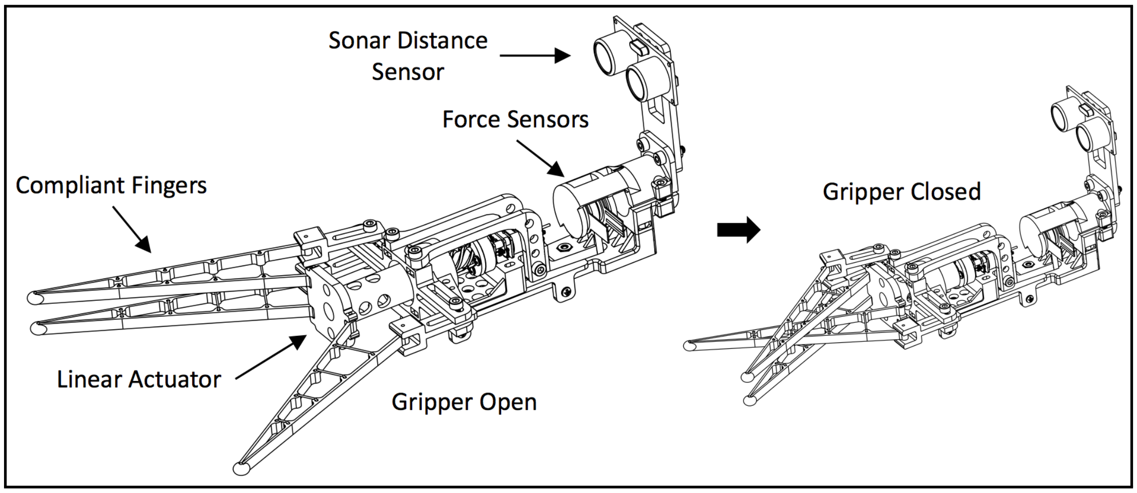


Figure 4.7. : 3-finger compliant gripper.

Physical Dimensions & Force Transmission

When fully opened, the span between the gripper fingers is 16 *cm* to accommodate up to ±8 *cm* of horizontal positioning error relative to the object to be gripped. Each of the three fingers are composed of two long, thin rectangular prisms (12 *cm* long × 5 *mm* wide, with a tapered thickness of 2.25 *mm* at the base to 1.5 *mm* near the tip), joined at the tip (at an angle of 11°) by an ellipsoid (5 *mm* × 7 *mm*). The elliptical tips on each finger help prevent object pull-out by providing geometric interference when the fingers are closed around an object.

The entire gripper mechanism is mounted on a sliding mechanism (see Fig. 4.8) so that longitudinal pushing and pulling forces (along the axis of the I-BC's front arm) can be measured during interaction tasks. The forces are transmitted through a force tab at the back of the gripper base that fits between two FSRs mounted on the clamp with VHB foam tape. The gripper base is fixed to a slider which is held vertically against a slide rail with two smooth horizontal rollers. Additionally, two vertical rollers mounted in the center of the slider are constrained in slots on the slide rail to keep the gripper aligned with the I-BC's forward axis. The rollers consist of #2 screws placed inside cylindrical nylon spacers, and help minimize the sliding friction between the slider and slide rail, particularly when the gripper simultaneously experiences twisting moments (such as the moment generated when a longitudinal force is applied to a finger on only one side of the gripper).

Preliminary force measurement tests with the sliding mechanism shown in Fig. 4.8 verified that there is negligible friction when force is applied to the center of the gripper. However, even with the alignment rollers in place, some twisting and jamming still occurred when unbalanced forces were applied to fingers on only one side of the gripper. To resolve this issue, the top finger on the two-finger side was connected to the lone finger on the opposite side with a nylon line passing through a small hole incorporated into the design at the center of each finger tip. When the gripper is fully open, the line is stretched tight and longitudinal forces applied anywhere in the gripper span are distributed between the two connected fingers, thus limiting the twisting affect of off-center forces. Also, when the gripper closes, the nylon line becomes slack so that objects can travel freely all the way to the base of the fingers.

The complete gripper mechanism, as shown in Fig. 4.7, has a mass of 120 g, with 84 g coming from the gripper mechanism, and the remaining 36 g coming from the sliding mechanism, clamp and sensors.

4.2.2 Compliance & Actuation Mechanism

The thin, flat walls on either side of each finger are connected at regular intervals by stiff cross-members (0.5 mm dia. steel wire), thus making them flexible in the horizontal direction and rigid in the vertical direction. This compliance allows the fingers to conform to a variety of shapes, and allows for variation in the shape of the door handle encountered during the door-opening task.

The gripper is actuated by a single motor with a stall-torque of 0.196 Nm (298:1 Pololu Micro Metal Gearmotor LP 6V with extended motor shaft), attached through a hub to a 3D-printed lead screw (see Fig. 4.9). Based on the speed of the motor, a four-thread lead screw and drive nut pair was designed with a pitch of 21.9 mm and a total length of 35 mm to allow the gripper to fully open or close in under 2 s. With the outer side of each finger attached

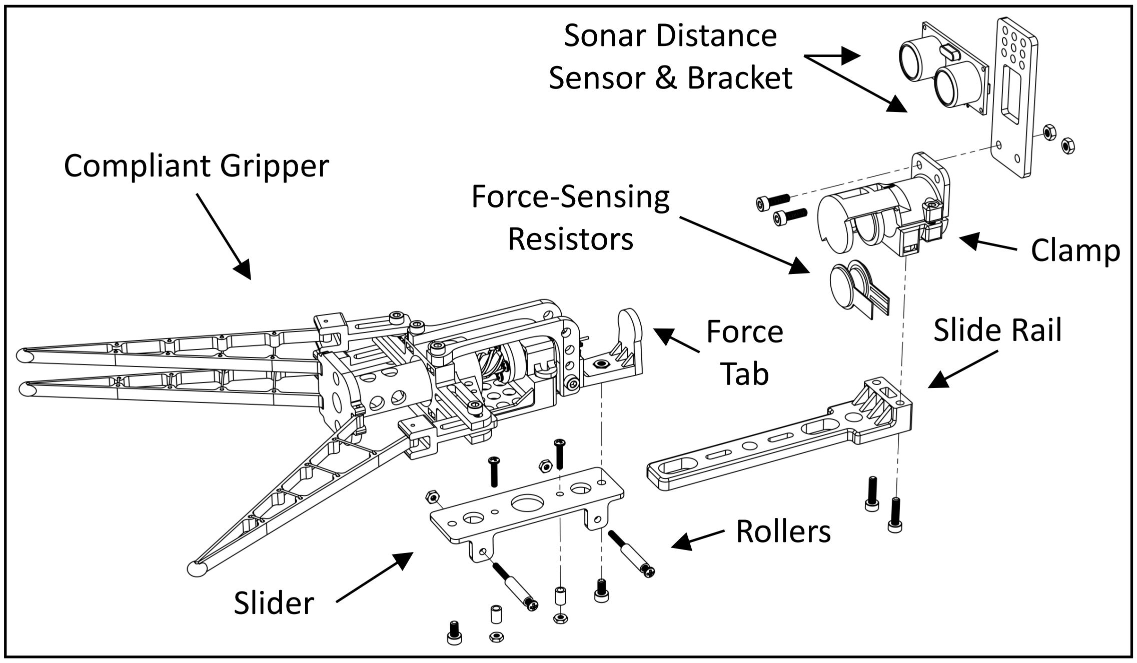


Figure 4.8. : Exploded view of 3-finger compliant gripper sliding mechanism.

to the anchor via a standoff, the linear motion of the plunge ring causes all three fingers to simultaneously rotate inward (close) or outward (open). When an object is encountered while the fingers are closing, the inside wall of each finger will deform (become concave) and the cross members will drive the outer wall to deform as well, causing the tip to curve inward, thus conforming around and caging the object. Two L-shaped support arms provide a rigid connection between the gripper base and the anchor (see Fig. 4.9) to prevent the anchor from twisting when an object is being gripped.

Gripper Position Control & State Machine

A custom Arduino library (IBCGripper) was developed to control the position of the gripper and implement a state machine to detect when an object is being gripped. This Arduino library was loaded onto the I-BC's Device Manager (see section 3.2.2) so that the gripper control interface and sensor data could all be accessed with a single ROS node on the onboard computer. The position of the gripper is controlled with classic PID control at

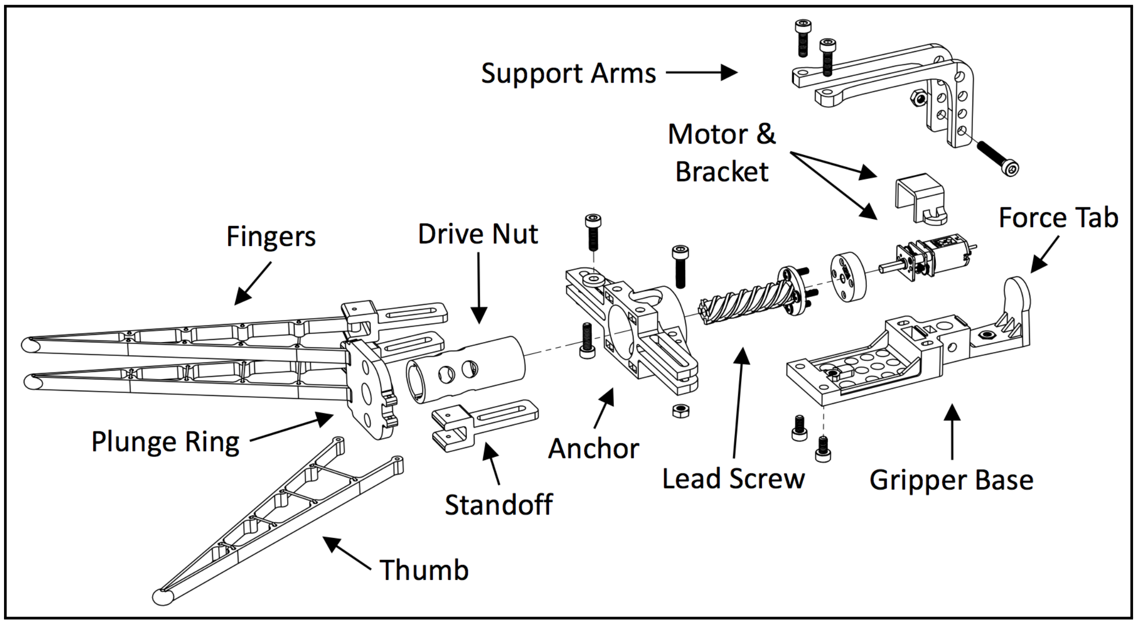


Figure 4.9. : Exploded view of 3-finger compliant gripper.

200 Hz with feedback from a magnetic encoder (1788 counts per revolution of the output shaft). Each time the gripper control program is started, the home position is determined by closing the gripper all the way until it stops moving (motor momentarily stalls), and then the gripper is opened all the way so it is ready to grab an object. When the gripper receives a grip command, it enters a `GRIP` state and closes until the motor stalls (indicating an object is being gripped) and then switches to a `HOLD` state and actively holds the gripper at its current position to maintain a firm grip on the object until an `OPEN` command is received.

4.2.3 Real-time Image Processing

The previous door/handle-detection algorithm in section 4.1.2 used a single layer of processing to identify the door and handle in real time, but suffered from occasional false detections without a uniform background behind the enclosure. For door opening in the air where the scene is more dynamic, additional processing layers were added to improve the robustness of the identification, including color-based filtering, extraction of histogram of oriented gradient (HOG) features, and applying a support vector machine (previously trained with sample images of the door). The updated door detection program uses both the RGB and depth video frames from the onboard RGB-D camera (Intel RealSense R200), which is mounted on the right arm of the I-BC (see Fig. 4.5). The R200 was configured to stream the RGB and depth frames at 30 Hz with an image resolution of 640×480 pixels. The updated algorithm is explained in detail below.

Door Detection

Pre-processing: The I-BC's onboard camera was mounted with the gripper in the field of view so that when the door handle is gripped, it can be seen in the video frames. However, this configuration occasionally results in the occlusion of the door, which poses a challenge in detecting it using image processing (see Fig. 4.10(a)). To remove this clutter, the pixels in the RGB frames where the gripper appears are replaced with pixels from the neighboring uncluttered regions. This was accomplished by copying the pixels from the rows just above and below the region where the gripper appears, and repeating them down/up to the midpoint of this region. This preserves vertical edges so that the door contour can still be detected, even when it is partially occluded by the gripper. Although the frames get somewhat distorted as a result of this modification (see Fig. 4.10(b)), the overall performance of the detection program is improved when the subsequent stages of image processing use this modified RGB image frame.

Contour Extraction: The goal of this stage is to extract the door contour from the RGB frame. The size, shape, and aspect ratio of the door contour, and the color of the door in the RBG frame were recorded as parameters. These parameters were then used to create thresholds to filter out the door contour candidates. During execution, this stage extracts the contours of all objects in the RGB frame (see Fig. 4.10(c)), and removes the unwanted contours with a filter based on the previously determined thresholds (see Fig. 4.10(d)). However, the size and shape of the door contour changes with the distance between the I-BC and the door. Thus, the thresholds used to remove the undesired contours were generalized to work in all possible cases. Though, as a result of this generalization, the filters are not able

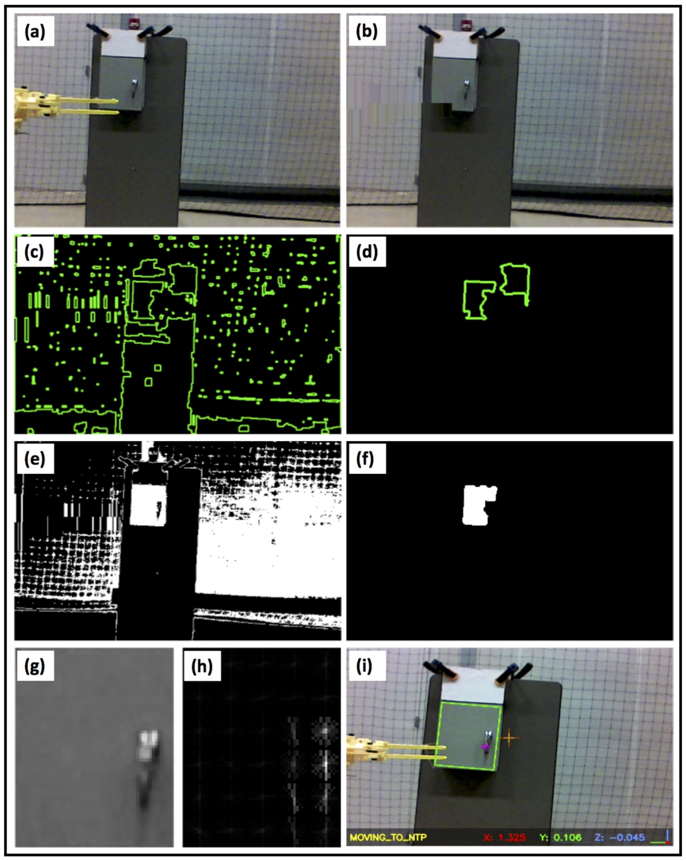


Figure 4.10. : Snapshots of the different stages of the door detection program.

to remove all of the unwanted contours (see Fig. 4.10(d)). Hence, a color filter is employed to remove the remaining unwanted contours. This filter's thresholds were chosen such that they encapsulate the RGB value of the door color while accommodating for variations due to changes in lighting conditions. The output of the color filter (see Fig. 4.10(e)) and the contour filter (see Fig. 4.10(d)) are then combined to create a mask for filtering out the actual door contour (see Fig. 4.10(f)).

Classification using a Support Vector Machine: While the previous stage works well in ideal conditions, it can occasionally produce false detections when other objects that have features matching the size, shape and color thresholds of the filters are encountered in the RGB frame. Thus, to further refine the results, we employed a support vector machine (SVM) trained on the histogram of oriented gradient (HOG) [103] features (see Fig. 4.10(h)) extracted from cropped RGB images of the door (see Fig. 4.10(g)). These HOG features give an idea of the approximate texture of the door. The SVM was trained with 117 positive examples and 296 negative examples. This trained model was used as an additional filter layer. Successful door detection occurs only when the set of HOG features associated with a given contour (produced by the *Contour Extraction* stage) is classified as positive. When the door is detected, the region of the door contour in the RGB frame is marked by a bounding rectangle in the final processed frame (see Fig. 4.10(i)). The application of this SVM effectively removed all remaining false detections, thus resulting in a consistent and robust door detection program.

Handle Localization

The X-Y-Z location of the CDH is determined relative to the center of the camera frame. The X, Y, and Z axes are directed out of the front of the camera, to the left of the camera, and upwards, respectively. The R200 camera provides a reliable estimate of depth (distance along X axis) only within a range of $\approx 0.5\ m - 2\ m$. Since the X distance between the gripper and the camera is 0.47 m, the minimum distance limit of the depth camera is not a concern. However, when the CDH is more than 2 m away from the camera, an alternative depth measurement is needed.

X Position of Door Handle: When the door is detected with $X > 2\ m$, the dimensions of the bounding rectangle in pixels are compared with the physical dimensions of the door in meters (known a priori) to obtain an approximation of the X distance to the CDH. For X distances $< 2\ m$, the depth value is obtained directly from the camera's depth frame. However, because the reflected infrared light projected by the R200 is not received back uniformly from all regions, the depth frames contain several black patches where the distance value is zero. If one of these black patches appears in the depth frame at the same location as the CDH, the reported X distance will be zero. To avoid this, when the door is detected, a histogram of the depth frame pixel values within the bounding rectangle of the door is created. This histogram consists of bins corresponding to every depth value, and the majority of the pixels in the bounding rectangle will fall in the bin with a depth value equal to the distance of the door. Thus, the depth of the bin with the most pixels gives a robust estimate of the X distance to the door. The known X offset of the CDH from the door plane can then be subtracted from this distance to get a reliable estimate of the X distance to the CDH.

Y-Z Position of Door Handle: In contrast to detecting the door, detecting the handle by itself is a more involved task due to its relatively small size, particularly at distances $> 2\ m$ where its shape becomes too generic and small. Even at distances $< 2\ m$ the view of the handle can occasionally be obscured by the gripper in the RGB frame, and the technique described above in *Pre-processing* is not helpful in this case, as it distorts the overall frame too much. However, the door can be detected consistently even at distances greater than $2\ m$. Hence, calculating the location of the CDH from the bounding rectangle (around the detected door) using the relative position of the CDH from the door edges (already known) provides a more reliable estimate of the Y-Z position. Once detected, the CDH is marked with a purple dot, and the Y and Z distance to the CDH from the center of the camera frame (marked with cross-hairs) are calculated and displayed in the final processed frame along with the X distance (see Fig. 4.10(i)).

4.2.4 Autonomous Control Strategy

An approach similar to the vision-based control strategy described in section 3.4 was used to accomplish autonomous door opening in the air using an EFSM. A simplified state transition diagram showing the process flow for a typical successful door-opening operation is shown in Fig. 4.11, with the following changes from the EFSM described in section 3.4. First, the `ALIGNING_TARGET` state uses the updated image processing algorithms described above to estimate the X-Y-Z location of the door handle relative to the I-BC. Second, in the `APPROACHING_TARGET` state, feedback from the gripper state machine is used to determine if the gripper has successfully grabbed the handle. If the gripper fails to grab the handle, the I-BC switches to the `LEAVING_TARGET` state to retreat from the door, and then, if the maximum number of retries (five) has not been exceeded, the I-BC re-enters the `APPROACHING_TARGET` state, otherwise, the mission is aborted and the I-BC enters the `WAITING_FOR_PILOT` state. Finally, the `OPENING_DOOR` state monitors the gripper position to determine if the handle has been grabbed successfully (indicated by the gripper being in its `HOLD` state), and then monitors the pull force to detect the door opening while the reverse boom-prop throttle is increased to 50%. If the pull force data does not indicate that the door has been opened within a specified time, or if the gripper position decreases below a minimum threshold (indicating that there is no object preventing the fingers from closing), then the I-BC switches to the `LEAVING_TARGET` state which is described in section 3.4.4 above.

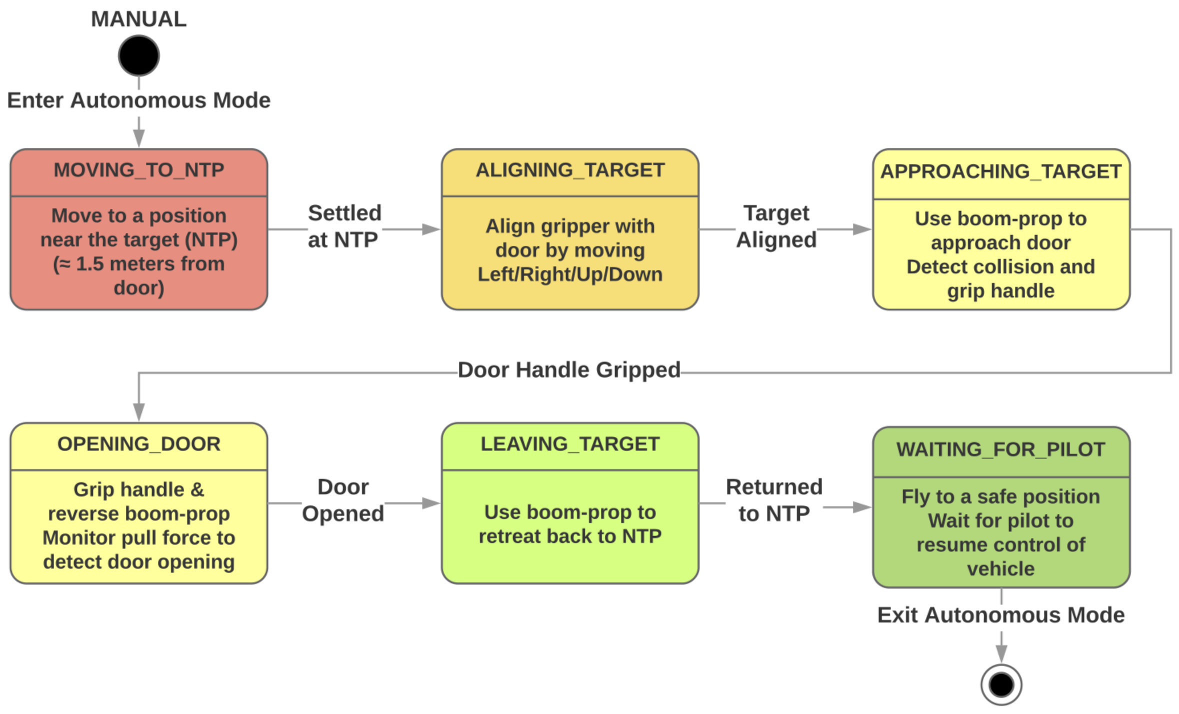

MANUAL
Enter Autonomous Mode
MOVING_TO_NTP
Move to a position near the target (NTP) (≈ 1.5 meters from door)
Settled at NTP
ALIGNING_TARGET
Align gripper with door by moving Left/Right/Up/Down
Target Aligned
APPROACHING_TARGET
Use boom-prop to approach door Detect collision and grip handle
Door Handle Gripped
OPENING_DOOR
Grip handle & reverse boom-prop Monitor pull force to detect door opening
Door Opened
LEAVING_TARGET
Use boom-prop to retreat back to NTP
Returned to NTP
WAITING_FOR_PILOT
Fly to a safe position Wait for pilot to resume control of vehicle
Exit Autonomous Mode

Figure 4.11. : Simplified state-transition diagram for in-air door-opening task.

The force thresholds used in the `APPROACHING_TARGET` and `OPENING_DOOR` states to detect the collision with the handle and the door opening were determined by manually bumping the gripper into the handle, and manually pulling the door open, respectively (with the gripper attached to the I-BC).

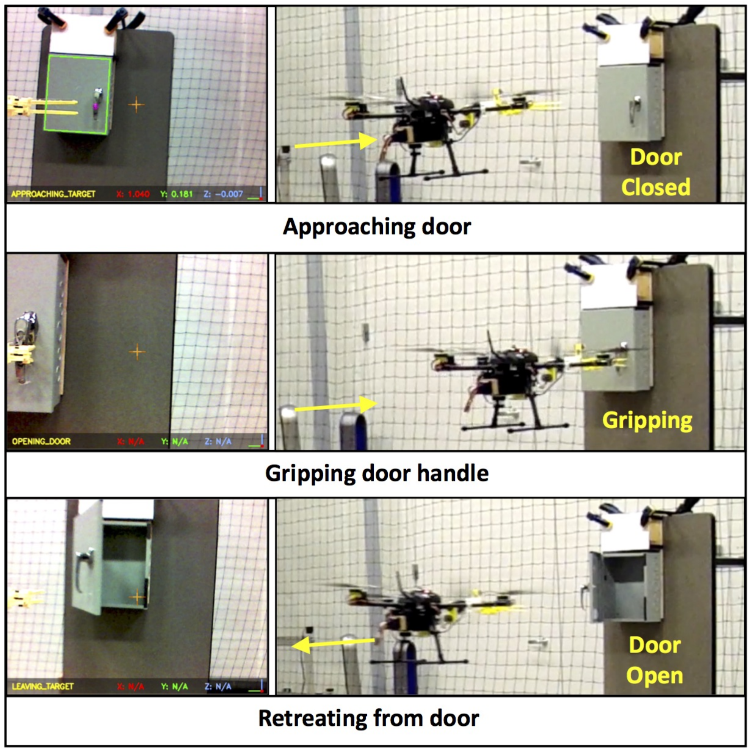

APPROACHING_TARGET X: 1.040 Y: 0.181 Z: -0.007
Door Closed
Approaching door
OPENING_DOOR X: N/A Y: N/A Z: N/A
Gripping
Gripping door handle
LEAVING_TARGET X: N/A Y: N/A Z: N/A
Door Open
Retreating from door

Figure 4.12. : Video screenshots from a successful autonomous door opening operation. Left column images are the view from the onboard camera.

4.2.5 Experimental Results

With the enclosure door mounted on a vertical surface as shown in Fig. 4.6, several autonomous door opening flight tests were performed to evaluate the performance of the I-BC's custom gripper and validate the efficacy of the image processing and autonomous control strategies described above. Fig. 4.12 shows video screenshots of a successful autonomous door opening operation from both onboard and external viewing angles.

In 29 total experiments, the I-BC successfully opened the enclosure door 23 times, requiring an average of ≈ 2 attempts. Five of the six unsuccessful door opening flight tests failed as a result of complications associated with vertical misalignment of the gripper with the handle. These flight tests would likely have succeeded if the grippable length of the handle (80 mm) was slightly longer.

Gripper Performance

In each of the flight tests, the compliant gripper was very effective at gripping the door handle. The nylon wire successfully distributed the handle impact forces across the gripper's fingers and guided the handle into the center of the gripper span in cases where the initial contact was off-center. The gripper sliding mechanism and force tab also performed as desired, successfully transmitting both pushing and pulling forces to the embedded FSRs and enabling robust detection of both the initial contact with the handle (see Fig. 4.13(a)) and the instant at which the door opens (see Fig. 4.13(b)).

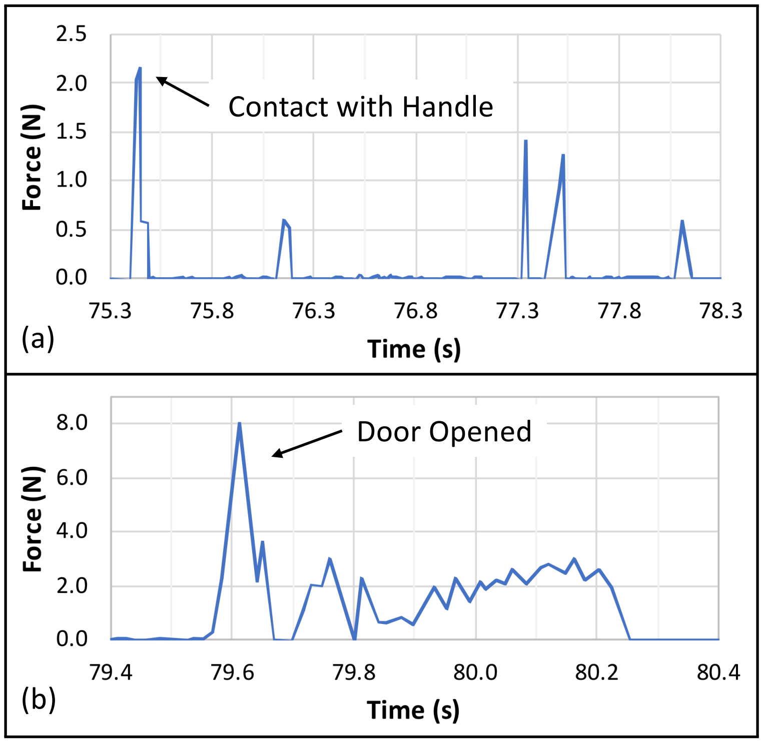
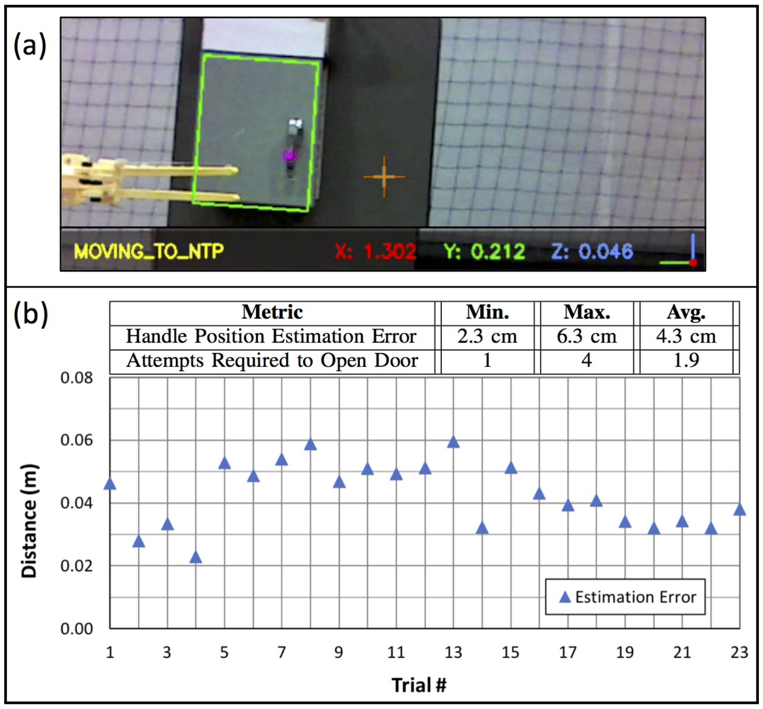


Figure 4.13. : Push and pull force sensor values during door opening operation.

Image Processing Performance

As depicted in Fig. 4.14(a), the door detection program successfully detected the door in all flight tests at a rate of $\approx 13\ Hz$, with no false detections. Fig. 4.14(b) shows the absolute error of the door handle position estimate for each of the 23 successful flight tests. The minimum, maximum, and average position estimation errors across all 23 successful flights were 2.3 *cm*, 6.3 *cm*, and 4.3 *cm*, respectively.



Metric	Min.	Max.	Avg.
Handle Position Estimation Error	2.3 cm	6.3 cm	4.3 cm
Attempts Required to Open Door	1	4	1.9

Figure 4.14. : (a) Output of the door detection program during a flight test; (b) Door handle position estimate error relative to ground truth (Vicon) for 23 successful autonomous flight tests. Inset table lists the minimum, maximum and average errors across all trials.

4.2.6 Conclusions

The effectiveness of the compliant end-effector and robust image processing algorithm designed for the in-air door-opening task were demonstrated with 23 successful flight tests conducted in the lab. Even though the I-BC occasionally fails to grab onto the door handle on its first attempt, the EFSM and onboard sensors enable the I-BC to recover from these errors and perform automatic retries. Also, the gripper's 16 *cm* span proved to be sufficiently wide to compensate for the horizontal positioning inaccuracies introduced by image processing errors and the drifting of the UAV.

5. ADVANCED AUTONOMOUS OPERATION IN UNSTRUCTURED ENVIRONMENTS

The experimental results presented in the previous two chapters show that the I-BC can perform both the autonomous sensor mounting task and the autonomous door opening task accurately and repeatably. However, so far, both tasks were only performed indoors with the vehicle's pose tracked by a Vicon motion capture system. A natural progression of this work is to consider the effects of larger uncertainties in the UAV position estimation introduced by less accurate vehicle localization methods, and consider the effects of disturbances encountered in outdoor scenarios (e.g. wind, bright sunlight, etc.). Thus, in the following sections, a strategy for precise localization relative to manipulation targets is introduced that enables the I-BC to perform autonomous aerial manipulation tasks in unstructured environments (without external positioning systems), including outdoors. This strategy will consider both of the prerequisites for autonomous execution of aerial manipulation tasks in real-world scenarios that were discussed in section 1.3, namely: UAV localization in unstructured environments, and UAV motion control relative to a target object.

5.1 I-BC Localization in Unstructured Environments

In order to operate the I-BC outside of the laboratory space, a new position estimation source was introduced in place of the Vicon motion capture system. Using the cameras (webcam, Kinect, Intel R200/D435) and SBCs (UP Board and Odroid XU4) available in the lab, several candidate VSLAM and VO algorithms were tested, such as ORB-SLAM [104], SVO [72], RTAB-Map [75], etc. However, as was reported in [69], each of these methods used most or all of the full computation power of the SBCs, leaving a very limited computation budget to run other necessary programs for autonomous control of the I-BC. Furthermore, many of the algorithms that were tested quickly stopped tracking or failed when the camera underwent sudden motions (a challenging scenario for many visual tracking algorithms [105]), and were not robust in areas with changing/poor lighting conditions or with few visual features (such as a large, relatively empty lab space). Thus, based on the discussion in section 1.3, we opted to use a downward-facing PX4Flow module, in addition to a LIDAR-Lite v3 distance sensor for position estimation of the I-BC. The positional drift of the PX4Flow reported in [68] of 0.25 m over a 28.44 m trajectory (0.87% drift) should be sufficient to perform the types of aerial manipulation tasks considered thus far, and is on par with the performance of other state-of-the-art VSLAM algorithms [74, 106]. Additionally, since the OF image processing and state estimation are offloaded to the PX4Flow and Pixhawk with this configuration, the full computational power of the onboard SBC can be dedicated to running visual target tracking and high level autonomous control algorithms.

5.2 I-BC Motion Control Relative to a Manipulation Target

The aerial manipulation tasks performed in chapters 3 and 4 both performed PBVS-type position control based on the output of different visual object trackers. In chapter 3, a green circle was placed near the target mounting location, with the intention that, in the future, a more generic visual tracking algorithm could be used to localize a desired sensor mounting location relative to some arbitrary visual features in the field of view (e.g. a bolt pattern on a bridge structure). In chapter 4, a more sophisticated tracking algorithm utilizing several different computer vision techniques was employed to autonomously identify a specific electrical enclosure door. This type of method may have applications in certain scenarios, such as industrial inspection/maintenance, where a common object or set of objects needs to be routinely visited, and thus, a more specific object tracking method (such as was used) is justified. However, considering these two applications, and other similar tasks involving horizontal interactions with vertical surfaces (such as contact inspection tasks), it is desirable to have a visual target tracking system that works consistently, independent of the specific visual features present in a given scenario. Moreover, given that most, if not all, aerial manipulation tasks are guided or at least monitored by a human operator, it is also desirable for the visual tracking system to allow for human selection of a desired manipulation target.

Thus, in the following sections, we present a method for visual target tracking that does not depend on traditional visual features at all. Instead, the algorithm accepts input from a human operator to identify a target point for a manipulation task. For brevity, we will refer to this featureless target tracking method as FTT. The FTT method uses a live RGB video stream from the UAV, viewed on a ground station computer, and allows the operator to select a target point by clicking on the point in the streamed image. Although the image is streamed to the ground station, all target tracking computation is done onboard the UAV. As will be shown in later sections, the output of this tracker can be directly inserted into the I-BC's autonomous position controller in the same way as the target tracking schemes presented in chapters 3 and 4.

5.2.1 Featureless Visual Target Tracking (FTT) Onboard a UAV

Figure 5.1 below provides an illustration of the main coordinate frames used for FTT on a UAV. **B** is a body-fixed frame (located at the FCU's origin), $\mathbf{C}^*$ is fixed on the camera's RGB imager (with a static offset from **B**), and **E** is the Earth-fixed frame of the onboard position estimator. Two additional coordinate frames not depicted in Fig. 5.1 are also used in the target tracking formulation: the camera frame **C**, which is co-located with $\mathbf{C}^*$, except with a right-down-front orientation (instead of the typical front-left-up configuration), and the end-effector frame **EE**, which is oriented the same as **B**, but located at the outer-most point of the end-effector along the x_b axis.

Considering some arbitrary, horizontal-facing aerial manipulation task to be performed, we will denote the vector from the camera to the target manipulation point expressed in the **C**-frame as ${}^{C}\mathbf{P}_{ct}$. As shown in Fig. 5.2, using a pinhole camera model [107], this point can be projected onto an image plane, yielding the 2D point ${}^{I}\mathbf{P}_{ct}$ (expressed in the image's right-down coordinate frame **I**, and measured in units of pixels).

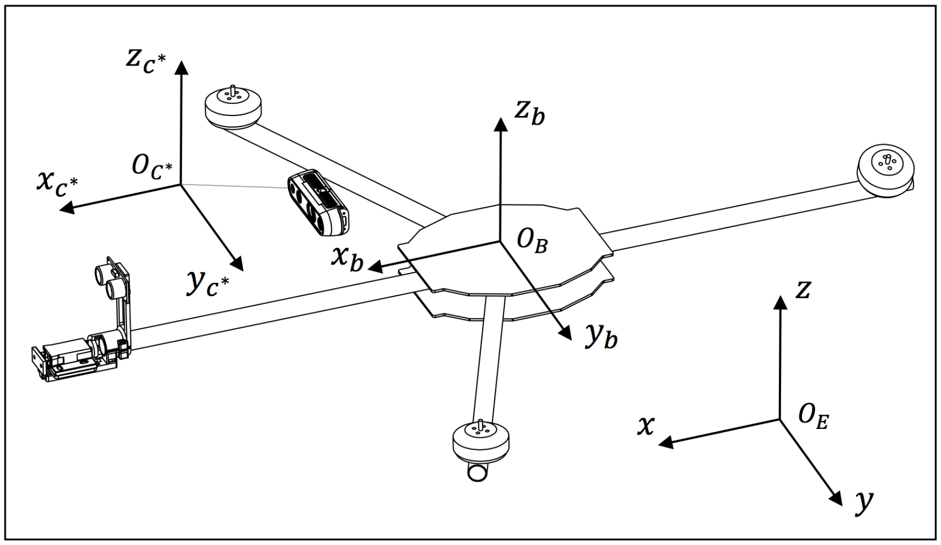


Figure 5.1. : Coordinate frames used for target tracking (depicted on the I-BC).

Given this desired vector from a camera to a target point represented by a pixel location in an RGB image, the following steps are performed to track the pose of the target point in the **E**-frame. First, the 3D point to be used for PBVS on the UAV ${}^{E}\mathbf{P}_t$ is calculated from ${}^{I}\mathbf{P}_{ct}$. Then, ${}^{E}\mathbf{P}_t$ is continuously projected back into the subsequent image frames and displayed to provide real-time feedback on the drift of the UAV's position estimator. We assume that the target point lies on a relatively flat, vertical surface (such as a bridge truss or a wall). Thus, only the yaw component of the target's orientation (which corresponds to the orientation of the surface normal in the **E**-frame) is required. This angle is continuously recalculated in each image frame based on the location of ${}^{E}\mathbf{P}_t$. Finally, an alignment error metric is computed based on the projected intersection point of the end-effector with the surface. Each of these steps is described in more detail below.

Converting a 2D Image Pixel Coordinate to a 3D Point

Using a pinhole camera model, as depicted in Fig. 5.2 (see [107] for more details), the 3D vector ${}^{C}\mathbf{P}_{ct} = [{}^{C}x_{ct}\ {}^{C}y_{ct}\ {}^{C}z_{ct}]^T$ from the camera to the target, expressed in the **C**-frame, is calculated from its corresponding pixel coordinate $({}^{I}x_{ct}, {}^{I}y_{ct})$ with the following equations:

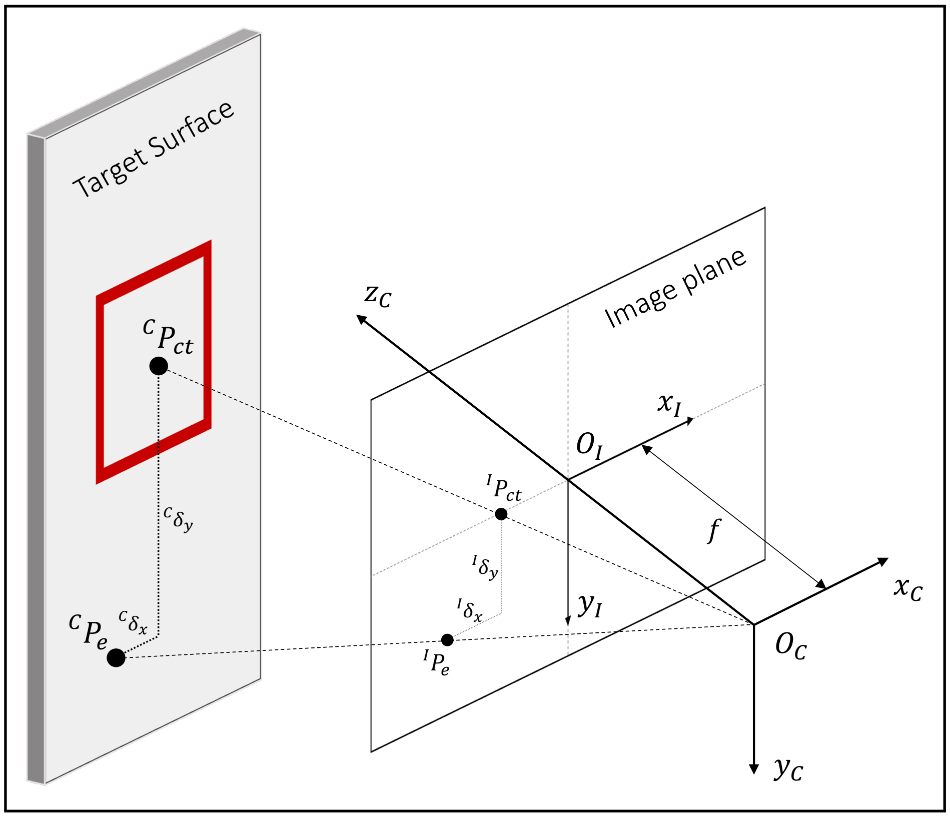


Figure 5.2. : Projection of a 3D target point onto a 2D image based on a pinhole camera model (f is the focal length of the camera) [107]. ${}^{C}\mathbf{P}_{ct}$ is the manipulation target point, ${}^{C}\mathbf{P}_{e}$ is the projected intersection of the UAV's end-effector with the target surface, and δ_x and δ_y are the alignment errors between $\mathbf{P}_e$ and $\mathbf{P}_{ct}$.

$$ {}^{C}x_{ct} = d \cdot \frac{{}^{I}x_{ct} - c_x}{f_x} \tag{5.1} $$

$$ {}^{C}y_{ct} = d \cdot \frac{{}^{I}y_{ct} - c_y}{f_y} \tag{5.2} $$

$$ {}^{C}z_{ct} = d \tag{5.3} $$

where f_x and f_y (measured in pixels) are the products of the physical focal length (measured in millimeters) of the lens and the size of the individual imager elements (which have units of pixels per millimeter), c_x and c_y are the displacement (in pixels) of the physical imager relative to the center of the image, and d is the distance (in meters) to the target point (obtained directly from the corresponding pixel in the depth image). In practice, these intrinsic camera parameters can be determined through a simple calibration routine, but for many cameras such as the Intel R200 and D435, these values are determined by the manufacturer, and can be accessed through the camera driver's SDK.

Now that ${}^{C}\mathbf{P}_{ct}$ has been obtained (measured in meters), it is converted into the **E**-frame in two steps. First, ${}^{C}\mathbf{P}_{ct}$ is transformed to the **C***-frame with:

$$^{C^*}x_{ct} = {}^{C}z_{ct} \tag{5.4}$$

$$^{C^*}y_{ct} = -{}^{C}x_{ct} \tag{5.5}$$

$$^{C^*}z_{ct} = -{}^{C}y_{ct} \tag{5.6}$$

Then, ${}^{C^*}\mathbf{P}_{ct}$ is rotated into the **E**-frame:

$$^{E}\mathbf{P}_{ct} = {}^{E}\mathbf{R}_{B}\,{}^{C^*}\mathbf{P}_{ct} \tag{5.7}$$

where ${}^{E}\mathbf{R}_{B}$ is the rotation matrix generated from the vehicle's roll (ϕ), pitch (θ) and yaw (ψ) Euler angles, given as:

$$^{E}\mathbf{R}_{B} = \begin{bmatrix} c_\psi c_\theta & -s_\psi c_\phi + c_\psi s_\theta s_\phi & s_\psi s_\phi + c_\psi s_\theta c_\phi \\ s_\psi c_\theta & c_\psi c_\phi + s_\psi s_\theta s_\phi & -c_\psi s_\phi + s_\psi s_\theta c_\phi \\ -s_\theta & c_\theta s_\phi & c_\theta c_\phi \end{bmatrix}$$

Note: s_θ, c_θ, t_θ are the abbreviated forms of $sin(\theta)$, $cos(\theta)$ and $tan(\theta)$ respectively in the above matrices.

Finally, the location of the target point in the **E**-frame is obtained as:

$$^{E}\mathbf{P}_{t} = {}^{E}\mathbf{P}_{ct} + {}^{E}\mathbf{P}_{uav} + {}^{E}\mathbf{R}_{B}{}^{B}\mathbf{P}_{BtoC} \tag{5.8}$$

where ${}^{E}\mathbf{P}_{uav}$ is the position of the UAV in the **E**-frame, and ${}^{B}\mathbf{P}_{BtoC}$ is the static offset from the **B**-frame to the **C**-frame expressed in the **B**-frame.

Projecting a 3D Point into a 2D Image Pixel Coordinate

Once a target point has been selected, it is useful to display its location in the image stream from the UAV so the user can confirm that it is correct. This also provides feedback to the user about the quality of the UAV position estimation, since the target point will drift in the image relative to the real-world features proportionally to the drift of the **E**-frame relative to the real world.

This is accomplished by recomputing ${}^{I}\mathbf{P}_{ct}$ (see Fig. 5.2) each time a new image frame is received from the onboard camera. First, ${}^{E}\mathbf{P}_{ct}$ is obtained as follows:

$$ {}^{E}\mathbf{P}_{ct} = {}^{E}\mathbf{P}_{t} - {}^{E}\mathbf{P}_{uav} - {}^{E}\mathbf{P}_{BtoC}. \tag{5.9} $$

Then, ${}^{E}\mathbf{P}_{ct}$ is rotated into the $\mathbf{C}^{*}$-frame:

$$ {}^{C^*}\mathbf{P}_{ct} = {}^{B}\mathbf{R}_{E}{}^{E}\mathbf{P}_{ct} \tag{5.10} $$

Note that ${}^{B}\mathbf{R}_{E} = ({}^{E}\mathbf{R}_{B})^{-1}$ and ${}^{B}\mathbf{R}_{E} = {}^{C^*}\mathbf{R}_{E}$ since the axes of the **B**-frame and $\mathbf{C}^{*}$-frame are always aligned. Finally, ${}^{C}\mathbf{P}_{ct}$ is obtained using Equations (5.4), (5.5), and (5.6) and projected into pixel coordinates with the following equations:

$$ {}^{I}x_{ct} = f_x \cdot \frac{{}^{C}x_{ct}}{d} + c_x, \tag{5.11} $$

$$ {}^{I}y_{ct} = f_y \cdot \frac{{}^{C}y_{ct}}{d} + c_y \tag{5.12} $$

Estimating End-effector Alignment Error

Since the **E**-frame will always drift somewhat relative to the real world, an alignment error metric is also computed to allow a human operator to visualize the potential effect of this drift on the manipulation task being performed. The alignment error is determined based on the projected intersection point of the end-effector with the target surface ${}^{C}\mathbf{P}_{e}$ (assuming forward motion only along the UAV's x_b-axis). Using the known static offset between the **C**-frame and **EE**-frame, the intersection point is defined in the **C**-frame as ${}^{C}\mathbf{P}_e = [{}^{C}x_{CtoEE}, {}^{C}y_{CtoEE}, {}^{C}z_{ct}]^T$ (see Fig. 5.2). The horizontal and vertical alignment errors relative to the actual target point can then be obtained by subtracting ${}^{C}\mathbf{P}_{ct}$ from ${}^{C}\mathbf{P}_e$. Since ${}^{C}\mathbf{P}_{ct}$ is computed using ${}^{E}\mathbf{P}_{uav}$ and ${}^{B}\mathbf{R}_E$ (see Equations (5.9) and (5.10)), any accumulated errors in the UAV pose estimate will be reflected in these alignment errors. The x and y components of these errors are depicted in Fig. 5.2 above as ${}^{C}\delta_x$, and ${}^{C}\delta_y$, respectively. Using the same formulation shown in Equations (5.11) and (5.12), ${}^{C}\mathbf{P}_e$ is also projected onto the image plane so it can be displayed to the user.

Estimating the Orientation of the Surface Normal

In addition to determining the 3D position of a manipulation target in a fixed global coordinate frame, its orientation relative to the UAV must also be determined so that a particular manipulation task can be performed successfully. How this orientation is defined depends largely on the type of manipulation being performed and the specific interface between the UAV end-effector and the target object. However, in the case of horizontal-facing manipulation tasks such as the sensor mounting and door-opening tasks discussed in the previous chapters, the orientation of the object can be generalized to the orientation of a vertical surface on which the manipulation target resides. Thus, we present here a method for determining the UAV's yaw angle ϕ relative to a vertical surface. This method can be applied in a variety of scenarios such as contact inspection, sensor delivery, surface cleaning, etc. where a vertical surface is to be inspected or interacted with.

Assuming that the target surface is relatively flat (as depicted in Fig. 5.2), the angle estimation is formulated as a linear regression problem, for which many robust and efficient solutions already exist. Viewing the scene from above, we seek the formula of a horizontal line (2D) on the face of the target surface (see Fig. 5.3):

$$h_\theta(x) = \theta_0 + \theta_1 x \tag{5.13}$$

Taking a set of n depth measurements across different x coordinates in the $\mathbf{C}$-frame, the parameters θ_0 and θ_1 are obtained by minimizing the following squared error function:

$$\frac{min}{\theta_0\theta_1} \; \frac{1}{2n}\sum_{i=1}^{n}(z_i - h_\theta(x_i))^2 \tag{5.14}$$

This is solved using the normal equation [108]:

$$\mathbf{\Theta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{z} \tag{5.15}$$

with $\mathbf{\Theta}$, $\mathbf{X}$ and $\mathbf{z}$ as defined below.

$$\mathbf{\Theta} = \begin{bmatrix}\theta_0 & \theta_1\end{bmatrix}, \qquad \mathbf{X} = \begin{bmatrix}1 & x_1\\ \vdots & \vdots\\ 1 & x_n\end{bmatrix}, \qquad \mathbf{z} = \begin{bmatrix}z_1\\ \vdots\\ z_n\end{bmatrix}$$

An illustration of this approach is shown in Fig. 5.3, where $n = 10$ depth measurements are taken to the left and right of the target point computed as described in the previous sections. Considering that the slope of $h_\theta(x)$ represents the change in depth divided by the change in horizontal position ($\frac{dz}{dx}$), we let $dx = 1$ and obtain ϕ with:

$$\phi = -tan^{-1}(\frac{\theta_1}{1}) \tag{5.16}$$

(note that the negative of the arc tangent is used so that ϕ is positive for counter-clockwise rotations of the camera relative to the target surface).

5.3 Experimental Validation of FTT

Several experiments were performed to characterize the accuracy and repeatability of the FTT method presented above, and validate its effectiveness for use in aerial manipulation tasks. An Intel D435 RGB-D camera was chosen as the preferred sensor for these experiments

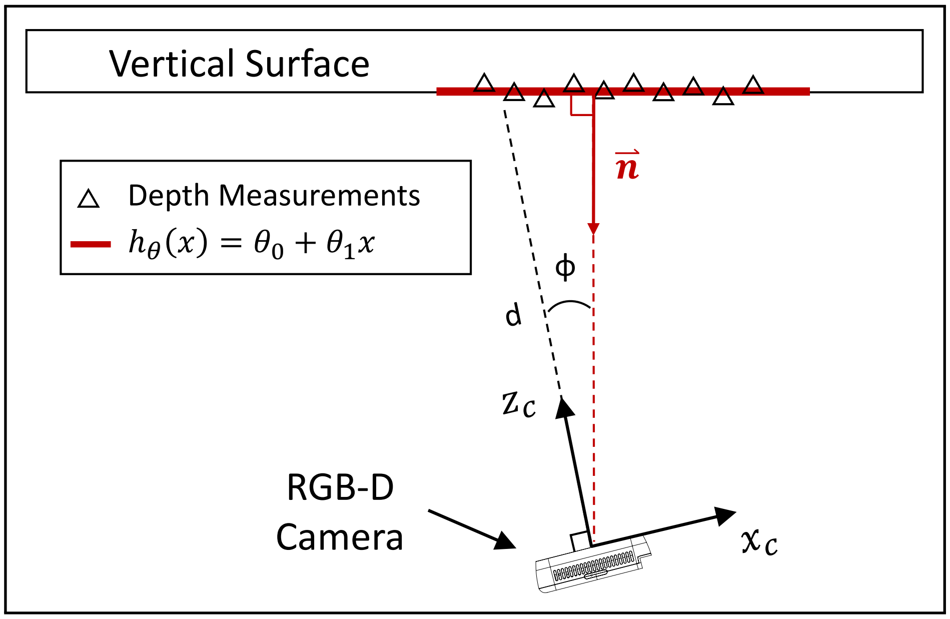


Figure 5.3. : Top view of angle estimation relative to a vertical surface using linear regression. ϕ is the counter-clockwise offset from perpendicular, $\vec{n}$ is the surface normal, and d is a depth measurement from the camera.

because of its small size and weight, and its superior performance in both indoor and outdoor environments [109,110]. The camera was configured to stream both RGB and depth images at 30 Hz with a resolution of 640×480 pixels, and the depth frames were synchronized and aligned with the RGB frames.

5.3.1 Graphical User Interface (GUI)

Using OpenCV and ROS, a GUI was developed to implement the FTT method on the I-BC. Raw color and depth images from the camera are retrieved and processed onboard using an Odroid XU4 SBC, and annotated images are compressed and streamed over WiFi to a ground station laptop using a ROS topic. Fig. 5.4 shows a screenshot of the FTT GUI during an experimental flight test. When the user left-clicks on a desired target point in the RGB image stream, a gray "X" is displayed in the image at ${}^{I}\mathbf{P}_{ct}$. Then, if the displayed location of the target point is deemed acceptable, the user right-clicks on the image to confirm the selection and the "X" changes from gray to red. The pose of the target point is then tracked as described in section 5.2.1 above, updating the displayed location of the red "X" accordingly. The projected intersection of the end-effector with the target surface is displayed with a green circle and dot, and details about the position and orientation of the target point are displayed at the bottom of the screen. The cyan rectangle around the target point shows the region used for angle estimation. This rectangle is divided evenly into ten smaller rectangles, which provide the x-z pairs needed for linear regression (see Equation (5.15) above). To help smooth out the noise in the angle measurements determined from the linear regression, a circular buffer of five angles is used to compute a moving average of the angle, which is rounded to the nearest integer. Based on initial experiments, the size of the cyan region in the image was chosen so that it spans an 8 cm × 60 cm area when the camera is 1.5 m from the surface. Lastly, the yellow text in the bottom left of the image in Fig. 5.4 displays information about the state of the I-BC's autonomous flight controller.

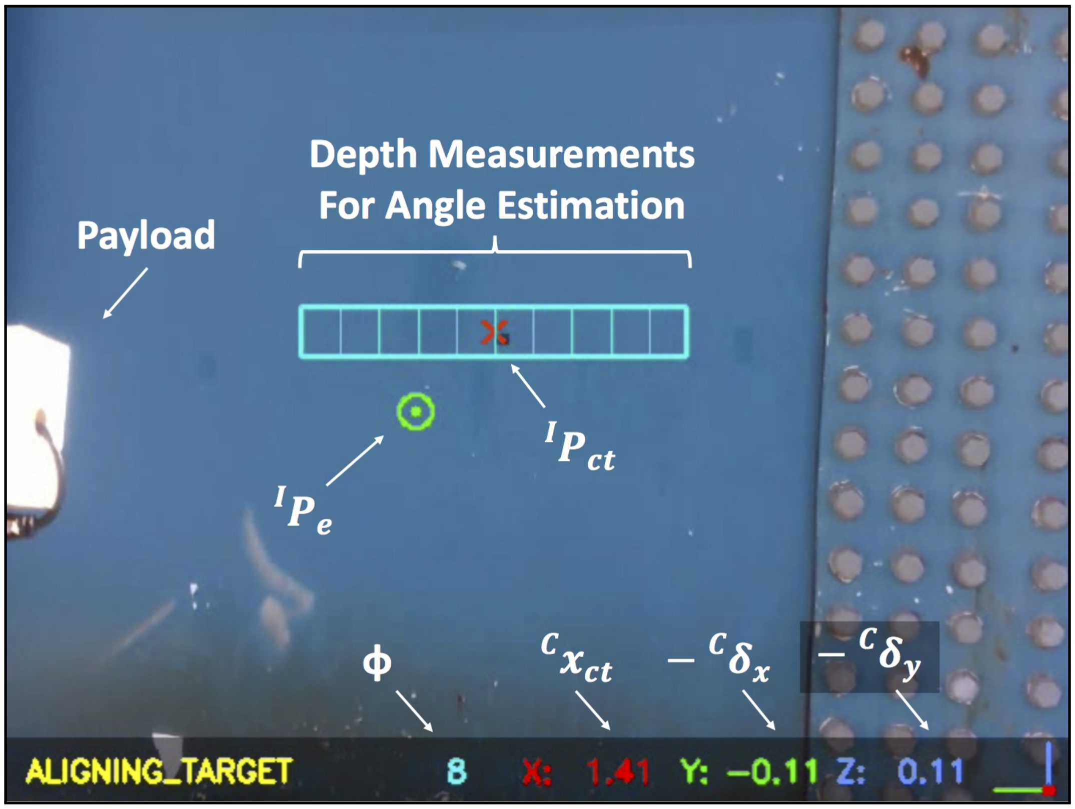


Figure 5.4. : Screenshot of the GUI for the FTT method. The pose of the manipulation target is displayed at the bottom, and relevant variables as defined in 5.2.1 are included for reference.

5.3.2 Depth Measurement Accuracy

It was noted during initial tests with the depth frames of the D435 camera that occasional black patches (indicating a measurement of zero distance) may appear in the image, more frequently at larger distances. To prevent potential issues associated with these patches, we employed the following method to obtain depth measurements from the D435. First, a histogram (5 mm bin size) is obtained for all pixels in a rectangular region around the selected point (20×20 pixels). Then, all bins for distances less than a threshold (based on the minimum detectable distance of the camera: ≈ 0.25 m) are thrown out, and out of the remaining bins, the two bins containing the highest number of pixels are used to compute the depth using a weighted average. The formula for the weighted average is:

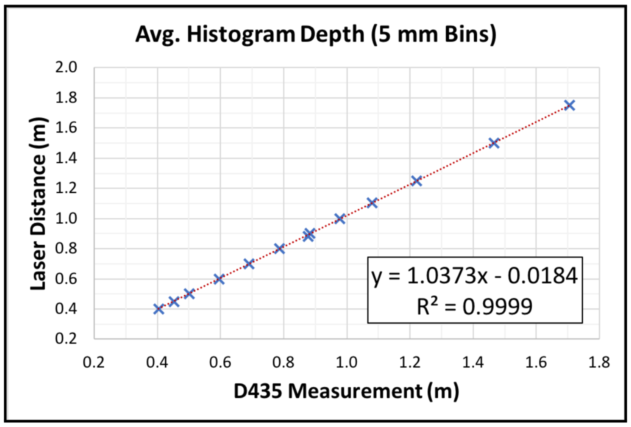
$$d = d_1 \frac{n_1}{n_1 + n_2} + d_2 \frac{n_2}{n_1 + n_2} \tag{5.17}$$

where d_1 and d_2 are the depth bins with the most and second most pixels, respectively, and n_1 and n_2 are the number of pixels at depths d_1 and d_2, respectively. This approach has the added benefit of smoothing out some of the noise in the depth measurements.

Prior to running any experiments, a laser distance measuring tool was used to calibrate the D435's depth measurements with a linear fit (see Fig. 5.5, left). The calibration was then verified with 25 additional measurements in a Vicon motion capture space at distances of ≈ 0.25 m to 1.5 m. The average and maximum errors in these measurements were 2 mm and 5 mm, respectively (see Fig. 5.5, right). This linear fit calibration was thus used for all subsequent depth measurements in our experiments.

5.3.3 Target Position Estimation Accuracy

The accuracy of the FTT method's target position estimation was assessed using the GUI described above and several reflective markers in a Vicon motion capture space. As shown in Fig. 5.6, five markers were distributed vertically and horizontally on a flat, vertical surface to test the FTT method's performance across the camera's full field of view. The estimated



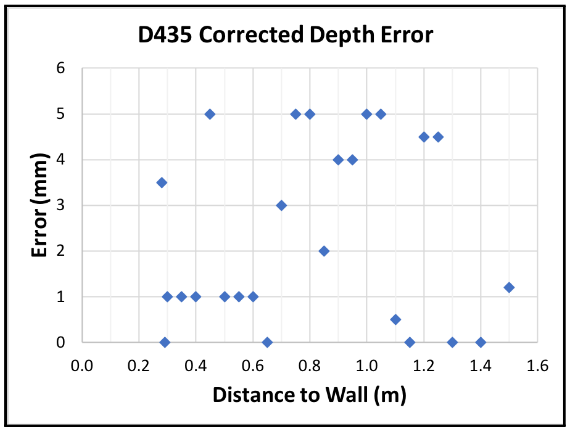


Figure 5.5. : Intel D435 depth calibration. Left: Linear fit of depth measurements; right: depth measurements corrected by linear fit (measured by Vicon).

position of the markers was calculated by clicking on each marker in the image stream five separate times at each of six different distances from the surface (resulting in 150 data points). Each time a marker was clicked, the position of the marker as estimated by the FTT method was recorded. Then, these positions were transformed into the Vicon coordinate frame and compared to the known locations of each marker. The results of these tests are shown in Fig. 5.7. Some variation and outliers are to be expected based on the difficulty in clicking on the exact center of each marker as it is displayed in the image, particularly at larger distances where the markers occupy only a few pixels. Even so, for all tests up to 1.5 m from the surface, the maximum error was 4.0 cm, the average error was 2.4 cm, and the average standard deviation was 0.4 cm. While it is difficult to compare this performance directly with many existing works in the visual object tracking literature (since they tend to report either qualitative tracking metrics such as detection success rate, or computational metrics such as FPS [79, 80, 84]), this estimation error is a significant improvement compared to the results obtained using artificial visual markers as reported in [83]. The error is also much smaller than the expected errors in the UAV position estimation, and was thus deemed acceptable for use in performing autonomous aerial manipulation tasks (this was later confirmed through several experimental flight tests, see section 5.4 below).

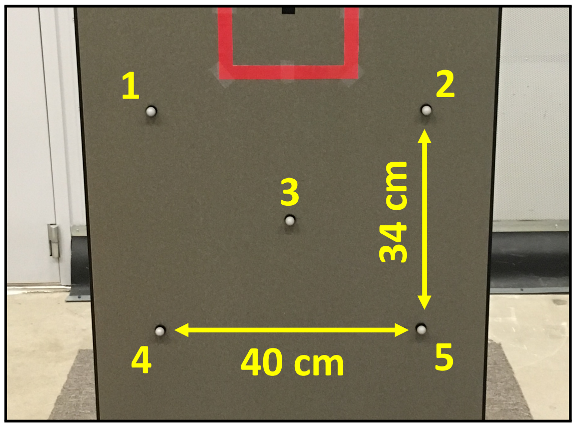


Figure 5.6. : Vicon markers used to characterize the FTT method's target position estimation performance.

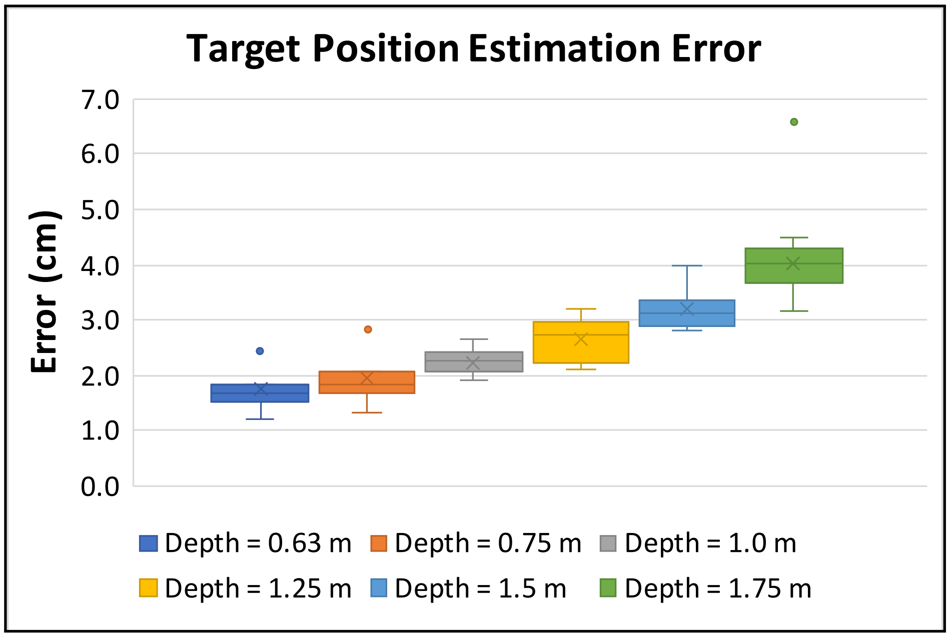


Figure 5.7. : FTT absolute target position estimation error at various distances.

5.3.4 Approach Angle Estimation Accuracy

The same vertical surface and Vicon markers shown in Fig. 5.6 were used to assess the accuracy of the FTT method's angle estimation onboard the I-BC. For ground truth, an equation was obtained for a line between markers 1 and 2 in Fig. 5.6 to determine the yaw angle of the front of the surface in the Vicon space (as seen from above). Then, the I-BC was placed in front of the surface, and, at each of three different distances, the orientation of the I-BC relative to the surface was measured (sweeping from $-35°$ to $35°$). Fig. 5.8 shows a plot of the absolute error of the angles measured using the FTT method compared to ground truth. Although the estimation error does increase for larger angles (up to $\approx \pm 10°$ error at $\pm 35°$), for angles between $-15°$ and $15°$, the maximum estimation error is $4°$ and the average error is only $2°$. In practice, it is reasonable to assume that a pilot, GPS heading, or other heading estimate can align the UAV within $< \pm 15°$ of the target surface in preparation for performing an autonomous aerial manipulation task. Furthermore, the high level autonomous controller on the UAV can use the FTT angle measurements to update its heading iteratively and eventually operate in the lower-error region ($\approx \pm 15°$).

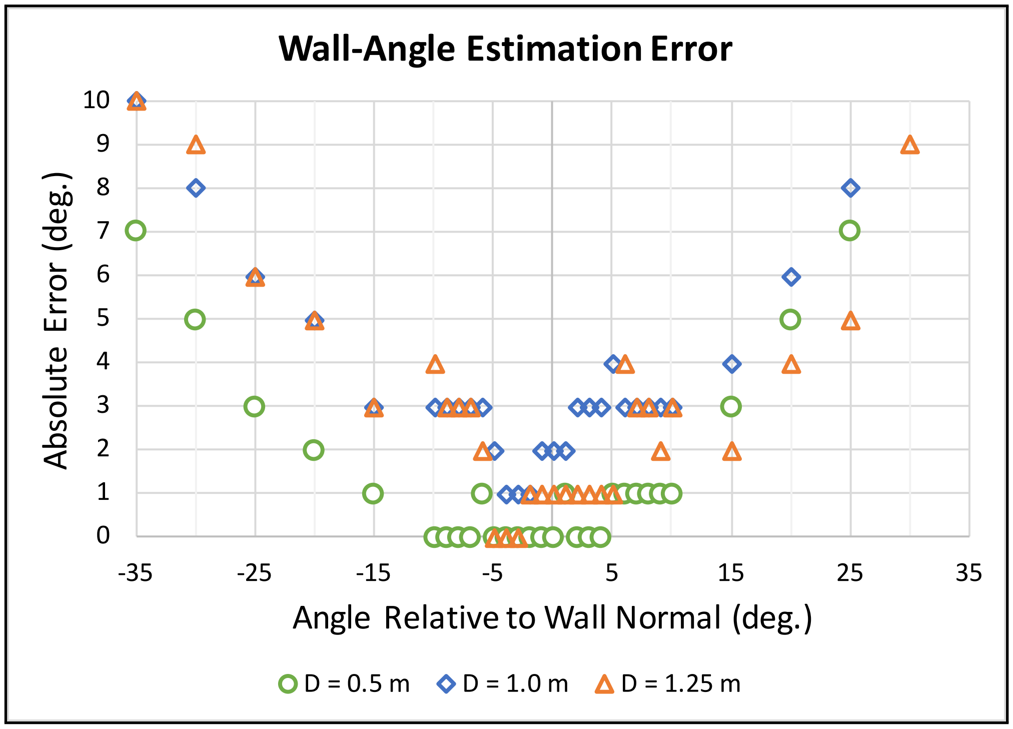


Figure 5.8. : Absolute error of FTT wall angle estimation.

5.4 Autonomous Sensor Mounting Flight Tests

With the promising results of the FTT validation described above, we implemented both the UAV localization method and the FTT method described above on the I-BC, and conducted several flight tests to evaluate their combined performance in unstructured environments. For these tests, we considered the autonomous sensor mounting task described in chapter 3, and performed several flights in both indoor and outdoor settings without the aid of external position estimation systems such as Vicon motion capture or GPS. For high level autonomous control of the I-BC, the state machine described in section 3.4.1 was used, with the following modifications. Instead of using a preset NTP, a pilot flies the I-BC close to the desired mounting location using direct line-of-sight, at which point, the I-BC hovers (using the assisted position control mode of the FCU) until the pilot toggles a switch on the RC transmitter to engage the autonomous flight controller. Then, using a laptop ground station, the desired mounting location is selected by clicking in the live video stream from the onboard camera. During the `ALIGNING_TARGET` state, the position and orientation setpoints for the I-BC are determined using the output of the FTT method. Specifically, ${}^{C}\delta_x$, ${}^{C}\delta_y$, and ϕ are used (see Fig. 5.2 and Fig. 5.3 above). Finally, the `APPLYING_SENSOR` state is combined with the `APPROACHING_TARGET` state since the initial impact force is sufficient to toggle the PTRM and release the payload.

5.4.1 Indoor Flight Tests

Similar to the experiments described in section 3.6 (see Fig. 3.9), a red square outline ($20\ cm \times 20\ cm$, $2\ cm$ line-width) was mounted to a vertical surface at a height of $1.25\ m$ above the ground to indicate the desired mounting location of the sensor. The PX4Flow module was attached underneath the I-BC and connected to the Pixhawk FCU using an I^2C serial connection, and the LIDAR-Lite v3 was connected using a PWM interface. The FCU natively supports these sensors as inputs, and performs onboard position estimation using an extended Kalman filter algorithm [111]. The position estimator algorithm is referred to as EKF2 in the FCU's PX4 firmware, and is labeled as such in the following sections. For safety in the experiments, the boom-prop throttle was set to $\approx 30\%$ during the `APPROACHING_TARGET` state.

UAV Pose Estimation Performance

With this setup, 12 autonomous sensor mounting flight tests were performed. Out of these 12 tests, 11 were completely successful, and 1 mounted the sensor to the surface, but failed to toggle the PTRM. This is likely due to the relatively low boom-prop throttle used for safety in the experiments, which can lead to a low impact velocity. To prevent this from happening in future flights we slightly increased the throttle of the boom-prop to 37%. Later, we also used a dry lubricant to reduce the increased friction between the PTRM slider and slide rail that can occur with off-angle impacts. Fig. 5.9 shows the X-Y position and yaw orientation of the I-BC during a successful flight. In the portion of the flight shown, the RMSE between the onboard EKF2 position estimation and Vicon was 10.1 *cm*. Over all of the flight tests, the average RMSE was ≈ 9.3 *cm*, and the maximum and minimum RMSE were 22.5 *cm* and 5.4 *cm*, respectively. And, for the yaw angle, over all of the flight tests the average RMSE was $1.0°$, and the maximum and minimum RMSE were $2.1°$ and $0.3°$, respectively.

Fig. 5.10 shows the EKF2 trajectory of the I-BC, along with the position setpoints calculated during the autonomous sensor mounting process. Each color of the trajectory represents a different state of the autonomous flight controller, as depicted in Fig. 3.6. Similarly, Fig. 5.11 shows the yaw angle and setpoints during the various stages of a sensor mounting test.

Sensor Placement Performance

Fig. 5.12 shows the absolute error in the sensor mounting location for all 11 successful flights compared to the center of the red square outline (measured with Vicon). The average placement error over all these flight tests was 9.6 *cm*, and the maximum and minimum errors were 22.0 *cm* and 1.2 *cm*, respectively. These errors matched our expectations based on our

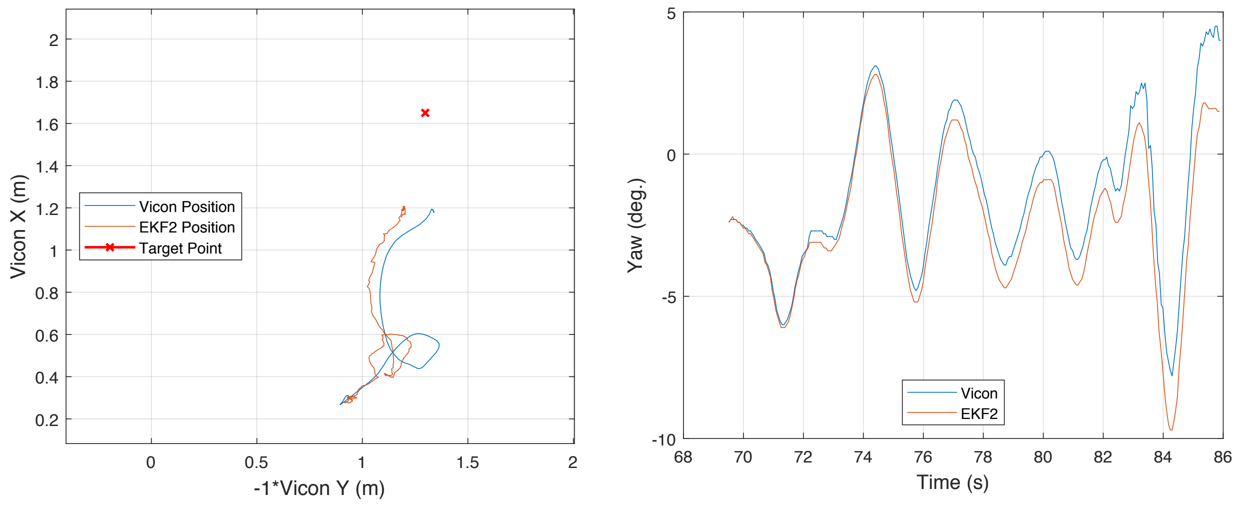


Figure 5.9. : Onboard position estimation (EKF2) vs. ground truth (Vicon). Left: X-Y trajectory of I-BC during a successful sensor mounting trial (top view); right: I-BC yaw orientation during a successful sensor mounting trial.

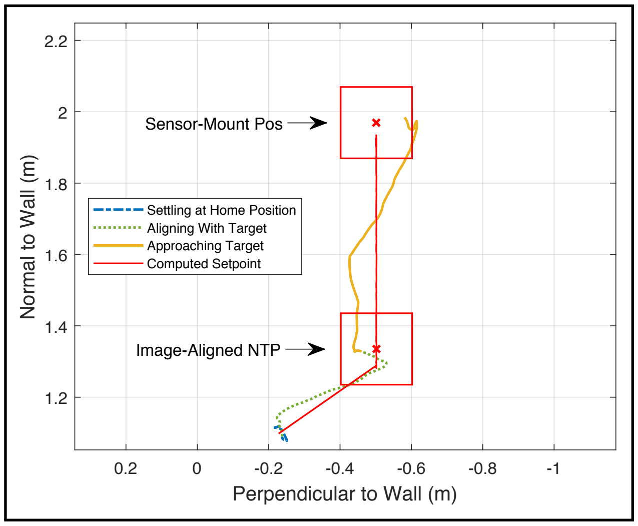


Figure 5.10. : Top view of I-BC trajectory during an indoor sensor mounting task.

initial expected UAV position estimation error of $\approx 10\ cm$, and the average target pose estimation error of $2.4\ cm$.

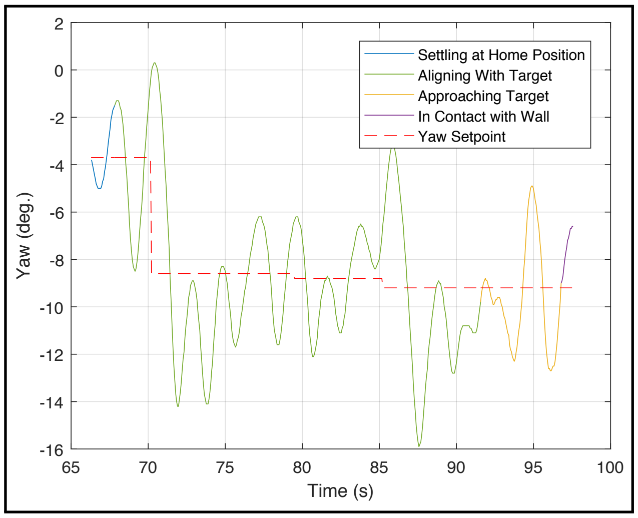


Figure 5.11. : I-BC yaw angle vs. setpoint during a successful sensor mounting test.

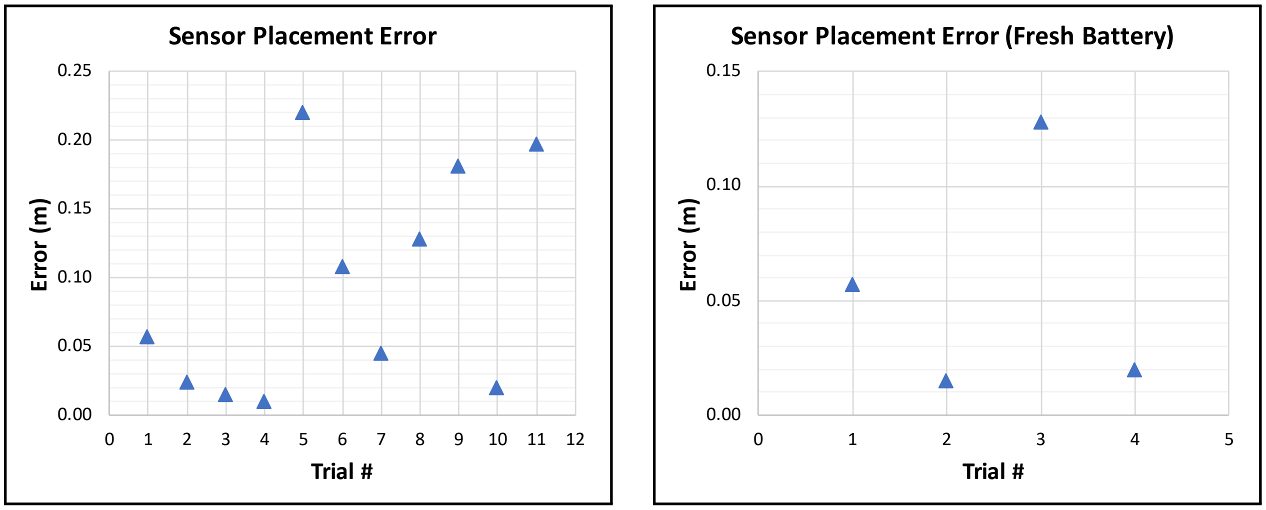


Figure 5.12. : Left: sensor placement error for 11 autonomous mounting tests; right: sensor placement error for tests with a fully charged battery.

A Note on the Effect of Battery Life on Performance

It was noted through the course of the flight tests that the remaining capacity of the battery, and also the age and condition of the battery itself seemed to be directly correlated with the sensor placement performance. This was confirmed by comparing the average sensor placement error for all flight tests with only those tests that were performed on a full charge (see Fig. 5.12, right). In this comparison, flights using one particular battery which consistently performed more poorly than the other batteries were also excluded. For the remaining flights shown, the average sensor placement error was 5.5 *cm*, 43% less than the overall average of 9.6 *cm*. In [112], a similar characteristic of LiPo batteries was noted, suggesting that, in general, a decrease of 10-20% in battery capacity is common with an increase in the power consumed.

5.4.2 Outdoor Flight Tests

Several outdoor flight tests were performed with the I-BC to consider the effects of disturbances such as wind and bright sunlight. Considering the large range of potential applications for outdoor remote sensor mounting capabilities in infrastructure inspection and maintenance applications, a sensor mounting task on the side of a steel bridge structure was considered.

Setup

All flight tests were performed at Purdues Steel Bridge Research, Inspection, Training and Engineering (S-BRITE) Center, which contains several full-scale bridge structures and individual components in its Bridge Component Gallery [113] (see Fig. 5.13). Specifically, a small piece of black tape was used to indicate a target mounting point on a steel girder from the Virginia Avenue Bridge (see Fig. 5.14, left). An Extech hot wire anemometer mounted near the flight test area was used to measure wind conditions during the test flights (see Fig. 5.14, right). Additionally, the tests were performed on a clear day to consider the effects of the bright sunlight on the performance of the Intel D435 (which uses an IR projector to improve its stereo depth measurements).

In preliminary tests, the two-sided adhesive used in the indoor experiments was effective in attaching the payload to the dusty bridge surface. However, since the tape adhesion method had already been validated in previous experiments, we used a 1-1/4$''$ neodymium magnet for the sensor mounting tests on the bridge in order to validate an alternative method for adhesion. The magnet was attached directly to the back of the payload using two-sided tape.

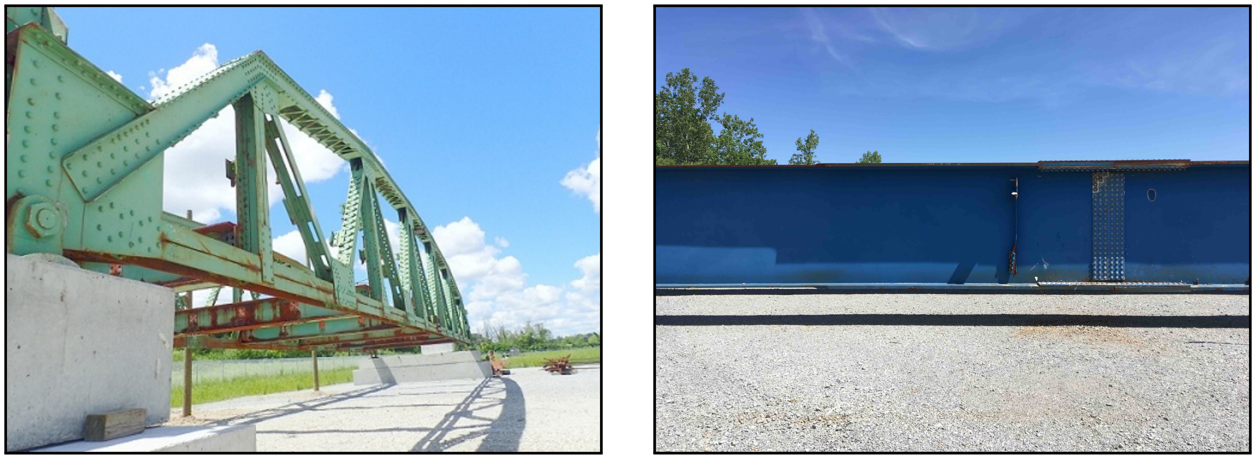


Figure 5.13. : Bridge components at Purdue's S-BRITE Bridge Component Gallery [113]. Left: Indian Trail Road Bridge (91-ft. Pony Truss); right: Virginia Avenue Bridge (155-ft. Main Span Girder Bridge).

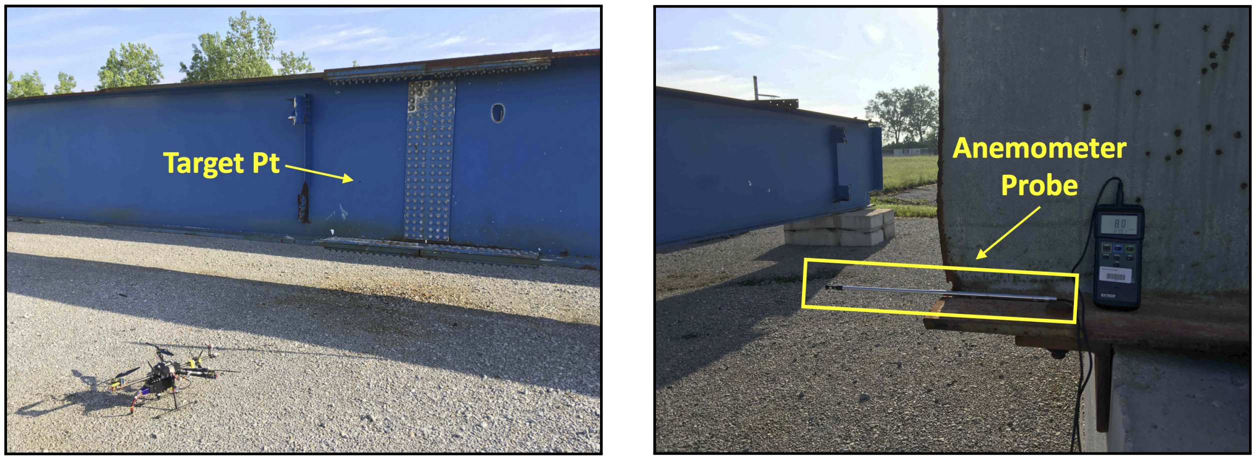


Figure 5.14. : Setup for outdoor autonomous sensor mounting flight tests at S-BRITE. Left: target point for sensor mounting; right: hot-wire anemometer setup to measure wind.

Finally, a WiFi router was set up near the flight test area to connect to the I-BC's onboard computer and view the video stream from the onboard camera. Fig. 5.15 shows the FTT GUI being used for target point selection during a sensor mounting flight test.

With this setup, a total of 8 autonomous sensor mounting flight tests were performed. The cross wind conditions for most flight tests fluctuated between ≈ 0 - 3 mph, while the tail wind fluctuated between ≈ 0 - 10 mph. The I-BC successfully mounted the sensor to the bridge surface in all 8 tests, but a gust of wind (10+ mph) in one test caused the I-BC to translate to the right just before impact, preventing the PTRM from toggling to release the payload. As a failsafe, the PTRM slider was allowed to slide off of the slide rail (see Fig. 3.2a), leaving the sensor attached to the surface.

Figure 5.16 shows the EKF2 trajectory and computed position setpoints from a successful outdoor sensor mounting flight test, and Fig. 5.17 shows the yaw angle during the various stages of an outdoor sensor mounting test. These plots are representative of the I-BC's performance in each of the outdoor flight tests, and demonstrate that the I-BC is able to achieve similar position and yaw angle tracking to the indoor tests in an outdoor setting.

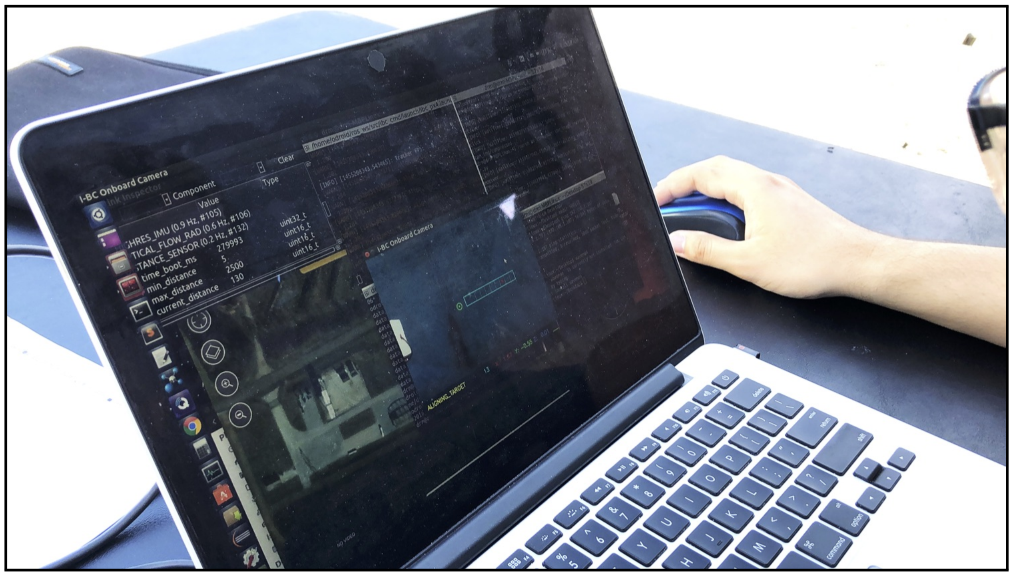


Figure 5.15. : Target point selection using FTT GUI during an outdoor sensor mounting flight test.

Fig. 5.18 shows the sensor placement error for all 8 outdoor trials performed compared to the location of the black tape marker. The average placement error over these tests was 17.3 cm, and the maximum and minimum errors were 33.2 cm and 5.9 cm, respectively. As with the indoor flight tests, the average sensor placement error for all flight tests was

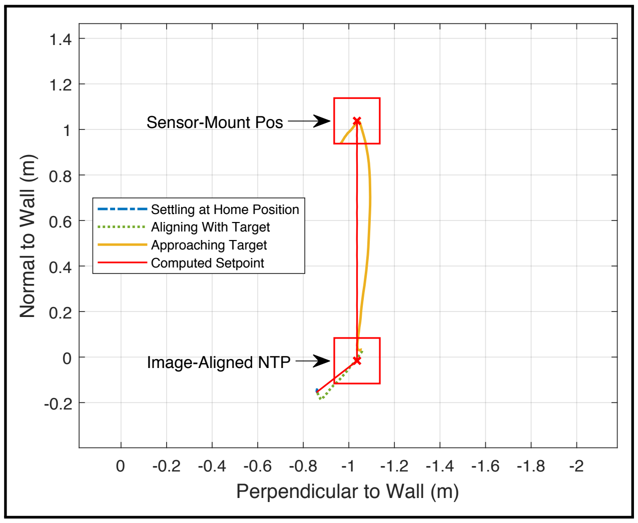


Figure 5.16. : Top view of I-BC trajectory during an outdoor sensor mounting task.

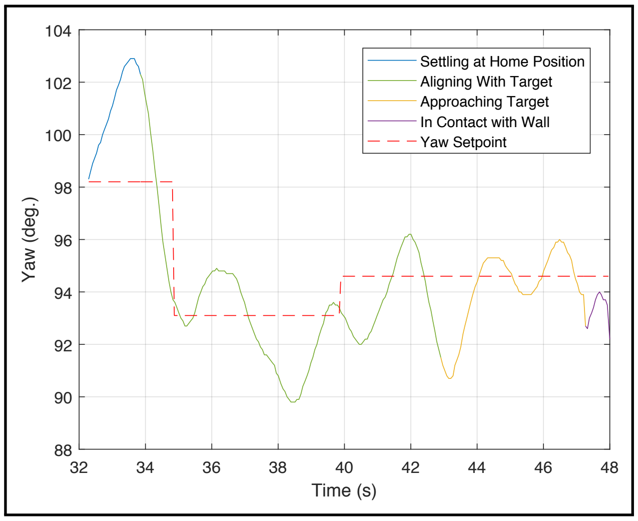
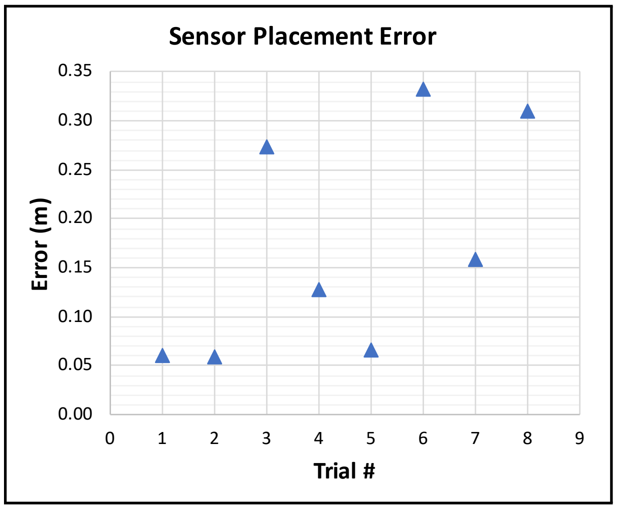


Figure 5.17. : I-BC yaw angle vs. setpoint during a successful sensor mounting test (outdoors).

compared with the average error for only those tests that were performed on a full charge and excluding the same battery mentioned in the discussion on indoor flight performance above (see Fig. 5.18, right). For the remaining flight tests shown, the average sensor placement error was 11.5 *cm*, 34% lower than the overall average of 17.3 *cm*. These results further indicate a correlation between decreased battery life/quality and decreased sensor placement accuracy.



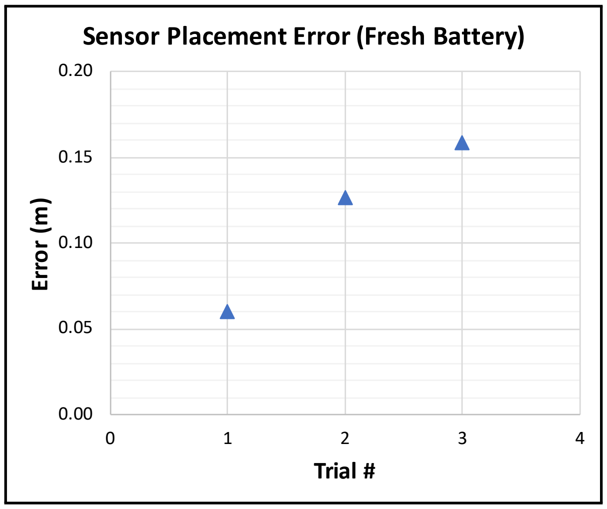


Figure 5.18. : Left: sensor placement error for 8 autonomous mounting tests (outdoors); right: sensor placement error for tests with a fully charged battery.

5.4.3 Bounds on Sensor Placement Accuracy

The accuracy of the sensor placement in the autonomous flight tests described above is affected by several potential sources of error, including UAV pose estimation errors, target pose estimation errors, and the UAV's pose tracking errors (influenced by the UAV's flight dynamics, wind disturbances, etc.). The performance of the system as a whole depends on how these errors either offset or combine together to decrease or increase the total error.

Some aspects of the FTT method presented above were included to help mitigate the effects of these errors. For instance, to compensate for drift in the UAV pose estimate, the user is able to re-click on the desired target mounting point, which causes a recalculation of the 3D target point relative to the **E**-frame (${}^{E}\mathbf{P}_t$), effectively resetting any drift in the UAV pose estimate. This recalculation helps to limit the effect of the accumulated UAV pose estimation errors relative to the real world. As described in section 5.2.1 above, the user is aided in the re-clicking process by the continuous updates of the alignment error calculated based on ${}^{C}\mathbf{P}_e$ (with visual feedback displayed in the GUI as shown in Fig. 5.4). Additionally, if alignment errors are detected above an acceptable threshold determined for the given manipulation task, the UAV's position setpoint relative to the target point will automatically be recalculated to ensure close alignment between $\mathbf{P}_e$ and $\mathbf{P}_{ct}$ (depicted as a green circle and red cross in Fig. 5.4, respectively).

The accuracy of the target pose estimation is presented in section 5.3 (see Figs. 5.7 and 5.8), and the accuracy of the UAV pose estimation is presented with RMSE metrics in section 5.4 (see Fig. 5.9). As a representation of the accuracy of the UAV position tracking, we considered the I-BC's position-tracking error while hovering in a position-controlled flight mode during a window of ≈ 20 *sec.* prior to each of the autonomous sensor mounting tasks (both indoor and outdoor) described in section 5.4 above. Across all of these tests, the minimum, maximum and average RMSE between the I-BC's commanded and actual position (based on the onboard EKF2 estimate) were: 4.0 *cm*, 9.7 *cm*, and 7.0 *cm*, respectively.

Considering these sources of error, the lower bound in sensor placement error is zero, occurring when the estimation and positioning errors are at a minimum or when they offset. And, based on the flight test results presented in section 5.4, the upper bound on the sensor placement error is ≈ 36.2 *cm*. This upper bound is based on the worst-case combination of a maximum position estimate RMSE of 22.5 *cm*, a maximum target position estimate error of 4.0 *cm*, and a maximum position-tracking error of 9.7 *cm*.

As expected, the sensor placement errors reported from the flight tests described above are all below the estimated upper bound. Furthermore, the average indoor sensor placement error (9.6 *cm*) is only $\approx 25\%$ of the upper bound, and the average outdoor sensor placement error (17.3 *cm*) is $\approx 48\%$ of the upper bound. Thus, using the combined position estimation and target tracking approaches described in sections 5.1 - 5.3, the I-BC is able to repeatably perform the autonomous sensor mounting task, and is suitable for applications where a maximum sensor placement error of $\approx$ 36.2 *cm* is acceptable. In addition, other aerial manipulation tasks, such as the door opening task described in chapter 4, may also be performed in unstructured environments using this localization approach, so long as sufficient mechanisms are provided to compensate for the estimation and positioning errors (such as automatic recovery in the case of failed attempts, etc. see section 4.2.4 above).

5.5 Conclusions

In this chapter, we presented a target tracking method (FTT) that is completely independent of traditional visual tracking features (e.g. those obtained through the use of SIFT, SURF, etc.), but instead accepts input from a human user. This method was implemented on the I-BC, in combination with the OF-based PX4Flow module which was used for position estimation. This combined approach has several advantages, including: 1) it allows tracking of target points on featureless surfaces (e.g. a wide, smooth portion of a bridge truss, such as the left side of the truss shown in Fig. 5.13, left), 2) it is computationally inexpensive, allowing for the implementation of other useful features like obstacle avoidance and autonomous control on the onboard computer, 3) it is does not suffer from tracking loss when the target point moves out of the FOV, and 4) since the FTT method tracks the target point in the coordinate frame of the UAV's estimator, any improvements introduced in the UAV pose estimation will directly translate to improved target tracking performance. Furthermore, in scenarios where it is desirable or even necessary to track a target using visual features (such as routine inspection or maintenance of a specific component), this method could be used to help with initial detection or recovery if tracking is lost.

Several indoor flight tests demonstrated the I-BC's ability to perform autonomous sensor mounting tasks accurately and repeatably using the combined position estimation and target tracking approaches described in this chapter. Additionally, several outdoor flight tests demonstrated that this same setup is effective in unstructured environments, and in the presence of light wind disturbances and bright sunlight.

6. CONCLUSIONS

In this work, the I-BC was presented as a novel platform for aerial manipulation in an effort to solve the complete autonomous aerial manipulation problem, where both the UAV localization and visual target tracking are performed onboard in unstructured, real-world environments. Below is a review of the main contributions of this work, followed by a discussion of potential areas for future work.

6.1 Contributions

Design of a Novel UAV Platform for Aerial Manipulation

As discussed in section 1.2, UAV design for aerial manipulation typically involves adding to or changing the configuration of the UAV's primary actuators to increase the number of independently controllable DOF, and/or adding a robotic arm with multiple DOF to augment the UAV's total number of independently controllable DOF. However, adding these additional actuators reduces the payload capacity of the UAV and reduces the total flight time, thus limiting the UAV's utility for deployment in large-scale, real-world operations. In this work, the I-BC was fitted with only one additional actuator on its forward-facing boom, the boom-prop, to provide an additional horizontal DOF, while minimizing the total weight and complexity added to the vehicle. The boom-prop's custom-made symmetric propellers allow for the generation of both forward and reverse thrust, to enable a broader range of manipulation capabilities. With this software-reversible boom-prop, the I-BC is able to fly forward/backward and apply pushing/pulling forces without pitching the airframe. This allows the I-BC to remain in a stable, hovering configuration while performing aerial manipulation tasks. Additionally, as a tricopter, the I-BC has more spacing between the front propellers to accommodate the boom-prop and custom end-effectors, and larger, more energy-efficient propeller blades can be used as the primary source of lift.

Design of Modular End-effectors with Integrated Force and Distance Sensing

In chapters 3 and 4, modular, 3D-printed end-effectors with integrated force and distance sensors were presented that enable autonomous aerial interactions with the environment. For the remote sensor mounting task in chapter 3, a lightweight, passive PTRM end-effector was designed, and several experimental flight tests demonstrated its ability to reliably deliver sensor payloads without the need of an actuated mechanism. For the more complex task of opening an enclosure door discussed in chapter 4, a lightweight, compliant, three-finger gripper was designed that can conform to a variety of shapes. The gripper utilizes a single DC motor attached to a 3D printed linear actuation mechanism to reliably grasp objects, and the entire mechanism slides between two forces sensors to enable both pushing and pulling force feedback during aerial manipulation tasks. Several experimental flight tests demonstrated the effectiveness of the gripper design in compensating for horizontal misalignment and grasping a door handle to open a small enclosure door.

Strategy for Featureless Visual Target Tracking (FTT)

In chapter 5, a featureless visual target tracking method (FTT) was introduced that enables the I-BC to perform aerial manipulation tasks without the need of artificial markers or beacons placed near the target object, and also allows target points to be selected by a human operator. As described in section 5.5, combining the FTT method with the UAV's onboard pose estimator has several advantages, including: 1) it allows tracking of target points on featureless surfaces, 2) it is computationally inexpensive, allowing for the implementation of other useful features like obstacle avoidance and autonomous control on the onboard computer, 3) it is does not suffer from tracking loss when the target point moves out of the FOV, and 4) since the FTT method tracks the target point in the coordinate frame of the UAV's estimator, any improvements introduced in the UAV pose estimation will directly translate to improved target tracking performance.

Realization of New Autonomous Aerial Manipulation Capabilities in Unstructured Environments

Several flight tests conducted in both indoor and outdoor experiments demonstrated that the I-BC, using only onboard computation, can successfully perform autonomous sensor mounting tasks in unstructured environments, even in the presence of wind disturbances. These capabilities have potential applications in areas such as large-scale industrial and infrastructure inspection and maintenance, precision agriculture, and nuclear decontamination efforts. Moreover, these results were achieved using low-cost actuators and sensors, and open-source flight control hardware and software (e.g. the Pixhawk FCU and PX4Flow OF module), thus making them more feasible for implementation at scale.

6.2 Outlook

To improve and extend the capabilities of the I-BC for application in a wider range of autonomous aerial manipulation scenarios, the following areas may be considered in future work.

Contact Inspection Tasks

As aerial manipulation capabilities have developed in recent years, growing interest has been shown in the use of UAVs for performing contact inspection tasks using ultrasonic NDT sensors for large-scale industrial and infrastructure inspection. The I-BC's ability to apply horizontal forces to vertical surfaces makes it ideally suited to perform this type of task. Considering the similarity between this task and the remote sensor mounting task, it is likely that autonomous contact inspection tasks could be achieved using the I-BC with the simple addition of a new modular end-effector designed specifically for these tasks. Such an end-effector could employ the use of lightweight wheels or bearings to allow the end-effector to roll along the surface as needed during the inspection. In these cases, a damping mechanism may also be designed to prevent the end-effector from bouncing away from the surface after initial contact.

Obstacle Avoidance

Several safety features were built into the I-BC's autonomous flight controller to prevent inadvertant collisions with the vertical surface of interest during the sensor mounting and door opening tasks considered in this work. However, additional side-to-side obstacle avoidance, and forward obstacle avoidance while in manual flight modes would further increase the I-BC's safety and ease of use for less experienced pilots. This feature would be particularly beneficial in applications where the I-BC is flying through cluttered environments and operating at a large distance from the pilot or at a difficult vantage point.

Code Optimization and Mass Reduction

Now that the localization and autonomous control algorithms for the I-BC have been validated, certain portions of the source code could be optimized to speed up their performance. For example, in the FTT GUI, image frames from the D435 could be pre-processed in a separate thread to speed up performance. The weight of the current I-BC prototype could also be reduced by using shorter, lighter cables between electronic components, using a custom PCB (instead of breadboard) for the Device Manager's connections, and reducing the size of various 3D-printed components such as the battery compartment and camera mount. These weight reductions and code optimizations will result in even longer flight times for performing aerial manipulation tasks, and allow for further experimentation with new algorithms and end-effectors.

REFERENCES

REFERENCES

[1] S. Bouabdallah. *Design and Control of Quadrotors With Application To Autonomous Flying*. PhD thesis, École Polytechnique Fédérale de Lausanne, 2007.

[2] P. Pounds, R. Mahony, and P. Corke. Modelling and control of a large quadrotor robot. *Control Engineering Practice*, 18(7):691–699, 2010.

[3] D. Schafroth, C. Bermes, S. Bouabdallah, and R. Siegwart. Modeling, system identification and robust control of a coaxial micro helicopter. *Control Engineering Practice*, 18(7):700–711, 2010.

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

[5] F. J. Perez-Grau, R. Ragel, F. Caballero, A. Viguria, and A. Ollero. Semi-autonomous teleoperation of UAVs in search and rescue scenarios. In *IEEE International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1066–1074, 2017.

[6] Q. Lindsey, D. Mellinger, and V. Kumar. Construction of Cubic Structures with Quadrotor Teams. *Robotics: Science and Systems*, 7:177–184, 2012.

[7] M. Kaamin, S. N. M. Razali, N. F. A. Ahmad, S. M. Bukari, N. Ngadiman, A. A. Kadir, and N. B. Hamid. The application of micro UAV in construction project. In *AIP Conference Proceedings*, volume 1891, page 020070, 2017.

[8] L. Marconi, F. Basile, G. Caprari, R. Carloni, P. Chiacchio, C. Hurzeler, V. Lippiello, R. Naldi, J. Nikolic, B. Siciliano, S. Stramigioli, and E. Zwicker. Aerial service robotics: The AIRobots perspective. In *International Conference on Applied Robotics for the Power Industry (CARPI)*, pages 64–69, 2012.

[9] P. J. Sanchez-Cuevas, P. Ramon-Soria, B. Arrue, A. Ollero, and G. Heredia. Robotic System for Inspection by Contact of Bridge Beams Using UAVs . *Sensors*, 19(2):305, 2019.

[10] M. Á. Trujillo, J. R. Martínez-De Dios, C. Martín, A. Viguria, and A. Ollero. Novel aerial manipulator for accurate and robust industrial NDT contact inspection: A new tool for the oil and gas inspection industry. *Sensors*, 19(6):1305, 2019.

[11] G. Jiang and R. Voyles. Hexrotor UAV platform enabling dextrous interaction with structures-flight test. In *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2013.

[12] G. Jiang, R. Voyles, K. Sebesta, and H. Greiner. Estimation and optimization of fully-actuated multirotor platform with nonparallel actuation mechanism. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6843–6848, 2017.

[13] Y. Tadokoro, T. Ibuki, and M. Sampei. Maneuverability Analysis of a Fully-Actuated Hexrotor UAV Considering Tilt Angles and Arrangement of Rotors. *IFAC-PapersOnLine*, 50(1):8981–8986, 2017.

[14] S. Rajappa, M. Ryll, H. H. Bulthoff, and A. Franchi. Modeling , Control and Design Optimization for a Fully-actuated Hexarotor Aerial Vehicle with Tilted Propellers. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4006–4013, 2015.

[15] E. Kaufman, K. Caldwell, D. Lee, and T. Lee. Design and development of a free-floating hexrotor UAV for 6-DOF maneuvers. In *IEEE Aerospace Conference*, pages 1–10, 2014.

[16] C. Papachristos, K. Alexis, and A. Tzes. Efficient Force Exertion for Aerial Robotic Manipulation : Exploiting the Thrust Vectoring Authority of a Tri TiltRotor UAV. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4500–4505, 2014.

[17] P. Segui-Gasco, Y. Al-Rihani, H. S. Shin, and A. Savvaris. A novel actuation concept for a multi rotor UAV. *Journal of Intelligent and Robotic Systems*, 74(1):173–191, 2014.

[18] M. Ryll, H. H. Bulthoff, and P. R. Giordano. A Novel Overactuated Quadrotor Unmanned Aerial Vehicle: Modeling, Control, and Experimental Validation. *IEEE Transactions on Control Systems Technology*, 23(2):540–556, 2015.

[19] A. Oosedo, S. Abiko, S. Narasaki, A. Kuno, A. Konno, and M. Uchiyama. Flight control systems of a quad tilt rotor Unmanned Aerial Vehicle for a large attitude change. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2326–2331, 2015.

[20] M. Ryll, G. Muscio, F. Pierri, E. Cataldi, G. Antonelli, F. Caccavale, and A. Franchi. 6D physical interaction with a fully actuated aerial robot. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5190–5195, 2017.

[21] Y. Long, L. Wang, and D. J. Cappelleri. Modeling and global trajectory tracking control for an over-actuated MAV. *Advanced Robotics*, 28(3):145–155, 2014.

[22] Y. Long and D. J. Cappelleri. Omnicopter: A novel overactuated micro aerial vehicle. In *Advances in Mechanisms, Robotics and Design Education and Research*, volume 14, pages 215–226. Springer, Heidelberg, 2013.

[23] S. Park, J. Her, J. Kim, and D. Lee. Design, modeling and control of omni-directional aerial robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1570–1575, 2016.

[24] A. Nikou, G. C. Gavridis, and K. J. Kyriakopoulos. Mechanical Design, Modelling and Control of a Novel Aerial Manipulator. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4698–4703, 2015.

[25] D. Brescianini and R. D. Andrea. Design, Modeling and Control of an Omni-Directional Aerial Vehicle. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3261–3266, 2016.

[26] A. Keemink, M. Fumagalli, S. Stramigioli, and R. Carloni. Mechanical design of a manipulation system for unmanned aerial vehicles. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3147–3152, 2012.

[27] M. Fumagalli, R. Naldi, A. MacChelli, R. Carloni, S. Stramigioli, and L. Marconi. Modeling and control of a flying robot for contact inspection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3532–3537, 2012.

[28] F. Forte, R. Naldi, A. Macchelli, and L. Marconi. On the control of an aerial manipulator interacting with the environment. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4487–4492, 2014.

[29] K. Kondak, K. Krieger, A. Albu-Schaeffer, M. Schwarzbach, M. Laiacker, I. Maza, A. Rodriguez-Castano, and A. Ollero. Closed-loop behavior of an autonomous helicopter equipped with a robotic arm for aerial manipulation tasks. *International Journal of Advanced Robotic Systems*, 10(2), 2013.

[30] F. Huber, K. Kondak, K. Krieger, D. Sommer, M. Schwarzbach, M. Laiacker, I. Kossyk, S. Parusel, S. Haddadin, and A. Albu-Schaffer. First analysis and experiments in aerial manipulation using fully actuated redundant robot arm. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3452–3457, 2013.

[31] C. D. Bellicoso, L. R. Buonocore, V. Lippiello, and B. Siciliano. Design, modeling and control of a 5-DoF light-weight robot arm for aerial manipulation. In *Mediterranean Conference on Control and Automation (MED)*, pages 853–858, 2015.

[32] M. Kamel, K. Alexis, and R. Siegwart. Design and modeling of dexterous aerial manipulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4870–4876, 2016.

[33] A. Suarez, G. Heredia, and A. Ollero. Lightweight compliant arm with compliant finger for aerial manipulation and inspection. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4449–4454, 2016.

[34] P. E. Pounds, D. R. Bersak, and A. M. Dollar. Grasping from the air: Hovering capture and load stability. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2491–2498, 2011.

[35] P. E. Pounds, D. R. Bersak, and A. M. Dollar. Stability of small-scale UAV helicopters and quadrotors with added payload mass under PID control. *Autonomous Robots*, 33(1):129–142, 2012.

[36] D. Mellinger, Q. Lindsey, M. Shomin, and V. Kumar. Design, modeling, estimation and control for aerial grasping and manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2668–2673, 2011.

[37] V. Ghadiok, J. Goldin, and W. Ren. Autonomous indoor aerial gripping using a quadrotor. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4645–4651, 2011.

[38] A. Gawel, M. Kamel, T. Novkovic, J. Widauer, D. Schindler, B. P. Von Altishofen, R. Siegwart, and J. Nieto. Aerial picking and delivery of magnetic objects with MAVs. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5746–5752, 2017.

[39] M. Stokkeland, K. Klausen, and T. A. Johansen. Autonomous visual navigation of Unmanned Aerial Vehicle for wind turbine inspection. In *International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 998–1007. IEEE, 2015.

[40] R. A. Mattar. Development of a Wall-Sticking Drone for Non-Destructive Ultrasonic and Corrosion Testing. *Drones*, 2(1):8, 2018.

[41] L. Marconi, R. Naldi, and L. Gentili. Modelling and control of a flying robot interacting with the environment. *Automatica*, 47(12):2571–2583, 2011.

[42] K. Alexis, C. Huerzeler, and R. Siegwart. Hybrid modeling and control of a coaxial unmanned rotorcraft interacting with its environment through contact. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5417–5424, 2013.

[43] T. Bartelds, A. Capra, S. Hamaza, S. Stramigioli, and M. Fumagalli. Compliant Aerial Manipulators: Toward a New Generation of Aerial Robotic Workers. *IEEE Robotics and Automation Letters*, 1(1):477–483, 2016.

[44] A. Albers, S. Trautmann, T. Howard, T. A. Nguyen, M. Frietsch, and C. Sauter. Semi-autonomous flying robot for physical interaction with environment. In *IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 441–446, 2010.

[45] H. W. Wopereis, J. J. Hoekstra, T. H. Post, G. A. Folkertsma, S. Stramigioli, and M. Fumagalli. Application of substantial and sustained force to vertical surfaces using a quadrotor. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2704–2709, 2017.

[46] S. Shimahara, R. Ladig, L. Suphachart, S. Hirai, and K. Shimonomura. Aerial manipulation for the workspace above the airframe. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1453–1458, 2015.

[47] H. W. Wopereis, T. D. Van Der Molen, T. H. Post, S. Stramigioli, and M. Fumagalli. Mechanism for perching on smooth surfaces using aerial impacts. In *IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, pages 154–159, 2016.

[48] H. Tsukagoshi, M. Watanabe, T. Hamada, D. Ashlih, and R. Iizuka. Aerial manipulator with perching and door-opening capability. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4663–4668, 2015.

[49] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. Mobile Robot Localization. In *Introduction to Autonomous Mobile Robots*, pages 265–367. MIT Press, 2nd edition, 2011.

[50] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping (SLAM): Part I. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006.

[51] T. Bailey and H. Durrant-Whyte. Simultaneous Localization and Mapping (SLAM): Part II. *IEEE Robotics and Automation Magazine*, 13(September):108–117, 2006.

[52] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–659, 2004.

[53] D. Scaramuzza and F. Fraundorfer. Tutorial: Visual odometry. *IEEE Robotics and Automation Magazine*, 18(4):80–92, 2011.

[54] F. Fraundorfer and D. Scaramuzza. Visual odometry: Part II: Matching, robustness, optimization, and applications. *IEEE Robotics and Automation Magazine*, 19(2):78–90, 2012.

[55] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad. An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics. *Intelligent Industrial Systems*, 1(4):289–311, 2015.

[56] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[57] W. Hess, D. Kohler, H. Rapp, and D. Andor. Real-time loop closure in 2D LIDAR SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278. IEEE, 2016.

[58] D. Scaramuzza, M. C. Achtelik, L. Doitsidis, F. Friedrich, E. Kosmatopoulos, A. Martinelli, M. W. Achtelik, M. Chli, S. Chatzichristofis, L. Kneip, D. Gurdan, L. Heng, G. H. Lee, S. Lynen, M. Pollefeys, A. Renzaglia, R. Siegwart, J. C. Stumpf, P. Tanskanen, C. Troiani, S. Weiss, and L. Meier. Vision-controlled micro flying robots: From system design to autonomous navigation and mapping in GPS-denied environments. *IEEE Robotics and Automation Magazine*, 21(3):26–40, 2014.

[59] A. Davison. Real-time simultaneous localisation and mapping with a single camera. In *IEEE International Conference on Computer Vision*, pages 1403–1410, 2003.

[60] B. K. Horn and B. G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, 1981.

[61] A. Briod, J.-C. Zufferey, and D. Floreano. Optic-Flow Based Control of a 46g Quadrotor. In *Workshop on Vision-based Closed-Loop Control and Navigation of Micro Helicopters in GPS-denied Environments, IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[62] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart. Robust visual inertial odometry using a direct EKF-based approach. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 298–304. IEEE, 2015.

[63] I. Sa, M. Kamel, M. Burri, M. Bloesch, R. Khanna, M. Popovic, J. Nieto, and R. Siegwart. Build your own visual-inertial drone: A cost-effective and open-source autonomous drone. *IEEE Robotics and Automation Magazine*, 25(1):89–103, 2018.

[64] G. Klein and D. Murray. Parallel Tracking and Mapping for Small ARWorkspaces. In *6th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 225–234, 2007.

[65] S. Shen, N. Michael, and V. Kumar. Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 20–25, 2011.

[66] Vicon motion capture - cameras, devices and software. `https://www.vicon.com/motion-capture`. Accessed: 2019-06-12.

[67] F. Kendoul, I. Fantoni, and K. Nonami. Optic flow-based vision system for autonomous 3D localization and control of small aerial vehicles. *Robotics and Autonomous Systems*, 57(6):591–602, 2009.

[68] D. Honegger, L. Meier, P. Tanskanen, and M. Pollefeys. An open source and open hardware embedded metric optical flow CMOS camera for indoor and outdoor applications. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1736–1741. IEEE, 2013.

[69] J. Delmerico and D. Scaramuzza. A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2502–2509, 2018.

[70] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart. The EuRoC micro aerial vehicle datasets. *International Journal of Robotics Research*, 35(10):1157–1163, 2016.

[71] Multistation - leica geosystems. `https://leica-geosystems.com/en-us/products/total-stations/multistation`. Accessed: 2019-06-21.

[72] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 15–22, 2014.

[73] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale. Keyframe-based visual-inertial odometry using nonlinear optimization. *International Journal of Robotics Research*, 34(3):314–334, 2015.

[74] T. Qin, P. Li, and S. Shen. VINS-Mono : A Robust and Versatile Monocular Visual-Inertial State Estimator. *IEEE Transactions on Robotics*, 34(4):1–14, 2018.

[75] M. Labbé and F. Michaud. RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36(2):416–446, 2019.

[76] H. Yang and D. Lee. Dynamics and control of quadrotor with robotic manipulator. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5544–5549, 2014.

[77] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[78] P. Li, D. Wang, L. Wang, and H. Lu. Deep visual tracking: Review and experimental comparison. *Pattern Recognition*, 76:323–338, 2018.

[79] M. Fiaz, A. Mahmood, and S. K. Jung. Tracking Noisy Targets: A Review of Recent Object Tracking Approaches. Technical report, Kyungpook National University, 2018.

[80] P. Janku, K. Koplik, T. Dulik, and I. Szabo. Comparison of tracking algorithms implemented in OpenCV. *MATEC Web of Conferences*, 76:04031, 2016.

[81] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

[82] F. Chaumette and S. Hutchinson. Visual servo control. I. Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4):82–90, 2006.

[83] M. Laiacker, M. Schwarzbach, and K. Kondak. Automatic aerial retrieval of a mobile robot using optical target tracking and localization. In *IEEE Aerospace Conference Proceedings*, volume 2015-June, pages 1–7. IEEE, 2015.

[84] D. Wagner and D. Schmalstieg. ARToolKitPlus for Pose Tracking on Mobile Devices ARToolKit. In *Computer Vision Winter Workshop (CVWW)*, pages 139–146, 2007.

[85] S. Kim, H. Seo, S. Choi, and H. J. Kim. Vision-guided aerial manipulation using a multirotor with a robotic arm. *IEEE/ASME Transactions on Mechatronics*, 21(4):1912–1923, 2016.

[86] R. Mebarki, V. Lippiello, and B. Siciliano. Toward image-based visual servoing for cooperative aerial manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6074–6080, 2015.

[87] J. Thomas, G. Loianno, K. Sreenath, and V. Kumar. Toward image based visual servoing for aerial grasping and perching. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2113–2118. IEEE, 2014.

[88] C. Kanellakis, M. Terreran, D. Kominiak, and G. Nikolakopoulos. On vision enabled aerial manipulation for multirotors. In *IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–7, 2017.

[89] P. R. Soria, B. C. Arrue, and A. Ollero. Detection, location and grasping objects using a stereo sensor on UAV in outdoor environments. *Sensors*, 17(1):103, 2017.

[90] G. Brunner, B. Szebedy, S. Tanner, and R. Wattenhofer. The Urban Last Mile Problem: Autonomous Drone Delivery to Your Balcony. Technical report, ETH Zurich, 2018.

[91] J. Wang and E. Olson. AprilTag 2: Efficient and robust fiducial detection. In *IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 4193–4198. IEEE, 2016.

[92] D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Design of the I-BoomCopter UAV for environmental interaction. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5209–5214, 2017.

[93] L. Meier, P. Tanskanen, L. Heng, G. H. Lee, F. Fraundorfer, and M. Pollefeys. PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision. *Autonomous Robots*, 33(1):21–39, 2012.

[94] I. Davidson. The boomcopter: Not what you first think... ***https://youtu.be/VKVpXCNZimc***, 2014.

[95] D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Design of the Interacting-BoomCopter Unmanned Aerial Vehicle for Remote Sensor Mounting. *Journal of Mechanisms and Robotics*, 10(2):025001–025001–8, 2018.

[96] D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Autonomous control of the interacting-boomcopter UAV for remote sensor mounting. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5219–5224, 2018.

[97] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng. ROS: an open-source Robot Operating System. In *ICRA Open-Source Software Workshop*, 2009.

[98] G. Bradski. The OpenCV Library. *Dr Dobbs Journal of Software Tools*, 2000.

[99] G. Jiang, R. Voyles, D. J. Cappelleri, D. R. McArthur, S. Mou, A. Yertay, R. Bean, P. Abbaraju, and A. Chowdhury. Purpose-Built UAVs for Physical Sampling of Trace Contamination at the Portsmouth Gaseous Diffusion Plant. In *WM2017 Symposia*, Phoenix, 2017.

[100] D. R. Mcarthur, A. B. Chowdhury, and D. J. Cappelleri. Autonomous Door Opening with the Interacting BoomCopter UAV. In *International Design Engineering Technical Conferences & Computers and Information in Engineering Conference (IDETC/CIE)*, 2019, to appear.

[101] Festo fingripper. `https://www.festo.com/cms/en_corp/9779_10391.htm#id_10391`. Accessed: 2018-05-31.

[102] Tri-max-gripper (conforming robotic gripper). `https://www.thingiverse.com/thing:193851`. Accessed: 2018-05-31.

[103] N. Dalal, B. Triggs, N. Dalal, and B. Triggs. Histograms of Oriented Gradients for Human Detection To cite this version : Histograms of Oriented Gradients for Human Detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 886–893, 2005.

[104] R. Mur-Artal, J. M. Montiel, and J. D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[105] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song. Recent advances and trends in visual tracking: A review. *Neurocomputing*, 74(18):3823–3831, 2011.

[106] Y. Lin, F. Gao, T. Qin, W. Gao, T. Liu, W. Wu, Z. Yang, and S. Shen. Autonomous aerial navigation using monocular visual-inertial fusion. *Journal of Field Robotics*, 35(1):23–51, 2018.

[107] G. Bradski and A. Kaehler. Camera models and calibration. In *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library*, chapter 18, pages 645–699. O'Reilly Media, 2016.

[108] J. W. Demmel. Linear Least Squares Problems. In *Applied Numerical Linear Algebra*, pages 101–138. Society for Industrial and Applied Mathematics, 1997.

[109] A. Vit and G. Shani. Comparing RGB-D sensors for close range outdoor agricultural phenotyping. *Sensors*, 18(12):4413, 2018.

[110] G. Halmetschlager-Funek, M. Suchi, M. Kampel, and M. Vincze. An empirical evaluation of ten depth cameras: Bias, precision, lateral noise, different lighting conditions and materials, and multiple sensor setups in indoor environments. *IEEE Robotics and Automation Magazine*, 26(1):67–77, 2019.

[111] ecl ekf - px4 v1.8.0 developer guide. `https://dev.px4.io/v1.8.0/en/tutorials/tuning_the_ecl_ekf.html`. Accessed: 2019-06-15.

[112] Y. Mulgaonkar, M. Whitzer, B. Morgan, C. M. Kroninger, A. M. Harrington, and V. Kumar. Power and weight considerations in small, agile quadrotors. In *Micro- and Nanotechnology Sensors, Systems, and Applications VI*, volume 9083, page 90831Q, 2014.

[113] S-BRITE Bridge Component Gallery. `https://engineering.purdue.edu/CAI/SBRITE/Facilities/BCGallery/BCG-home`. Accessed: 2018-05-31.

VITA

VITA

Daniel R. McArthur

Education	Purdue University, West Lafayette, IN Doctor of Philosophy (Mechanical Engineering), August 2019
	Purdue University, West Lafayette, IN Master of Science in Mechanical Engineering, August 2016
	Brigham Young University, Provo, UT Bachelor of Science (Mechanical Engineering), April 2014
Research Experience	Graduate Research Assistant (Aug. 2014 - Present) Multiscale Robotics and Automation Lab, Purdue University, West Lafayette, IN.
Work Experience	Research & Development Intern (Dec. 2013 - Aug. 2014) Action Target, Provo, UT
	Automation Engineering Technician (Oct. 2011 - Nov. 2013) US Synthetic, Orem, UT
Publications	D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Autonomous Door Opening with the Interacting-BoomCopter UAV. ASME International Design Engineering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE), 2019, to appear.
	D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Autonomous Control of the Interacting-BoomCopter UAV for Remote Sensor Mounting. IEEE International Conference on Robotics and Automation (ICRA), pp. 5219-5224, 2018.
	D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Design of the Interacting-BoomCopter Unmanned Aerial Vehicle for Remote Sensor Mounting. Journal of Mechanisms and Robotics, 10(2):025001-025001-8, 2018.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Design of the I-BoomCopter UAV for Remote Sensor Mounting. ASME International Design Engineering Technical Conference & Computers and Information in Engineering Conference (IDETC/CIE), 2017.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri. Design of the I-BoomCopter UAV for Environmental Interaction. IEEE International Conference on Robotics and Automation (ICRA), pp. 5209-5214, 2017.

G. Jiang, R. Voyles, D. J. Cappelleri, D. R. McArthur, S. Mou, A. Yertay, R. Bean, P. Abbaraju, and A. B. Chowdhury. Purpose-Built UAVs for Physical Sampling of Trace Contamination at the Portsmouth Gaseous Diffusion Plant. WM2017 Symposia, 2017.

Presentations D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri (2018, November). Purpose-Built UAVs for Physical Interaction with the Environment. Presentation at EPRI 4th Unmanned Aircraft Systems (UAS) for Utility Applications Workshop, Charlotte, North Carolina.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri (2018, February). Interacting-BoomCopter UAV Design for Aerial Manipulation. Poster presented at INDOT Research Program Innovation Fair, Indianapolis, Indiana.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri (2017, September). Interacting-BoomCopter UAV Design for Aerial Manipulation. Poster presented at Dawn or Doom Conference, West Lafayette, Indiana.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri (2017, August). Design of the I-BoomCopter UAV for Remote Sensor Mounting. Paper presented at ASME International Design Engineering Technical Conferences, Cleveland, Ohio.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri (2017, May). Interacting-BoomCopter UAV Design for Aerial Manipulation. Poster presented at Midwest Robotics Workshop, Chicago, Illinois.

D. R. McArthur, A. B. Chowdhury, and D. J. Cappelleri (2016, August). Swabbing UAVs. Technical demonstration performed at DOE-EM Science of Safety: Robotics Challenge, Portsmouth Gaseous Diffusion Plant, Piketon, Ohio.

Honors/Awards

Helen & John Lozar Assistantship (Aug. 2018 - Present)
School of Mechanical Engineering, Purdue University

NSF Graduate Research Fellowship (Aug. 2015 - Aug. 2018)
National Science Foundation

Laura Winkelman Davidson Fellowship (Aug. 2014 - Aug. 2015)
School of Mechanical Engineering, Purdue University

Heritage Scholarship (Sep. 2006 - Aug. 2013)
Brigham Young University