ABSTRACT

Title of Thesis: AUTONOMOUS HYBRID AERIAL-TERRESTRIAL TRANSPORT AND RETRIEVAL FOR IN-SITU COLLECTION

Madelyne G. Rossmann
Master of Science, 2025

Thesis Directed by: Professor Derek Paley
Department of Aerospace Engineering

This thesis investigates the use of hybrid vehicles, capable of both aerial and terrestrial motion, for sample search and retrieval, enabling autonomy through the customization of existing software for new applications. This research addresses challenges related to autonomous detection of, and navigation with respect to, fiducial markers - in both the aerial and terrestrial domains. A key contribution of this work is that of the hybrid mobility system, which is capable of transitioning autonomously between flying and driving. To assist in sample detection, an existing software package for fiducial marker recognition is utilized and adapted for localization of the target and positioning of the vehicle for sample collection. For sample collection, a simple gripper is designed to grasp the target object before depositing at a home location. This research leverages software-in-the-loop simulation for validating code performance prior to deployment on the actual vehicle, which is tested in a controlled, indoor laboratory environment.

AUTONOMOUS HYBRID AERIAL-TERRESTRIAL TRANSPORT AND RETRIEVAL FOR IN-SITU COLLECTION

by

Madelyne G. Rossmann

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2025

Advisory Committee:
Professor Derek Paley, Chair/Advisor
Professor Joseph Conroy
Professor Mumu Xu

Acknowledgements

For the completion of this thesis, I would like to foremost thank my parents for their unwavering support throughout my academic career. They have stood by me and encouraged me from afar, and I am proud to be their daughter. I would also like to thank my aunt, Missy, for opening my eyes to the world of higher education back when I was just a kid, and always holding me accountable as I have fumbled my way through early adulthood.

To Tom and Mary, I am forever appreciative that our paths crossed five years ago. You have offered me a home on the East Coast, and I can attribute so much of my personal and professional growth to your guidance. I leave every one of our interactions feeling as if I have learned something new about the world, and I cannot imagine being in the position I am now without you both in my corner.

I am eternally grateful for my friends that have kept me grounded throughout this whole process. Max, thank you for always picking up the phone when I needed it and offering me a healthy dose of realistic optimism. Josh and Nic, thank you for always being willing to grab coffee and make me laugh on even my worst days. Will, thank you for spending countless of hours by my side so that I would always have someone to lean on when times were tough. To the Pit, words cannot express how much you all mean to me. Our continued friendship as we've all set off on our own life paths is a testament to the bond we've forged.

I'd also like to thank Mahima Thirukkonda and Sabrina Zaleski, my two undergraduate

research assistants during the summer and fall of 2024. Mahima provided invaluable knowledge on the aerial platform utilized throughout this research, and Sabrina masterfully crafted the ground chassis of the terrestrial platform. I know they will do great things, and I am honored to have been even a small part of their careers.

Gratitude is also due for Ivan Penskiy of the Maryland Robotics Center. You always answered my requests for assistance with patience and grace, and encouraged me to think like an engineer. If I ever learn to read an oscilloscope, you will be the first person I think of.

I am also appreciative of those serving as committee members for this thesis. Dr. Xu, thank you for serving as my introduction to ROS when I was an undergraduate student and helping lay the groundwork for the skills I've honed through the process of this thesis. Dr. Conroy, thank you for being my guide to the VOXL platform and always offering a helping hand, and an ear for listening, when I needed it. Dr. Paley, thank you for taking a chance on me as a graduate student in your lab and helping me grow as a researcher in this process.

For funding my graduate studies I'd like to thank two entities: Microsoft and the Maryland Robotics Center. For the 2023-2024 academic year, I was appointed a Microsoft Diversity Fellow through the Microsoft Diversity in Robotics and AI Fellowship Program at the University of Maryland. For the 2024-2025 academic year, I was awarded a Maryland Robotics Center Graduate Research Assistantship.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ABS	Acrylonitrile Butadiene Styrene
A-HAT-TRIC	Autonomous Hybrid-Aerial Terrestrial Transport and Retrieval for In-Situ Collection
AQT-HR	Autonomous Quadrotor Tilting Hybrid Robot
CAD	Computer-Aided Design
COTS	Commercial Off-The-Shelf
DDS	Data Distribution Service
EKF	Extended Kalman Filter
EVA	Ethylene–Vinyl Acetate
FRD	Front-Right-Down
HITL	Hardware in the Loop
HyTAQ	Hybrid Terrestrial/Aerial Quadrotor
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit
IP	Internet Protocol
JPL	Jet Propulsion Laboratory
MTABot	Morphable Terrestrial-Aerial Robot
NASA	National Aeronautics and Space Administration
PID	Proportional-Integral-Derivative
PLA	Polylactic Acid
PU	Polyurethane Rubber
PWM	Pulse-Width Modulation
ROS 2	Robot Operating System 2
SITL	Software in the Loop
SRH	Sample Recovery Helicopter
UAV	Uncrewed Aerial Vehicle
UDP	User Datagram Protocol
UGV	Uncrewed Ground Vehicle
VIO	Visual-Inertial Odometry

Chapter 1: Introduction

1.1 Motivation and Objectives

With the success of the Mars 2020 mission, landing both the Perseverance rover and Ingenuity helicopter on the Martian surface, the National Aeronautics and Space Administration (NASA) is in the midst of preparing an ambitious follow-up. Aptly named Mars Sample Return, this program is tasked with collecting the samples deposited by Perseverance in Jezero Crater and returning them to Earth [1] This is a multi-mission campaign, requiring highly-specialized spacecraft for each individual task required. For sample retrieval specifically, the NASA Jet Propulsion Laboratory (JPL) has proposed twin Sample Recovery Helicopters (SRHs) [2].

This rotorcraft concept, shown in Figure 1.1, draws on the heritage of Ingenuity, but expands current capabilities to include terrestrial locomotion and sample tube collection. Flight allows SRH to avoid complex terrain during transit to sample locations, and the ability to drive on the surface allows it to more precisely reposition itself relative to a sample tube for pick-up.

While the SRH concept proposed by NASA JPL has (at the time of writing) yet to be confirmed for flight, it is an interesting design with implications for use on Earth. Certain regions on Earth contain hazardous environments that cannot safely be surveyed by field researchers, such as active volcanoes or areas experiencing extreme weather conditions. Volcanologists, specifically, have a history of utilizing robots for performing science during periods of intense

Figure 1.1: Sample Recovery Helicopter concept art [3]

volcanic activity [4]. Multiple robotic platforms have been designed to traverse the complex terrain surrounding a volcano, including a multi-legged configuration [5] and a Marsokhod rover [6], and aerial robots have been deployed for surveillance and remote-sensing purposes [7]. There are currently no published studies of a hybrid aerial-terrestrial robot for volcanic exploration.

Aerial transit allows a robot to maneuver unencumbered by the challenging, and at times changing, terrain typically present in volcanic regions [8], making it an ideal method of long-distance travel. However, the ability to operate at ground-level is still imperative for certain science goals - especially collection of surface materials when precise position control of the vehicle is required. Combining these two methods of travel clearly has the potential to increase mission efficiency and science return, which is reflected by NASA JPL's proposal to send the SRHs to Mars for collection of the sample tubes deposited by Perseverance.

This thesis work aims to apply the planned capabilities of SRH to a vehicle operating

within Earth's atmosphere, specifically designed to assist researchers with gathering samples of solid objects from a desired search area. The vehicle constructed through this research requires minimal user input to complete its sample detection and collection mission, utilizing commercially available software, in conjunction with custom software packages, to enable this autonomous feature. By engineering a hybrid, autonomous vehicle with sample collection capabilities, this research aims to provide a robotic platform capable of assisting field researchers on Earth in surveying hazardous environments. This ultimately has the power to mitigate the potential health issues incurred by researchers working in such environments, and also removes the need to continuously pilot the vehicle during operation, relaxing the training requirements and allowing researchers to focus on scientific goals rather than vehicle performance.

1.2 Relation to Previous and Ongoing Work

The increasing exploration of the hybrid aerial-terrestrial vehicle concept signifies the potential for such a class of robot. Existing research largely focuses on optimizing the design of the vehicle, itself. This thesis is an addition to this ongoing work, focused on applying a simple hybrid aerial-terrestrial vehicle to a specific use case.

A significant amount of the research done investigates this vehicle concept as a potential solution to the power efficiency problem of rotary wing uncrewed aerial vehicles (UAVs), used interchangeably with the term drones throughout this thesis. UAVs consume significant battery power when in hover, which is only exacerbated by the addition of a payload [9]. For activities where the drone is expected to operate in hover near the surface, such as performing scientific observations or measurements, equipping the vehicle with terrestrial maneuverability is a potential

solution to this issue.

Many vehicle designs leverage the aerial flight motors for terrestrial motion in an attempt to improve power efficiency. One such design, the hybrid terrestrial/aerial quadrotor (HyTAQ), features a drone mounted within a cylindrical cage such that it can affect terrestrial movement by pitching [10]. On a linoleum surface the HyTAQ indicated a terrestrial range about 11 times greater than its maximum aerial range at an equivalent speed, although this ratio does decrease as the coefficient of friction of the terrestrial surface increases.

Another proposed vehicle design, the autonomous quadrotor tilting hybrid robot (AQT-HR), also adopts this idea of utilizing the quadrotor's capabilities for driving the vehicle on the ground [11]. The AQT-HR consists of a quadrotor drone mounted on a tilting platform such that the vehicle is able to drive and steer on the ground by rotating about any of its body axes. When operating in flight the vehicle required about 340 Watts to operate, while terrestrial operations needed only 110 Watts for movement. This is a dramatic difference in operational power, illustrating the potential effectiveness in incorporating ground mobility for drone platforms for the purpose of increasing power efficiency.

As the power savings of these hybrid vehicles have been realized, studies focused on improving terrestrial maneuverability have appeared. One concept, the morphable terrestrial-aerial robot (MTABot), is capable of three distinct methods of movement: rolling, climbing, and flying [12]. This vehicle achieves flight using a bicopter design and leverages its flight motors for terrestrial movement, rotating them 90 degrees between flight and ground operations. This vehicle demonstrated the ability to overcome ground obstacles of a height up to 1.2 times its wheel radius while maintaining a greater power efficiency than demonstrated in flight.

At the time of writing, the most recent study published proposes a quadcopter design

featuring a wheeled leg for terrestrial maneuvering mounted beneath the vehicle's chassis [13]. This passively reconfigurable appendage enables a smooth transition from flight to ground operations, where the dynamics of the quadcopter itself are exploited for terrestrial motion. By commanding steering through roll this vehicle indicated an improved robustness, as compared to other hybrid vehicle designs, on uneven terrain: both indoor and outdoor. This enhanced maneuverability did result in a lower power efficiency than comparable vehicles, highlighting the way distinct optimization goals can affect performance.

Instead of optimizing vehicle design, this thesis strives to validate a specific application for this class of hybrid vehicle: sample search and collection. The proposed platform, known through this thesis as autonomous hybrid aerial-terrestrial transport and retrieval for in-situ collection (A-HAT-TRIC), consists of a commercial off-the-shelf (COTS) quadcopter mounted on a ground chassis designed specifically for this thesis work. A-HAT-TRIC is designed such that it can, with minimal operator input, be deployed for sample search and retrieval in a defined search area. Exploring the validity of a simple hybrid aerial-terrestrial vehicle for sample collection adds to the existing body of work by introducing a specific application this unique robotic platform.

1.3 Technical Approach

To construct the A-HAT-TRIC platform, a ground chassis and simple gripper are designed using computer-aided design (CAD) software such that they are compatible with the selected COTS quadcopter chassis. A microcontroller enabling command of the ground system is integrated into the system, and communication between the microcontroller and the flight computer is established utilizing user datagram protocol (UDP). The electronics for the ground system are

powered entirely by the flight battery. Although the two platforms can, with some adjustment, function independently, this vehicle design produces one platform capable of both flight and terrestrial locomotion.

Sample search and detection is achieved using a custom autonomous framework, and a high-level concept of operations is displayed in Figure 1.2. Upon startup, the algorithm requests the dimensions of the search area, the starting coordinate for the search, and the number of samples expected. After the operator manually arms the vehicle, it lifts off vertically and navigates to the desired search area. In its search state, A-HAT-TRIC traces a lawnmower pattern while utilizing its image sensors for sample detection. The samples are small cubes with an identical fiducial marker, specifically an AprilTag, on each face, and are each positioned next to a larger marker of a unique identity.

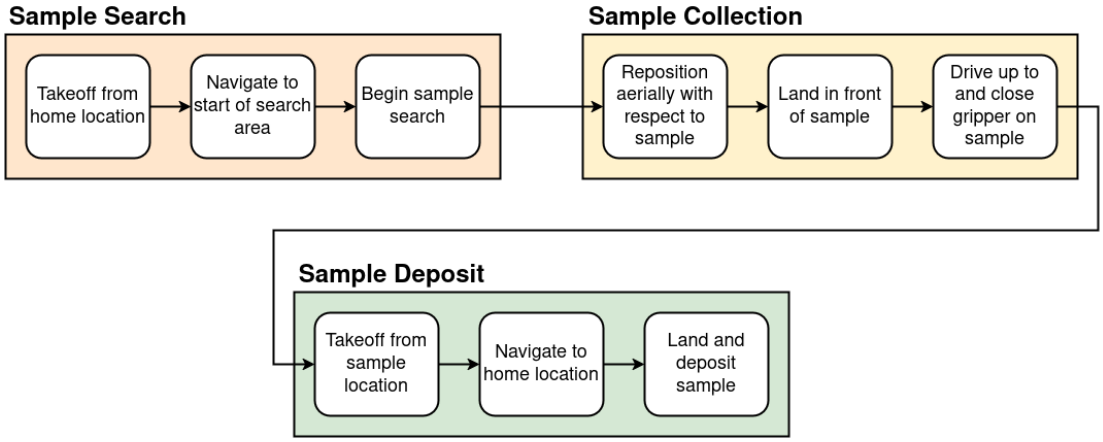


Figure 1.2: High-level concept of operations

Once the vehicle detects the larger marker, it exploits it for localization in inertial space. With these localized coordinates, A-HAT-TRIC performs a precision landing such that it will be able to simply drive forward to collect the sample. Once the vehicle has successfully maneuvered itself to the optimal location and orientation, it closes its gripper and performs a vertical takeoff.

The vehicle then navigates aerially back to the home location before landing and depositing the sample. This cycle repeats until the indicated number of samples are collected.

During early development of the autonomous framework for A-HAT-TRIC, rigorous software in the loop (SITL) testing is performed to remove any unwanted actions from the program. Only after SITL testing is satisfactory is the software deployed onto the actual platform. Hardware in the loop (HITL) experimentation is a major part of verifying vehicle performance, as simulation cannot exactly replicate the reality of the laboratory environment. HITL assessments also allow for any issues with communication between the flight computer and ground microcontroller to be adequately assessed and handled.

This approach to enhancing the capabilities of the vehicle allows for rapid prototyping and development, significantly aiding in progress and timely mitigation of errors. Although the focus of this thesis is on the autonomous sample collection framework, a considerable emphasis was also placed on ensuring optimal performance of the custom-built vehicle. To successfully execute the algorithm for sample retrieval, the vehicle needed to be fully capable of performing as required. The outlined steps for achieving the goals of this research enabled an efficient process for designing a vehicle specifically for sample search and collection with minimal operator input.

1.4 Contributions

The primary contributions of this thesis are as follows:

1. **Simple Hybrid Aerial-Terrestrial Vehicle Design:** A custom-built terrestrial platform is created to enable a commercial quadcopter drone to move terrestrially. The simplicity of the design allows for greater emphasis to be placed on software development that exploits

the vehicle for a specific use case, sample detection and collection, without being difficult to replace in case of damage to mission-critical hardware.

2. **Mission Commanding from Primary Computer:** Each mode of transit is individually commanded by its own respective computer. Establishing a communication protocol between them enables smooth commanding from the flight computer to the ground microcontroller using only one computer program for mission execution. This dramatically simplifies the development of code and identification of any errors.

3. **Sample Collection Routine Requiring Minimal Operator Input:** Aside from arming the vehicle, an autonomous framework is deployed to command the vehicle at each stage of the sample detection and collection process. This algorithm adeptly manages the transition between each stage in the mission, and commands the vehicle accordingly. This sample collection routine is rigorously testing in both simulation and laboratory demonstrations to validate its performance.

4. **Experimental Analysis of Sample Collection Mission:** The designed vehicle and its autonomous algorithm for sample search and retrieval is tested in multiple laboratory tests. Analyzing vehicle performance in these trials assists in determining the usefulness for this design scheme in real-world applications.

1.5 Outline of Thesis

Chapter 2 provides an overview of essential concepts leveraged in this thesis. This includes quadrotor UAV dynamics, flight control systems, the utility of fiducial markers, and the software

enabling the development of robotic platforms. The hardware design process for the A-HAT-TRIC vehicle is detailed in Chapter 3, culminating in the complete assembly of this hybrid robotic platform. In Chapter 4, the autonomous framework enabling the vehicle to perform its mission is described. Chapter 5 contains an analysis of experimental results obtained from the A-HAT-TRIC vehicle in an indoor laboratory environment. This thesis concludes with Chapter 6, where a summary of research contributions is provided along with suggested future work.

Chapter 2: Background

This chapter provides an essential overview of concepts related to the core technologies and methods used throughout this thesis.

2.1 Overview of the Quadcopter Uncrewed Aerial Vehicle

2.1.1 Quadcopter System Definition

A quadcopter is comprised of four rotors capable of providing thrust for flight. These rotors are equidistant from the center of the airframe in a cross configuration. One pair of rotors rotate counter-clockwise, and the other pair rotates clockwise. A pair is comprised of the two rotors immediately opposite one another across the center of the airframe. This definition of the quadcopter system, including its body frame, is illustrated in Figure 2.1.

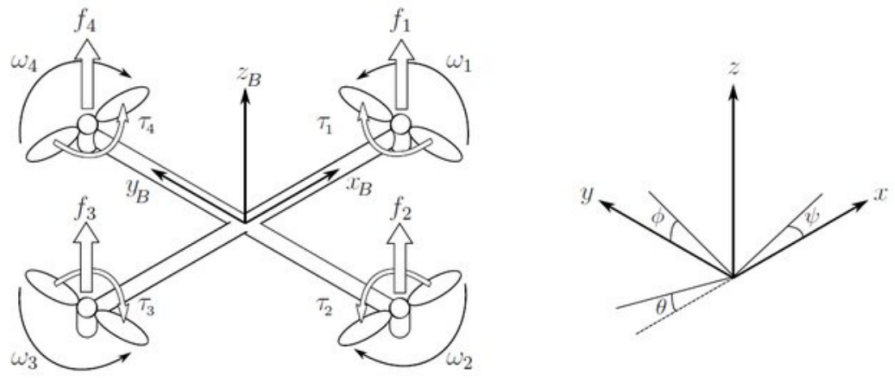


Figure 2.1: Quadcopter model with body frame labeled [14]

Aerial maneuvers are achieved by varying the magnitude of the angular velocity of each rotor, which in turn affects the thrust and torque produced. The quadcopter system can be linearized such that the differential equations of motion can be derived for controlling the vehicle. By selecting the inputs to the controller as the torque and thrust produced by each rotor, both translational and rotational motion of the UAV can be commanded.

2.1.2 Flight Controller

The PX4 flight control system was utilized for this research, though it is worth noting that there are other open-source flight controllers available for use. PX4 is an active project with international collaborators from both industry and academia [15]. This platform supports a variety of different aircraft, including fixed-wing configurations, and has even been applied to UGVs and submersible vehicles. PX4 is especially appealing due to its robust documentation and effective support forums.

For a quadcopter, the PX4 flight controller implements a mix of proportional and proportional-integral-derivative (PID) cascaded controllers [16]. Translational position and velocity control are managed with respect to the inertial frame, while angular position and velocity control are conducted with respect to the drone's body frame. The cascaded controller accepts state estimates of the position and yaw angle from an Extended Kalman Filter (EKF) as inputs, and ultimately outputs the required thrust for each rotor. A high-level block diagram for this control scheme is displayed in Figure 2.2. Further detail on each of the individual controllers is available in the PX4 controller documentation.

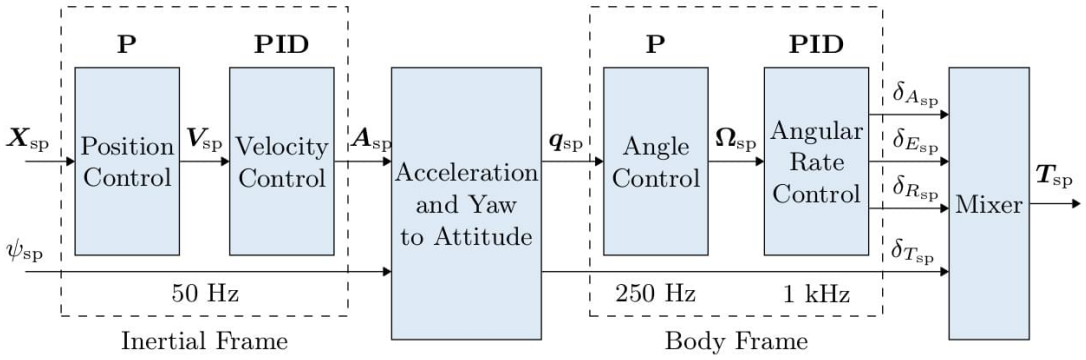


Figure 2.2: PX4 control architecture for multicopter platforms [16]

2.2 Fiducial Marker Detection and Localization

Slight modifications were made to the included software highlighted in this section to make the detection framework directly compatible with the autonomous algorithm. These customizations are detailed in Appendix A.

2.2.1 Reference Frame Definitions

There are four reference frames utilized in this work: the inertial frame, the vehicle-fixed body frame, the camera frame, and the AprilTag frame.

The inertial frame is in a Front-Right-Down (FRD) configuration. The inertial z-axis is nadir-pointing, and the exact orientation of the x- and y-axes are variable upon vehicle boot up. This frame remains fixed as the vehicle operates, and is the frame in which all setpoints are sent to the flight computer.

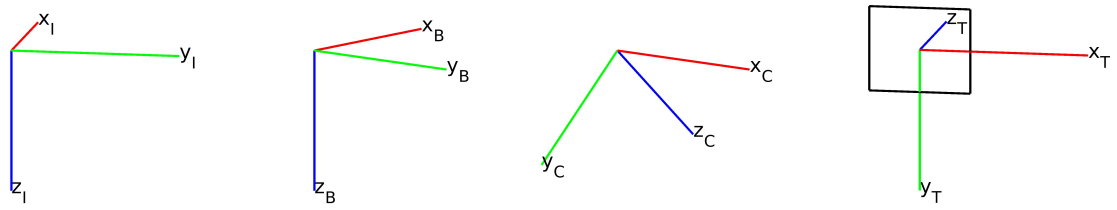
The vehicle-fixed body frame is also arranged in the FRD format and centered on the vehicle's flight deck. The body z-axis points directly out of the bottom of the vehicle, aligned with the inertial z-axis before moving. The body x-axis is oriented in the direction of the vehicle's

heading, which is defined as the face of the vehicle's flight deck containing the cameras utilized for AprilTag detection. The body y-axis completes the set. The body frame heading is calculated as the yaw rotation between the inertial and body frames.

For visual-inertial odometry (VIO) and AprilTag detection, the ModalAI VOXL infrastructure is utilized. In this framework, the camera frame z-axis points directly out of the lens, the x-axis points to the right, and the y-axis completes the set [17]. The origin of the frame is located at the lens of the camera. The cameras utilized for this project are at a pitch angle of -45 degrees with respect to the body frame.

The definition of the AprilTag reference frame is also defined by the ModalAI software environment [18]. The origin of the frame is at the center of the marker. The x-axis points to the right, the y-axis points to the bottom edge, and the z-axis is directed into the marker.

An example configuration of these four reference frames is displayed in Figure 2.3. In this illustration, the body frame (b) is at a yaw angle of 30 degrees with respect to the inertial frame (a) and the camera frame (c) is oriented accordingly. The AprilTag frame (d) is arbitrary in this depiction, but a square is included to emphasize the orientation of the frame with respect to the marker itself.



(a) Inertial frame (b) Body frame (c) Camera frame (d) AprilTag frame

Figure 2.3: Reference frames in example configuration

2.2.2 Fiducial Marker Detection

The onboard ModalAI service, `voxl-tag-detector`, is employed for AprilTag detection [18]. This service can subscribe to up to three camera image feeds, though the work in this thesis only applies it to the tracking camera and high-resolution image sensor. This requires the cameras to be calibrated and indicated in a configuration file that directs the service to the relevant image feeds.

The service also requires that certain AprilTag information be provided in a configuration file. This includes the size, type, and identification number of the markers expected. The service currently includes support only for AprilTags in the 36h11 family. An example AprilTag in this family is shown in Figure 2.4.

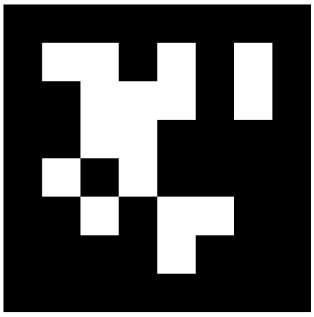


Figure 2.4: AprilTag ID 0 in 36h11 family [19]

The `voxl-tag-detector` service outputs the AprilTag's position and orientation with respect to the frame of the camera by which it was detected. These outputs are accessed via the `voxl-mpa-to-ros2` node, which converts camera information into a usable format for autonomous software development [20].

2.2.3 Fiducial Marker Localization in Inertial Space

Localizing an AprilTag in the inertial frame requires a series of coordinate transformations, including rotations and translations. There are four points of interest in this process:

1. The position of the body frame origin with respect to the inertial frame, $[\vec{r}_{body}]_{inert}$. Reported by the PX4 `VehicleLocalPosition` message [21].

2. The position of the intermediate inertial measurement unit (IMU) frame origin with respect to the body frame, $[\vec{r}_{IMU}]_{body}$. Obtained from the ModalAI extrinsic configuration file [22].

3. The position of the camera frame origin with respect to the intermediate IMU frame, $[\vec{r}_{cam}]_{IMU}$. Defined in the ModalAI extrinsic configuration file [22].

4. The position of the AprilTag frame origin with respect to the camera frame, $[\vec{r}_{tag}]_{cam}$. Calculated by the `voxl-tag-detector` service.

The appropriate rotation matrices can also be obtained from the information made available by the flight computer. The PX4 `VehicleAttitude` message provides the rotation from the body frame to the inertial frame ${}^{inert}\boldsymbol{R}^{body}$ [23]. Both the rotation from the IMU frame to the body frame ${}^{body}\boldsymbol{R}^{IMU}$ and the rotation from the camera frame to the IMU frame ${}^{IMU}\boldsymbol{R}^{cam}$ are obtained from the extrinsic configuration file [22].

Each of the position vectors can then be converted to the inertial frame and summed to

obtain the position of the AprilTag in the inertial frame.

$$[\vec{r}_{IMU}]_{inert} = \left[{}^{inert}\boldsymbol{R}^{body}\right] [\vec{r}_{IMU}]_{body} + [\vec{r}_{body}]_{inert}$$

$$[\vec{r}_{cam}]_{inert} = \left[{}^{inert}\boldsymbol{R}^{body}\right] \left[{}^{body}\boldsymbol{R}^{IMU}\right] [\vec{r}_{cam}]_{IMU} + [\vec{r}_{IMU}]_{inert}$$

$$[\vec{r}_{tag}]_{inert} = \left[{}^{inert}\boldsymbol{R}^{body}\right] \left[{}^{body}\boldsymbol{R}^{IMU}\right] \left[{}^{IMU}\boldsymbol{R}^{cam}\right] [\vec{r}_{tag}]_{cam} + [\vec{r}_{cam}]_{inert}$$

The package for performing this localization works with the `voxl-mpa-to-ros2` node, so that the localized coordinates are available to the autonomous algorithm deployed.

2.3 Robotic Software Overview

There are two different branches of development for the software used in this thesis work. In each branch, a different combination of the robotic software discussed in this section is applied. These combinations are shown in Figure 2.5, where each box represents a separate branch of development. The specifics of each branch are discussed in Chapter 4.

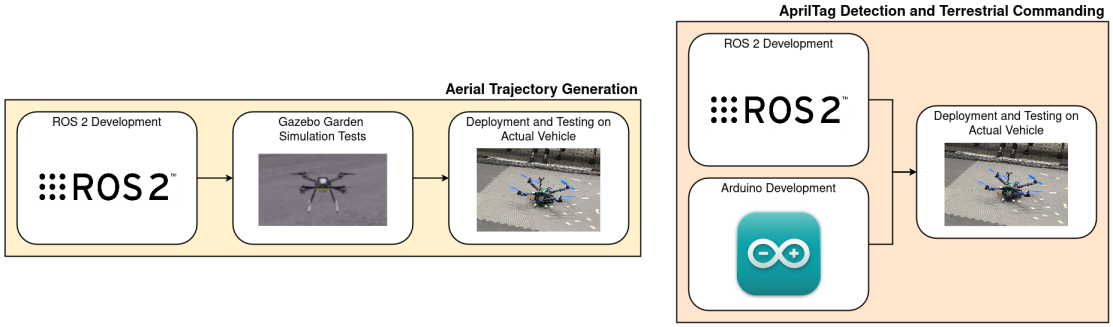


Figure 2.5: Use of robotic software in different branches of software development

2.3.1 Robot Operating System 2

Robot Operating System 2 (ROS 2) is an open-source software environment that assists in the development of robotic applications and is the successor of ROS 1, which is slated for end of life in May 2025 [24]. ROS 2 is designed for more complex robotic systems, addressing the limitations of ROS 1 with respect to the rapid pace of technological advancement since its inception [25].

ROS 2 utilizes industry standard protocols, namely the Data Distribution Service (DDS), to improve its middleware [26]. This layer of software enables communication between different components of a robotic system and this standardized approach allows for modularized software development. ROS 2 middleware is designed for decentralized platforms, such as multi-robot systems, while its predecessor relied on a single master node. This improvement makes ROS 2 a more viable solution for modern robotics projects.

The open-source nature of ROS 2 facilitates a streamlined process of package development for specific use cases. This allows for the design of custom message types compatible with actual hardware utilized on a robotic platform. ROS 2 is also compatible with a variety of simulation environments, enabling rigorous testing in virtual setups prior to deployment on a real robot. The developers also maintain extensive documentation for the ROS 2 environment online [25]. The robust support from both the community and its developers emphasizes ROS 2 as a valuable framework within which to develop the software necessary for robotic systems.

In this project, ROS 2 is utilized on the flight computer to execute the vehicle's sample detection and collection algorithm. It also communicates with the microcontroller for the ground system, allowing for both aerial and terrestrial control in one node.

2.3.2 Arduino Integrated Development Environment 2

The Arduino Integrated Development Environment (IDE) 2 is an editor utilized for creating code for deployment on Arduino microcontrollers [27]. The Arduino programming language is a simplified version of C++, and the Arduino IDE 2 features a compiler that is used prior to uploading code to the targeted microcontroller. This all-in-one platform is an effective tool for code testing and development, and its simplicity makes it very accessible for users of all backgrounds and proficiencies.

The Arduino framework is exploited for the ground system in this research. It provides a simplistic control scheme for the motors for both locomotion and gripper operation, accepting inputs from the ROS 2 node operating on the flight computer.

2.3.3 Virtual Simulation Configuration

The Gazebo Garden realistic, physics-based simulation environment is utilized for SITL testing in this project. This simulator is partnered with the `ros_gz_bridge` package so that software developed in the ROS 2 framework can be applied to a virtual robotic platform [28]. Gazebo is a non-deterministic physics simulator. This means that, even if the same inputs are provided to the simulation, different outputs can be produced unless the seed is fixed. In this research, the random seed was not fixed, which allowed for uncontrollable discrepancies between virtual trials. This can be observed in the simulation data presented in Chapter 4.

The PX4 software, used for flight control, works with ROS 2 through its `px4_ros_com` package. This package allows ROS 2 to both receive information about vehicle performance and send commands to the flight controller [29]. The ability to do this is provided through the

uXRCE-DDS middleware, which converts PX4 messages into a format readable by ROS 2 [30]. Onboard the actual vehicle, this service is automatically enabled. For simulation purposes, it must be launched manually.

The simulation environment itself leverages files available from PX4 to populate the scene. An X500 quadrotor model is used as the virtual drone [31] and the Baylands model serves as the simulation world [32]. A screenshot of this simulation configuration upon boot up is displayed in Figure 2.6.

Figure 2.6: Virtual quadcopter in Gazebo simulation environment

Chapter 3: Vehicle Hardware Design and Assembly

3.1 Modified Aerial Platform

The aerial platform is centered on the ModalAI VOXL 2 framework, utilizing compatible components from other manufacturers to complete the assembly. A list of the main parts procured for this build can be found in Table B.1 in Appendix A.

The only modification made to the COTS components for the aerial platform was to install a custom mount for the high-resolution image sensor, hereafter referred to as the hi-res sensor, on the flight deck. This angled the hi-res sensor down 45 degrees and extruded it such that its field of view was not blocked by the tracking camera beneath it. The mount was designed in a CAD program and 3D printed. An image of the aerial platform, including this fabricated hi-res sensor mount, is displayed in Figure 3.1.

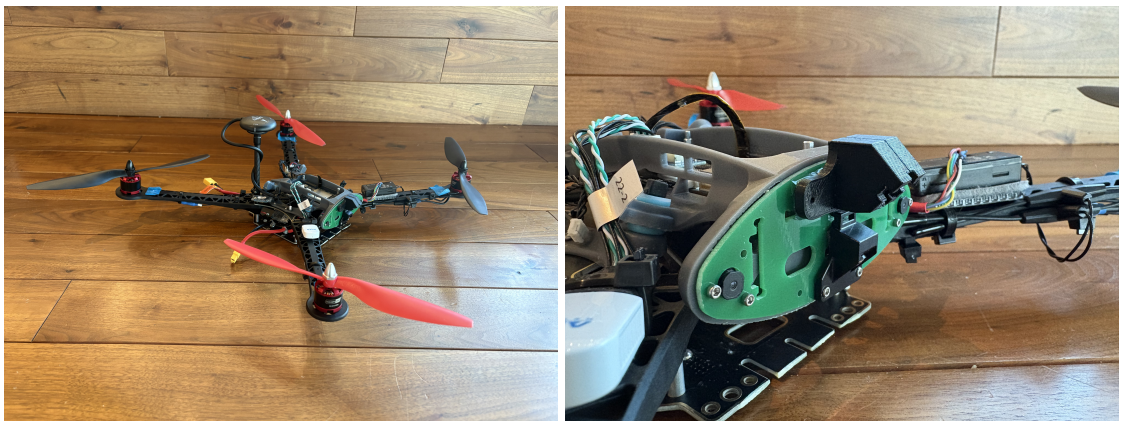


Figure 3.1: A-HAT-TRIC aerial platform with camera mount

The VOXL 2 flight computer functions using the Ubuntu 18.04 Linux-based operating system. The Foxy distribution for ROS 2 is installed for implementing autonomous control of the vehicle. The flight computer is of system image 1.7.10, and relies on version 1.14 of the PX4 flight controller software.

3.2 Custom Terrestrial Platform

The main components for the terrestrial platform are provided in Table B.2 in Appendix A. The ground chassis design, specifically, was engineered by University of Maryland undergraduate student Sabrina Zaleski.

3.2.1 Design Constraints

The design of the terrestrial platform was mainly limited by the maximum allowable payload and battery capacity of the aerial system. For the selected aerial platform, the ground chassis was restricted to 1 kilogram [33] and had an input of 12 volts to accommodate the electronic speed controllers [34]. Since the scope of this project does not require a terrestrial platform capable of traversing complex terrain, powerful motors were not necessary and mass became the primary constraint. It was also desirable to create a terrestrial platform with a uniform mass distribution centered about the aerial system's center of mass, as this would limit the impacts on flight performance.

To mitigate the strict mass constraints, it was decided that the ground chassis and gripper would be entirely 3D-printed. Acrylonitrile butadiene styrene (ABS) plastic was utilized for the 3D prints due to it being both more durable and lighter than polylactic acid (PLA) plastic [35].

An additional benefit to 3D printing these parts is that it made rapid prototyping possible. Parts were ready within 24 hours of submitting a print job, enabling a more streamlined process for custom-fitting the ground chassis to the aerial platform.

3.2.2 Locomotive Motor Selection

The motors for driving were selected using a basic calculation of the torque required to transport the full chassis of the drone, including battery, and maximum payload capacity, to account for the terrestrial platform. This resulted in a total allowable mass m of 1.347 kilograms [33]. Wheels designed to attach to standard servo splines were also selected prior to the analysis, defining wheel diameter d as 70 millimeters [36]. A conservative, estimated static friction coefficient μ of 0.16 was utilized, as the floors in the testing facility are a blend of ethylene–vinyl acetate (EVA) and polyurethane rubber (PU) foams, and this value represents the maximum coefficient possible for these materials [37].

To minimize weight, only two motors were included for terrestrial locomotion. The total propulsive force required to overcome static friction was calculated and then halved to determine the force necessary on a per motor basis. The required torque per motor was then obtained using the diameter of the wheel and the force per motor.

$$\tau_{req} = \frac{1}{4}\mu m g d = 0.064 \text{ N-m}$$

This yielded a relatively small amount of required torque per motor, allowing for significant flexibility in the selection of the driving motors.

To avoid potential issues with mounting, only the motors cited as compatible with the

selected wheels were directly considered. This resulted in two options to compare: the SpringRC SM-S4303R [38] and FEETECH FS5106R [39] continuous rotation servos, respectively. The simple quantitative analysis used to decide the motor for the locomotion is shown in Table 3.1. Both motors boast an excess of torque, so mass was optimized in this analysis. As such, the

	SpringRC SM-S4303R	**FEETECH FS5106R**
Operating Voltage	5 V	5 V
Mass	41 g	43 g
Torque	0.50 N-m	0.59 N-m

Table 3.1: Selection criteria for driving motors

SpringRC SM-S4303R continuous rotation servos were selected. These motors operate using pulse-width modulation (PWM) signals from the Arduino Nano ESP32 microcontroller and are capable of reversing the direction of spin, allowing a simple set-up process.

3.2.3 Gripper Design

The gripper was designed based on the selected shape of the sample to be collected, a cube. Although not especially versatile to different sample shapes, this research is meant to demonstrate the feasibility of sample collection rather than showcase the variety of use cases. Different end effectors that leverage the design put forth in this thesis could be optimized for other sample shapes.

To minimize mass, a 9 gram motor was selected to operate the gripper. The FT90B digital micro servo [40] utilizes PWM signals and requires 3 volts to function, allowing it to be powered by the Arduino Nano ESP32 that controls the ground system. Since one motor was utilized, the gripper was restricted to opening and closing only. It was decided that adding the ability to lift the sample off of the ground was not necessary, as the vehicle does not drive on the ground after

collection except to ascertain the sample has been grasped.

Simple spur gears, of a one-to-one ratio, were included to ensure that both halves of the end effector were moving in the same manner. One gear was attached directly to the motor and the other was mounted to the box containing the motor, both equidistant from the centerline of the vehicle. The spur gears were integrated into the same part as the end effectors themselves, ensuring the end effectors could not rotate independent of motor commands and would remain in their desired position.

Upon assembly, however, it was discovered that the rotation of the non-motorized half of the end effector would loosen its mounting hardware. As such, the gears were sanded down, and the non-motorized half of the end effector was set in its closed position. This allows the gripper to clasp the sample with its motorized end effector half, leveraging the rigidly mounted non-motorized half to secure the sample.

To combat the low friction coefficient between the 3D-printed part and the surface of the sample, rubber thimbles were acquired to augment the texture of the end effectors. The thimbles were custom-fit to the designed parts and attached using super glue. This simple measure proved successful in enabling the gripper to hold the sample during flight.

As previously mentioned, a box was designed to house the motor and provide a surface on which to mount the end effector. This box was designed around the dimensions of the motor itself and includes two rectangular extrusions, which are mirrored across the centerline of the vehicle. The rectangular shape of these extrusions prohibit rotation of the box, as an arm attaching the box to the chassis connects directly to this mounting point via rectangular cut-outs. This design removed the need for mounting hardware, leveraging the rigidity of the 3D prints and also lowering overall mass.

An image of the final gripper assembly is shown in Figure 3.2. This depiction specifically showcases the gripper in the closed position, where the tips of the gripper are parallel to one another.

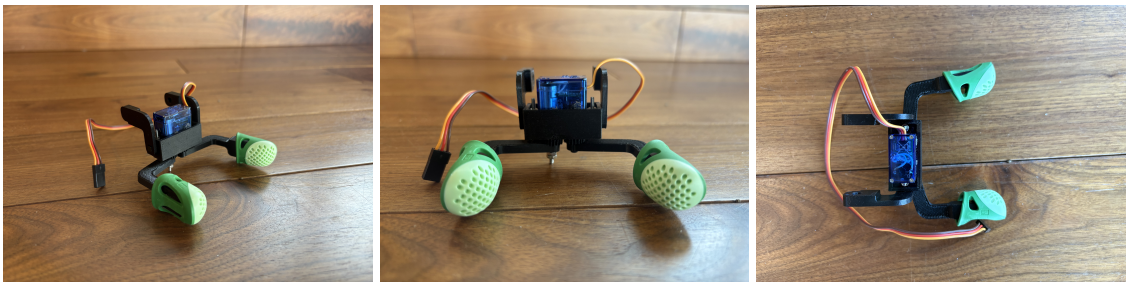


Figure 3.2: A-HAT-TRIC gripper assembly

3.2.4 Ground Chassis Design

The ground chassis itself was designed specifically to mount to the frame for the aerial platform. The available CAD model for the airframe [33] was leveraged in this process, ensuring that all attachment points were an exact fit. The ground chassis includes an upper deck for mounting to the aerial platform, and a lower deck on which to house the electronics for the terrestrial platform. This configuration includes the added benefit of less material required, assisting in minimizing the weight of the chassis.

Simple brackets were designed to mount the locomotive motors to the underside of the ground chassis. The motors were positioned in the rear of the chassis so as to offset the weight of the collected sample during flight. This rear-wheel drive design is complemented by a plastic ball caster placed at the front of the chassis. A ball caster permits multi-directional movement, which can be affected by the motors. This design choice also served to reduce the weight of the ground chassis, as it is a single component made of lightweight materials. A simple mount was

designed for this ball caster to be attached to the main body of the ground chassis.

The ground chassis also includes a mounting point for the gripper centered on its front end. This mounting scheme leverages the friction of the 3D printed parts, removing the need for mounting hardware and allowing for easy removal of the gripper from the chassis if necessary.

Upon final assembly of the ground chassis, it was realized that the locomotive motors were unable to provide the precise position control necessary for orienting the vehicle with respect to the sample on the ground. It also resulted in the cameras' fields of view being insufficient for making AprilTag detections while navigating on the ground. To mitigate this issue, it was decided that all repositioning with respect to AprilTags would be done aerially. The flight motors provided greater position control than the locomotive motors, and also kept AprilTags within the detecting cameras' frames.

The final ground chassis design is shown in Figure 3.3. This configuration showcases the gripper mounted in its closed position.

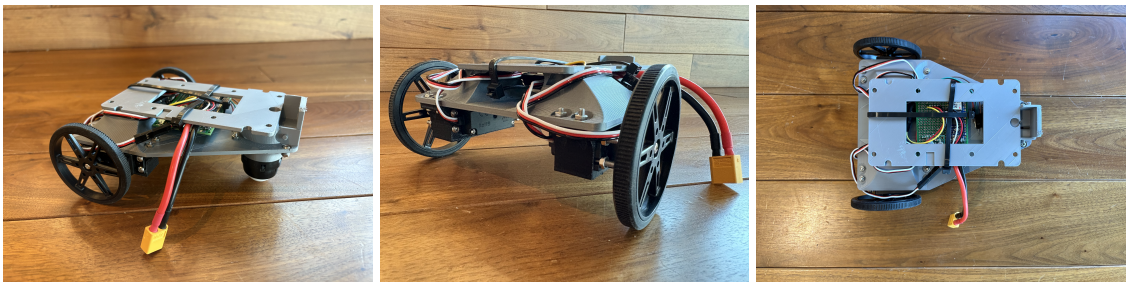


Figure 3.3: A-HAT-TRIC ground chassis

3.2.5 Circuit Design

A simple electronic circuit was designed to siphon power from the aerial platform to the terrestrial platform. The locomotive motors require 5 volts to operate, and the motor for

the gripper needs 3.3 volts. The Arduino Nano ESP32, selected to control the motors for the terrestrial platform, is both capable of operating on 12 volts and providing an output of 3.3 volts for the gripper.

Power for the ground electronics was set up as a parallel circuit. Coming from the flight battery, one branch goes directly to the Arduino Nano ESP32 and the other connects to a voltage regulator board. The voltage regulator board converts the voltage received from the battery to 5 volts, so that the locomotive motors are properly powered. The motor for the gripper receives its power from the 3.3 volts output on the Arduino Nano ESP32.

Each of the motors, regardless of power source, have a signal pin connected to the Arduino Nano ESP32 PWM output for commanding movement. This circuit design is illustrated in Figure 3.4. For implementation onto the actual vehicle, this circuit is soldered onto a perfboard and secured to the lower deck of the ground chassis.

3.3 Integrated Aerial-Terrestrial Vehicle

The complete, assembled A-HAT-TRIC vehicle is shown in Figure 3.5. Piloted flights were conducted to assess the stability of this vehicle when navigating aerially, and there were no substantial impacts observed to flight dynamics. One aspect of vehicle operation that was noticeably affected was rate of battery discharge, as the motors required more power to sustain and equivalent thrust with the increased vehicle weight.

The only notable effects of this vehicle configuration relate to the cameras. The gripper occupies the lower portion of each camera frame, as shown in Figure 3.6. This does provide minor difficulties in performing AprilTag detection, but it is not considerable enough to warrant

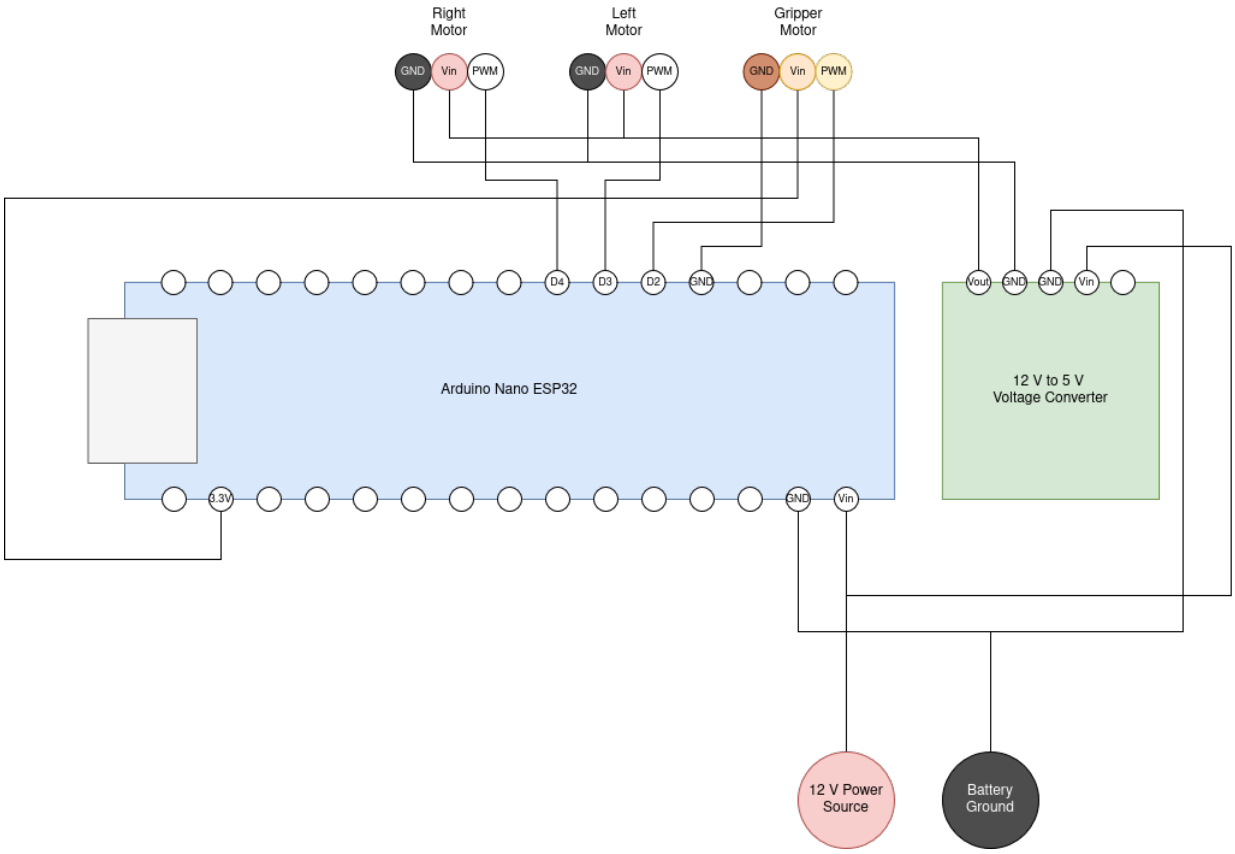
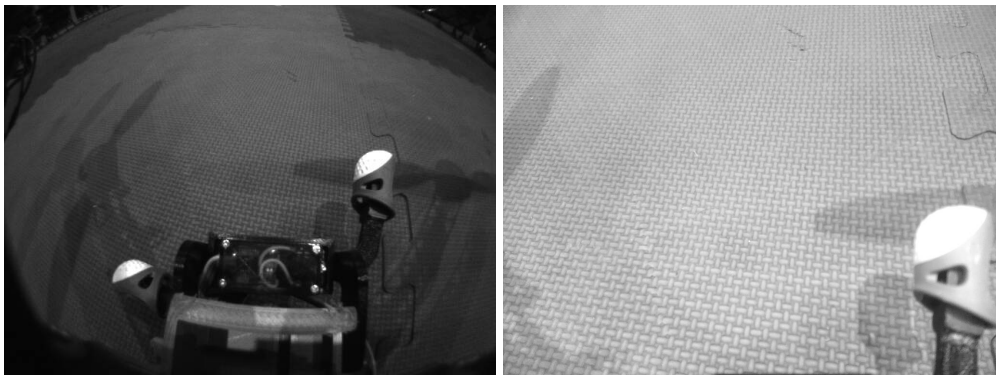


Figure 3.4: Designed electronics circuit for ground system

a design revision. It did, however, significantly impact the flight controller's ability to accurately estimate the vehicle's inertial position during operation. Impacts to the flight controller's VIO algorithm were mitigated by utilizing the rear stereo cameras for position estimation, rather than defaulting to using the tracking camera.

An overview of A-HAT-TRIC hardware, including ground station components, is displayed in Figure 3.7. The battery is omitted from this system, but both the terrestrial and aerial platforms are powered using the same source.

Figure 3.5: Complete A-HAT-TRIC vehicle assembly



(a) Tracking camera (b) Hi-res image sensor

Figure 3.6: Field of view of utilized cameras

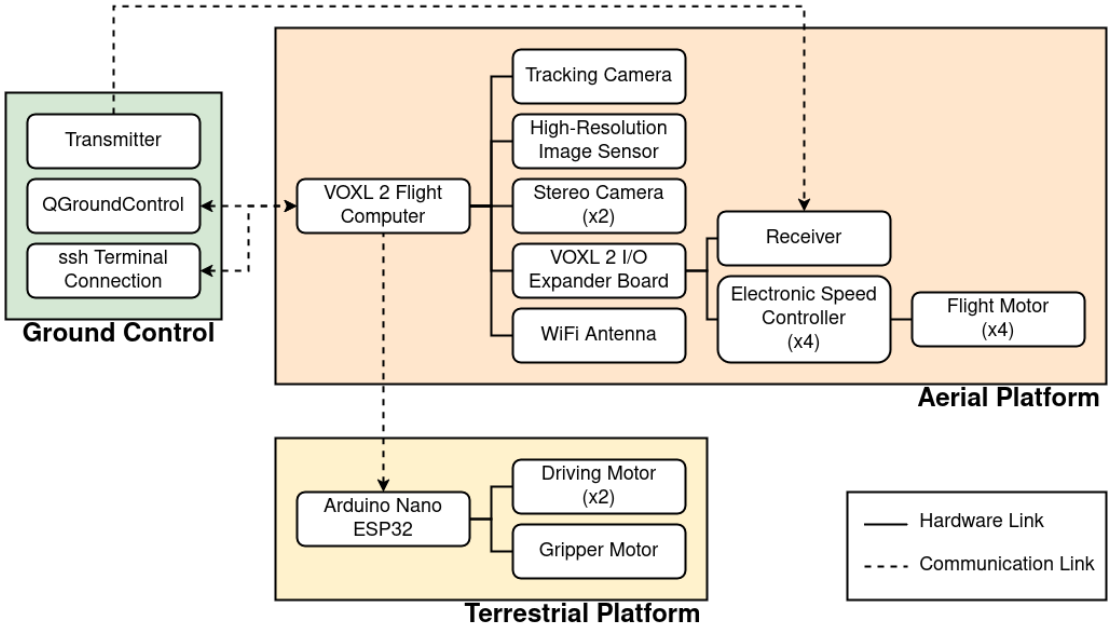


Figure 3.7: Hardware block diagram

Chapter 4: Design and Implementation of Autonomy Architecture

4.1 Autonomous Sample Retrieval Mission

The vehicle begins at the home location before being commanded to autonomously takeoff and assume its prescribed flight altitude. Once the vehicle is sufficiently close to this altitude, it navigates planarly to the starting coordinates of the search area. These coordinates are defined as the bottom left vertex of a rectangle, where the length is oriented to the right along the positive y-axis and the width is oriented upward along the positive x-axis.

After the vehicle navigates to the starting coordinates, it begins to execute the sample search. The trajectory commanded for the search phase of the mission is commonly referred to as a "lawnmower" search pattern. Throughout the search, the vehicle continuously monitors its camera feeds for instances of AprilTag detections that signify a sample. If an eligible AprilTag is detected, the vehicle abandons its search to maneuver toward the fiducial marker.

The process of repositioning for sample collection involves three phases:

1. Planar navigation at the flight altitude to reduce distance from the AprilTag
2. Planar navigation at a reduced altitude to leverage the improved estimates of fiducial marker coordinates to further minimize the distance from the AprilTag
3. Yawing with respect to the sample itself before landing to initiate collection

Upon landing, the vehicle relies only on its terrestrial platform for sample pickup. It drives forward in the direction of the sample before stopping to close its gripper. The vehicle reverses briefly to assess the distance between the coordinates of the AprilTag on the sample. This distance is utilized as the metric for determining whether a sample has successfully been grasped or not. If the distance has sufficiently increased upon driving backwards, the sample is considered grasped. If the distance does not change by a significant amount, then the vehicle has failed to collect the sample.

Whether the vehicle has successfully grasped the sample or not, it is next commanded to takeoff to its flight altitude before navigating back to the home location where it executes a vertical landing. After landing, if the vehicle has successfully collected a sample then it is commanded to deposit it. Sample deposit is confirmed in a similar way to collection, where distance between AprilTag detections is expected to significantly change. If the vehicle has not been confirmed as having grasped a sample, it does not undergo this process of attempting to deposit a sample.

Once these terrestrial tasks have been completed, the vehicle is once again commanded to takeoff and execute the same search pattern as before - but only if there are samples remaining in the search area. If all samples have been accounted for throughout the search and retrieval process, then the mission is complete.

All autonomy for this research is conducted through the algorithm detailed in the following section. The logic dictating the transitions between operational states is visualized in Figure 4.1.

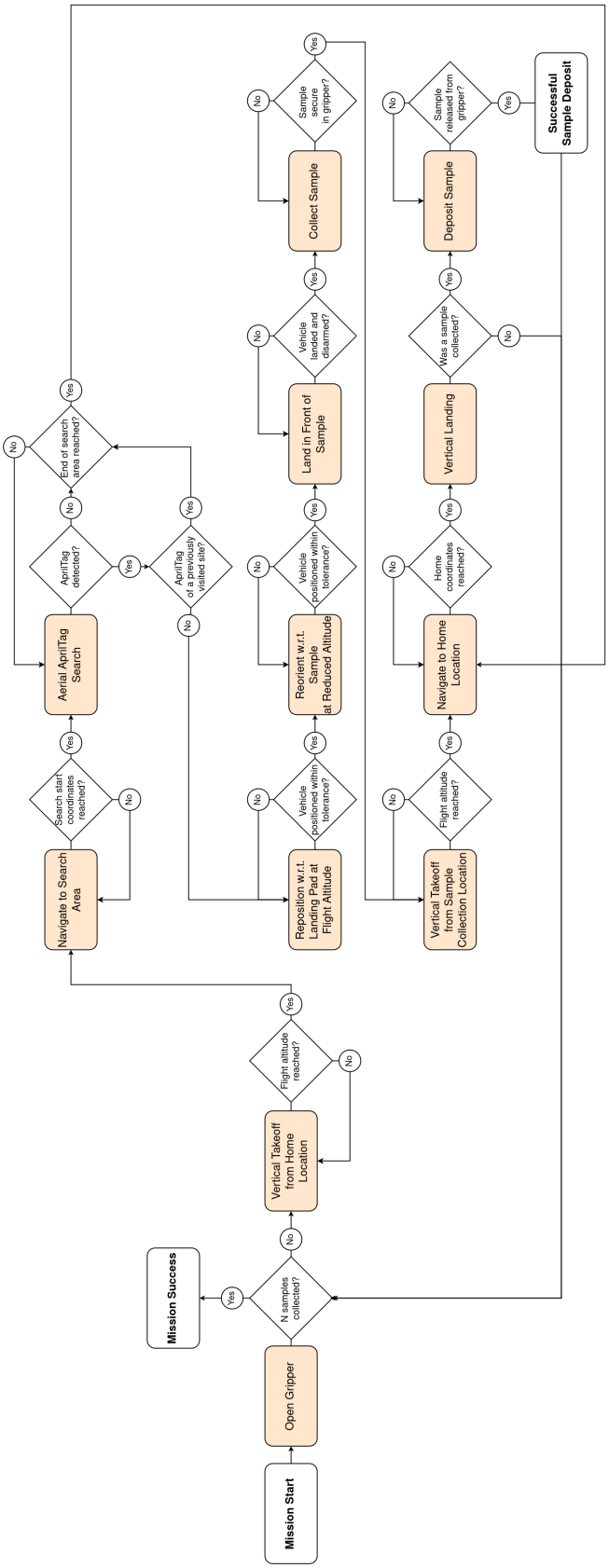


Figure 4.1: A-HAT-TRIC autonomous algorithm

4.2 Autonomy Implementation

Upon boot up, the initial heading of the vehicle is stored. This allows for the search to be conducted relative to the vehicle's initial orientation, rather than the true inertial frame. Since the heading may vary upon boot up, this allows for the search area to be properly defined rather than based on an arbitrary orientation. All setpoints are generated as if the inertial frame is aligned with the vehicle's heading upon boot up, and then transformed prior to setpoint publishing so that they are in true inertial coordinates.

The ROS 2 node that contains the autonomous framework for the sample collection mission requests operator input prior to commanding the vehicle. The prompts ask for the starting coordinate of the search area, the dimensions of the search area, and the number of samples expected. If no input is given then default values are assumed. The boot up screen output, assuming default values, is given below:

```
Prior to launching the vehicle, information is needed to define
the search area.
The x-axis is defined as parallel to the heading of the drone
upon boot up.
The y-axis is defined as positive to the right of the drone's
heading upon boot up.
All units are in meters.


The length of the search area is defined as parallel to the
y-axis.
```

The width of the search area is defined as parallel to the x-axis.
The starting coordinates of the search area as defined as the 'bottom left' corner.

When prompted, please enter the length, width, and starting coordinates for the search area.
Additionally, the number of samples in the search area must be defined.

(Press enter key when finished typing each entry.)

Starting x-coordinate for search area (in meters):
Starting y-coordinate for search area (in meters):
Length of search area (in meters):
Width of search area (in meters):
Number of samples expected:

Search area of length 2m and width 1m described.
Initial coordinates given of (1, 0)m.
The vehicle will be searching for 1 sample(s) in this search area.

After the necessary information is obtained, the ROS 2 node begins the execution of its autonomous algorithm.

There are three main ROS topics utilized in the autonomous framework, two are subscribed to in the ROS 2 node and the third is published to. Vehicle position data, relative to the inertial frame, is obtained from the PX4-defined ROS topic `/fmu/out/vehicle_local_position` [21]. AprilTag coordinates are received from the modified `voxl-mpa-to-ros2` service, as described in Appendix A. The algorithm subscribes to two topics containing this information: `/track_detections/tagpose_wrt_body` providing localization via the tracking camera and `/hires_detections/tagpose_wrt_body` localizing the AprilTag using the hires image sensor. Commands for aerial maneuvering are published to the PX4-defined ROS topic `/fmu/in/trajectory_setpoint` [41]. Commands for ground motors, including the gripper, are sent to the Arduino Nano ESP32 from the VOXL 2 flight controller. This communication protocol is documented in Appendix C. This overall software structure for mission execution is displayed in Figure 4.2.

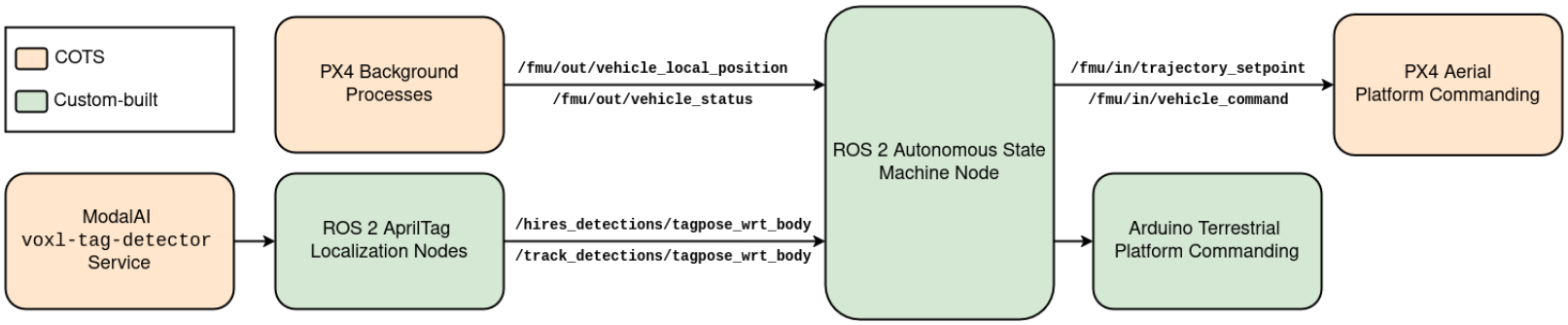


Figure 4.2: Software architecture as executed on primary computer

Certain parameters that are used across multiple states are defined globally as constants. These include the flight altitude, planar aerial speed, vertical speed, loop rate, and distance between passes in search. Another important aspect of this algorithm is the internal tracking of the samples themselves. After the operator indicates the number of samples to be collected, the code creates two arrays: one array stores the AprilTag ID of each expected sample and the other array utilizes ones and zeros to indicate whether a sample has been collected (1) or not (0). For this purpose, it is assumed that each sample has a unique AprilTag ID and that they are sequential starting with AprilTag ID 1.

The algorithm tracks states through the use of an integer flag. Each flag, and its corresponding state, is given in Table 4.1. State transitions are handled logically, largely based on data available by subscribing to various ROS topics. Each state, and its corresponding transition logic, will be explained in the following subsections.

Integer Flag	**State Behavior**
1	takeoff from home location
2	fly to search area
3	execute sample search
4	sample found, precision hover above landing site
5	aerially reorient and land vehicle relative to sample
6	pick up sample
7	takeoff from sample location
8	fly to home location
9	land at home location
10	deposit sample

Table 4.1: States and corresponding integer flag

4.2.1 Vertical Takeoff

Prior to takeoff, it is required that the operator arm the vehicle for flight. This is necessary as the ModalAI infrastructure prohibits the VOXL 2 from arming itself autonomously. Aside from providing information for defining the search, this is the only instance during the execution of this autonomous algorithm that the user is required to interfere manually. The ROS 2 node outputs a request to arm the vehicle to the command window. Once the vehicle is armed, the algorithm proceeds to command the vertical takeoff.

The algorithm utilizes the vehicle's x- and y-coordinates from its position prior to takeoff to maintain purely vertical motion. This allows the takeoff routine to be executed from any position in the inertial frame. It commands the vehicle to increase its altitude, using the prescribed vertical velocity as a guide and iterating the z-position using the vertical velocity and the period of the autonomous control loop. Once the vehicle has achieved an altitude within 5 centimeters of the target altitude, takeoff is considered successful.

If the vehicle performed this vertical takeoff in the state described by integer flag 1, the algorithm proceeds to fly to the defined starting coordinates for the search area. If the vehicle performed this vertical takeoff after collecting a sample, the algorithm switches the integer flag to 8 and the vehicle will return aerially to the home location.

Vertical takeoff simulation results from five trials of ascending from ground level to an altitude of 1 meter are shown in Figure 4.3. It can be observed that the simulated vehicle initializes its z-coordinate as though it were located below the surface, a phenomenon not observed onboard the actual vehicle. Due to this discrepancy, the violated assumption that the vehicle initializes somewhere along the negative z-axis affected perturbations to the purely vertical takeoff

commands sent. The real vehicle does not reflect this unsteadiness in its performance.

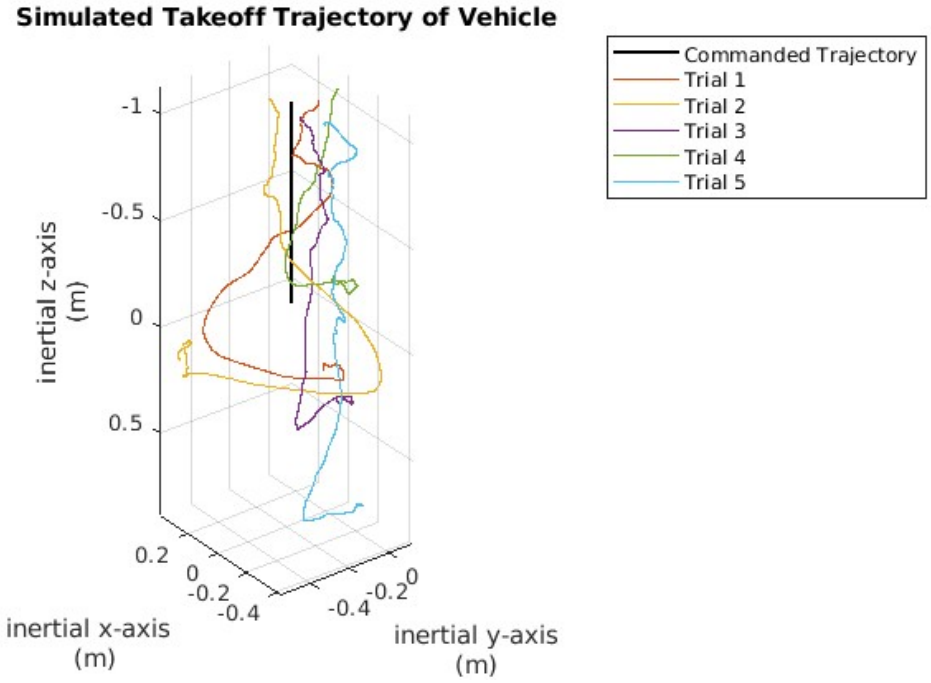


Figure 4.3: Simulation trials for vertical takeoff to altitude of 1m

4.2.2 Navigation to and from Home Location

This state requires that the drone be at its defined flight altitude. Thus, all navigation is planar in this state. It utilizes the starting (x_1, y_1) and ending (x_2, y_2) x- and y-coordinates to compute the total distance D to travel between the coordinate pairs. Then, the planar velocity vector (v_x, v_y) is computed for commanding:

$$v_x = V_L \frac{(x_2 - x_1)}{D} \qquad v_y = V_L \frac{(y_2 - y_1)}{D}$$

Where V_L represents the defined aerial linear velocity for the vehicle.

The commanded planar position is iterated using this velocity vector and the period of the

autonomous control loop. The vertical position is held constant at the flight altitude, and there is zero commanded vertical velocity. Once the vehicle is within 10 centimeters of its goal, a state transition will occur.

If this planar navigation is commanding the vehicle to fly to the start of the search area, the integer flag is switched to 3 and the sample search begins. If, instead, the vehicle is navigating back to the home location, the integer flag will switch to 9 and the vehicle will land.

This planar aerial navigation scheme was tested extensively in simulation before deployment on to the actual vehicle. Figure 4.4 showcases the results from five separate trials conducted, where the vehicle was commanded to navigation from the home location $(0, 0)_I$ to the starting coordinates for the search area $(-2, 1)_I$. Compared to the vertical takeoff simulation data, the simulated flight performance is much closer to the commanded trajectory.

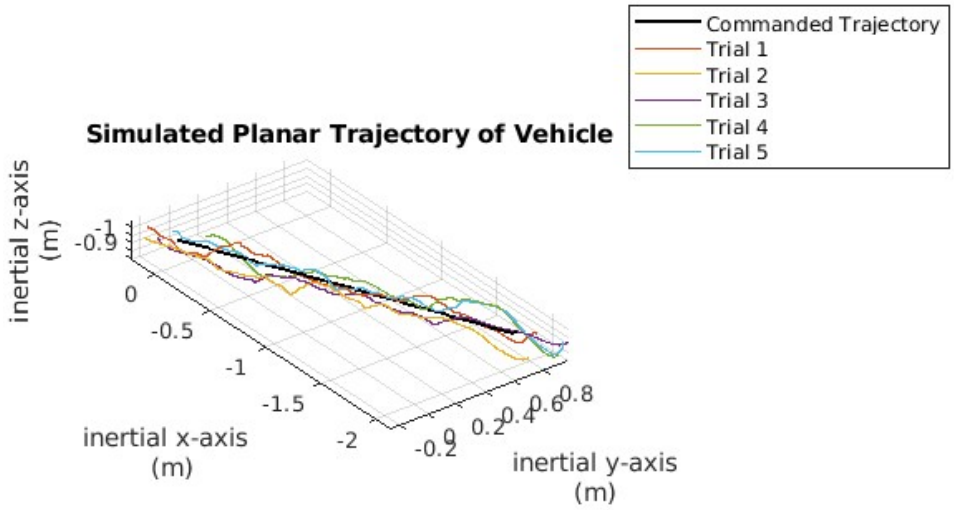


Figure 4.4: Simulation trials for planar aerial navigation from home to search area start

4.2.3 Aerial Search Pattern

The sample search utilizes a lawnmower pattern on account of its simplicity to implement and ability to provide adequate camera coverage of the search area. An example of a lawnmower search pattern generated using this autonomous framework is shown in Figure 4.5 to provide context for the following discussion. The flight altitude is held constant throughout the execution of the aerial search.

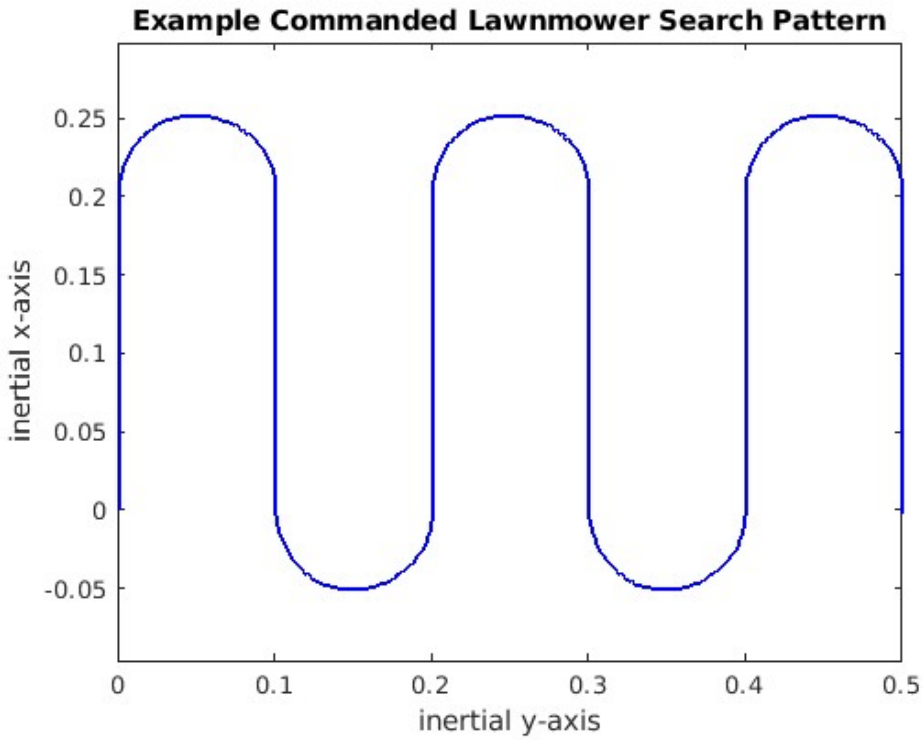


Figure 4.5: Example generated search pattern

When turning, the vehicle travels along a semicircle of diameter 0.5 meters. This value was selected as it provided adequate coverage of the search area without resulting in an abundance of overlap. The angular velocity of the vehicle throughout the turn is calculated using the aerial planar velocity and defined distance between passes in the search area.

The algorithm for generating the trajectory setpoints for the search pattern is governed by integer flags that dictate the vehicle's position along the search path. These integer flags, and their corresponding definitions as they pertain to trajectory generation, are provided in Table 4.2. Transition logic between trajectory generation states will be explained throughout this section.

Integer Flag	**Trajectory Description**
-1	hover at start coordinates
0	move along positive x-axis
1	turn clockwise
2	move along negative x-axis
3	turn counter-clockwise
4	hover at end coordinates

Table 4.2: Search pattern internal integer flags

At the initialization of the sample search state, the vehicle is held at its current position for a brief period of time to ensure a smooth transition between states. After 1 second has elapsed, the search transitions to integer flag 0. This internal transition does not require any information about vehicle performance to occur.

In the internal search pattern state 0, the vehicle proceeds to navigate solely in the positive x-axis direction. In this internal state, the vehicle's y-coordinate is held constant. Only the x-position is iterated utilizing the defined aerial planar velocity and the loop rate. Once the vehicle's actual x-position has surpassed the positive boundary of the search area, as defined by its width, the internal flag is changed to 1 and the vehicle begins to make a clockwise turn.

The angle of the turn is iterated throughout the generation of the semicircle arc with respect to the angular velocity and loop rate. The x- and y-coordinates are computed in the following manner:

$$x = r\sin\theta + x_0 \qquad\qquad y = r\cos\theta + y_0 + r$$

Where r represents the radius of the semicircle, $\theta\epsilon[0, \pi]$ radians is the angular position of the vehicle in the arc, and (x_0, y_0) are the current planar coordinates of the vehicle. The commanded velocity of the vehicle is then simply the derivative of these expressions, and the acceleration being the derivative of the velocities. The commanded heading of the drone is then calculated such that it is aligned with the velocity vector of the vehicle. Once the vehicle has reached an angular position in the arc of π radians, the semicircle is considered complete and the internal state transitions to motion along the negative x-axis, internal flag 2.

The trajectory setpoints for flight along the negative x-axis are generated opposite of those setpoints commanding motion in the positive x-axis direction. The vehicle transitions to a counter-clockwise turn once it has crossed the negative boundary of the search area, as defined by its width. Generating y-coordinates and vehicle heading for the counter-clockwise turn is identical to the generation for the clockwise turn. Calculating x-coordinates requires negating the radial term of the clockwise turn x-coordinate generation formula. After the vehicle has achieved an angular position of π radians in the semicircle arc, the vehicle transitions back to commanding motion in the positive x-axis direction.

If, at any point during the commanding of the search trajectory, the vehicle surpasses the boundary imposed by the length of the search area without having detected a viable AprilTag, it is commanded to maintain its current position before the autonomous algorithm transitions the vehicle into the state described by integer flag 8 and returns to the home location.

However, if the vehicle detects an AprilTag while executing its search, it may instead transition to the state described by integer flag 4 and conduct a precision landing. If the detected AprilTag ID is both included in the array storing expected AprilTag IDs and not indicated as having been collected, the vehicle will change states to land. If the AprilTag detected fails these

checks, then the vehicle continues its search.

The results of five separated simulated search trajectory demonstrations are displayed in Figure 4.6. It is observed that the data for the simulated vehicle performance experiences regular deviations from the commanded trajectory. This characteristic is exhibited onboard the physical A-HAT-TRIC vehicle as well, but does not noticeably impact flight performance.

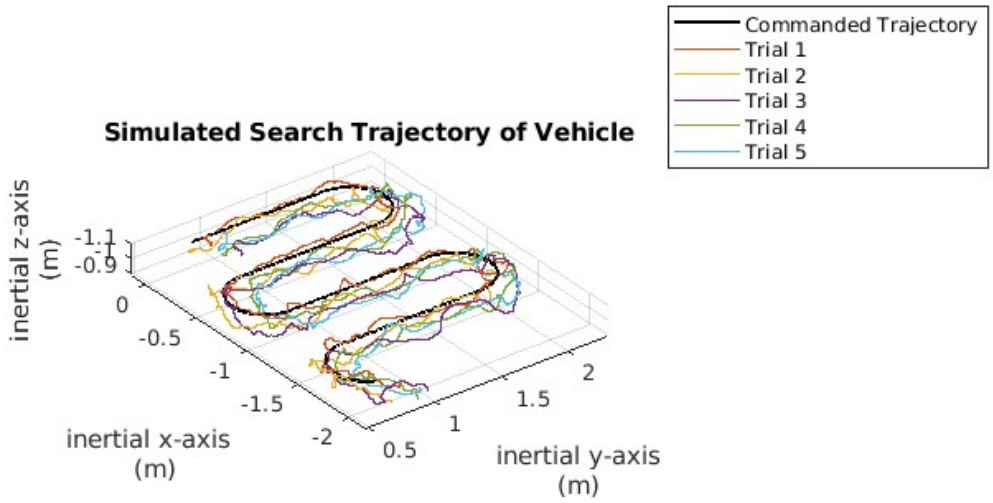


Figure 4.6: Simulation trials for aerial search pattern trajectory generation

4.2.4 Precision Hover Above Sample Location

The AprilTag ID detected in the search state is utilized exclusively for repositioning the drone relative to the marker before descending to a hover. Any detections made of other AprilTags in this state are rejected. The target AprilTag for this state is located in the inertial x-y plane at zero altitude.

This state uses the same approach as the aerial planar navigation state for generating

trajectory setpoints. The vehicle's current position is taken as the starting point and the inertialized AprilTag position is set as the endpoint. Since AprilTag detections are made by both the hires image sensor and tracking camera, there is logic implemented to determine which camera has most recently detected the AprilTag by comparing the detected AprilTag ID to the desired value. If both cameras have detected the correct AprilTag, preference is given to the coordinates of the marker provided by the hires image sensor.

The AprilTag coordinates themselves are then utilized to determine whether the values provided are the result of an active or a previous detection. To accomplish this, the previously detected AprilTag coordinates are stored for comparison with the new set returned. In the case that no new detections are made, the ROS 2 message will contain the exact same coordinates as what has been previously detected. If this occurs, the vehicle is commanded to move along the trajectory defined by the most recent, unique detection for up to 1.5 seconds. If the vehicle has not made an updated detection of the AprilTag in this time, it will preserve its current planar coordinates and rotate about itself to regain sight of the marker.

For each updated detection of the target AprilTag, the algorithm utilizes these new coordinate estimates to adjust the vehicle's planar trajectory in the direction of the marker. Once the vehicle is within a tolerable distance of the AprilTag, the algorithm switches to integer flag 5 for landing.

4.2.5 Aerial Positioning and Landing Relative to Sample

Prior to any further planar repositioning, the vehicle decreases its altitude so that it can obtain a more accurate estimate of the AprilTag's inertialized coordinates. Using the same framework as described for the previous state (integer flag 4), the vehicle navigates planarly at

the reduced altitude to further reduce its distance from the AprilTag at the nominal landing site. After the vehicle has achieved a position within the bounds of tolerance, the algorithm switches its dependency to the AprilTag marking the sample itself. The samples are always denoted with AprilTag ID 0, regardless of quantity or location of the samples.

Using the sample's AprilTag, the algorithm commands the vehicle to minimize its heading with respect to the tag. When the tolerance for yaw has been met, the vehicle readies itself for landing. The vehicle first navigates along the negative y-axis in the body frame for 5 centimeters, which was found to be a consistent bias in AprilTag position estimation. Then, it hovers in its current orientation for less than 1 second to stabilize before touching down for sample collection. Once the hold period has elapsed, the vehicle is commanded to land by iterating vehicle position along a vertical path. Upon landing, the integer flag is switched to 6 for sample collection to occur.

4.2.6 Sample Grasping

Gripper actuation occurs in one of two states: sample collection (integer flag 6) or sample deposit (integer flag 10). When collecting the sample, the float value corresponding to closing the gripper is sent to the Arduino Nano ESP32. For depositing the sample, the float value corresponding to opening the gripper is sent to the microcontroller.

In both states, the position of the AprilTag after the griper command is sent is saved. This is done to create a baseline for comparison in determining whether the sample has been either successfully collected or deposited. After the gripper has been commanded and the AprilTag coordinates have been defined, a command is sent to the terrestrial motors to move the vehicle

backwards away from the sample.

After the vehicle has backed away for 2 seconds, the updated AprilTag coordinates are collected. For sample collection, if these updated coordinates are a distance less than 5 centimeters from the initial AprilTag coordinates, the sample is considered successfully collected. For sample deposit, if these updated coordinates are a distance greater than 5 centimeters from the initial AprilTag coordinates, the sample is considered successfully released.

In the case of sample collection, if this check fails then the vehicle is commanded to drive forwards for 3 seconds to re-attempt collection. If this condition is satisfied, then the autonomous algorithm transitions to integer flag 7 to initiate the return to the home location.

If, instead, the vehicle is meant to deposit the sample and this check fails, the vehicle will repeat its attempt to open the gripper. The subsequent evaluation of the AprilTag's position after driving in reverse is then again utilized to determine if the vehicle was successful in releasing the sample. This cycle repeats until the sample is successfully deposited. If the sample cannot be successfully deposited, the mission must be terminated prematurely. However, if the sample is successfully deposited, there are two actions the algorithm performs. First, the Boolean operator indicating the collection status of a specific sample, as indicated by its AprilTag ID, is updated to reflect is has been successfully collected. Next, the algorithm determines whether there are any samples left to be collected. If all samples have been collected then the mission is successful and is terminated. If there are samples remaining, the integer flag is changed to 1 and the search repeats.

4.2.7 Vertical Landing at Home Location

A built-in command for a vertical landing from the PX4 framework is utilized for landing at the home location. This leverages the PX4 ROS 2 topic `/fmu/in/vehicle_command` [42] to send the required information to conduct a simple vertical landing. In the `command` field, the PX4 parameter `VEHICLE_CMD_NAV_LAND` is sent.

This custom PX4 command automatically maintains the vehicle's current x- and y-coordinates while decreasing altitude for landing. The only additional argument is the desired heading that the vehicle should maintain during landing. This value is selected as the heading of the vehicle upon boot up. This aligns the vehicle's heading with what will be commanded upon takeoff from the home location, allowing for a more stable takeoff.

This state is considered complete once the vehicle is disarmed. This occurs automatically upon touchdown, and the algorithm is able to access the arming status of the vehicle through the PX4-defined ROS 2 topic `/fmu/out/vehicle_status` [43]. Once the algorithm detects that the vehicle is disarmed, it will either transition to immediately taking off and repeating the search (integer flag 1) or depositing a sample (integer flag 10). The algorithm indicates whether a sample has been collected through Boolean logic, where a variable is assigned 0 if the gripper does not contain a sample or is assigned 1 if a sample is grasped. Thus, it can be determined through a simple conditional which state to proceed to from integer flag 9.

The results of five separate simulation runs where the vehicle was commanded to perform a vertical landing at the home location are shown in Figure 4.7. The simulated vehicle performance agrees much more closely with the commanded trajectory in the vertical landing scenario rather than in vertical takeoff (Figure 4.3). However, these data points do still indicate that the vehicle in

its landed configuration is sub-surface. This is not observed on the actual A-HAT-TRIC vehicle.

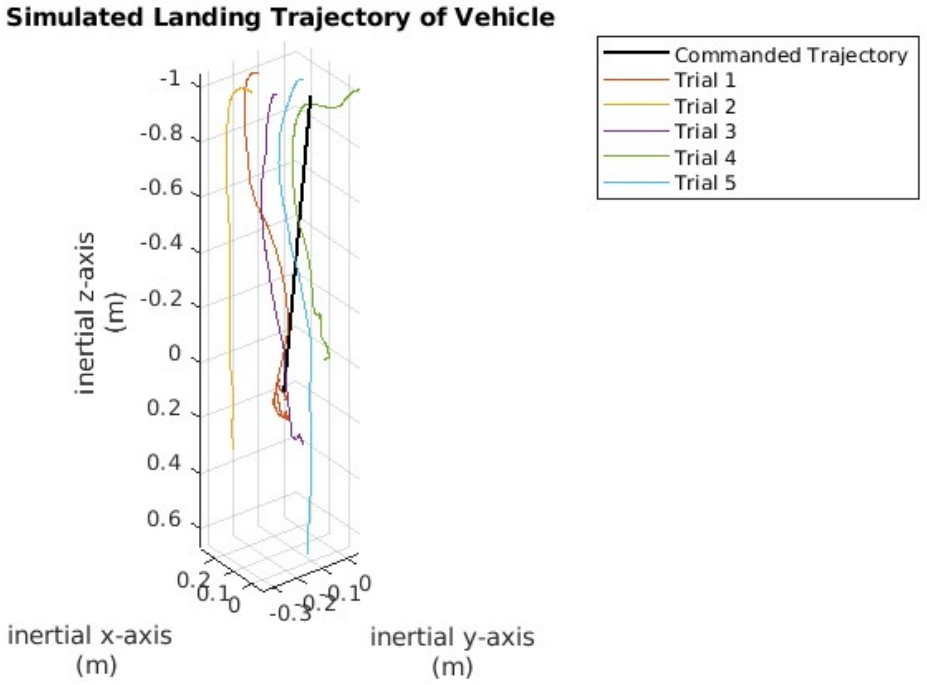


Figure 4.7: Simulation trials for vertical landing at home location

Chapter 5: Experimental Evaluation

5.1 Methodology

5.1.1 Laboratory Environment

Testing with the A-HAT-TRIC vehicle was performed indoors at the University of Maryland's Brin Family Aerial Robotics Lab, pictured in Figure 5.1. This facility features 430 square feet of netted space for testing aerial platforms, and is maintained by the Maryland Robotics Center.

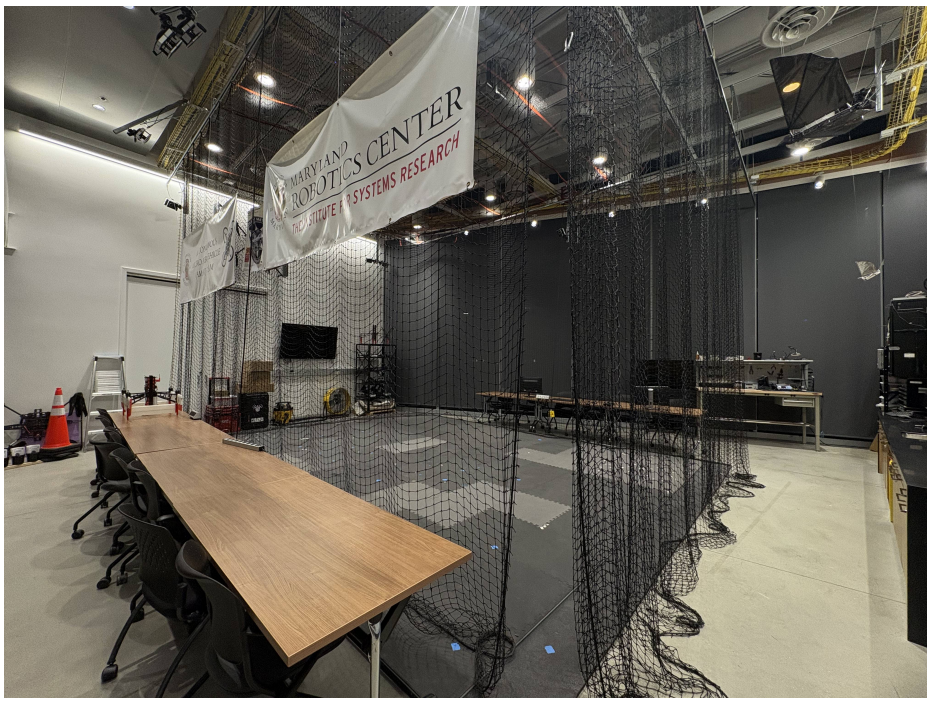


Figure 5.1: University of Maryland indoor netted flight facility

The AprilTags used for detection, of the "sample" and "landing" variety, are shown in

Figure 5.2. The sample, in the shape of a 2.5 inch cube, was constructed specifically for the A-HAT-TRIC experiments. It features an oversized top face, which provides extra white space around the AprilTag to improve both the quality and quantity of aerial detections.

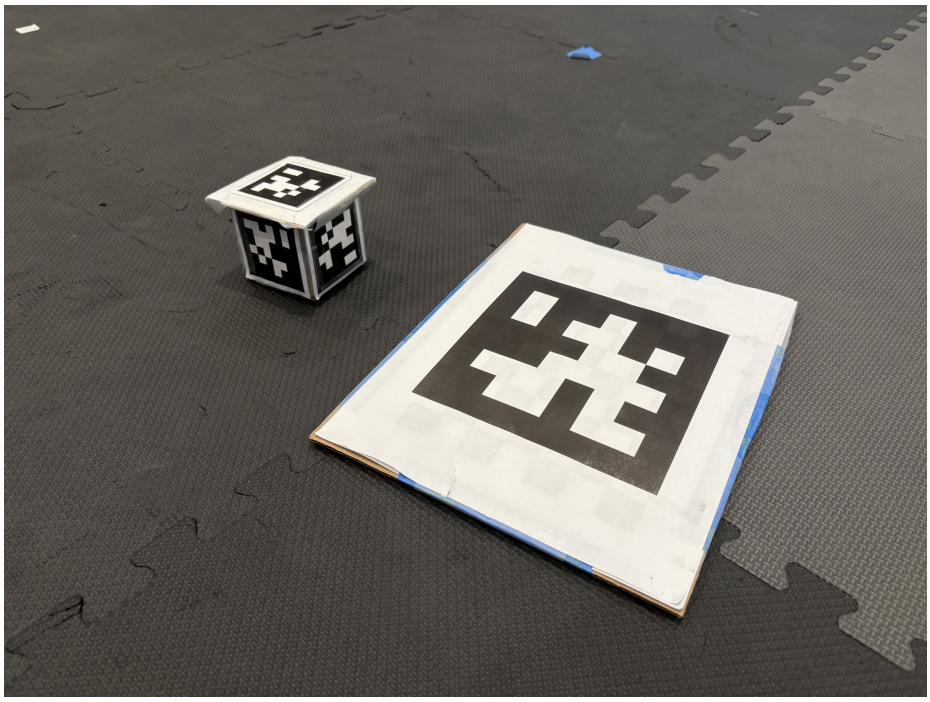


Figure 5.2: AprilTags as utilized for indicating landing sites (right) and samples (left)

5.1.2 Vehicle Configuration

Physically, the vehicle is configured as previously shown in Figure 3.5 for operation in the laboratory environment. An image of A-HAT-TRIC in the netted space, including AprilTags for scale, is given in Figure 5.3.

For operations, a ground station, the flight computer, and the microcontroller must all be connected to the same wireless network. The flight computer's ability to provide a software-enabled access point was leveraged for this purpose. For the ground station, the free open source software QGroundControl [44] was utilized. Additionally, a transmitter was required both for

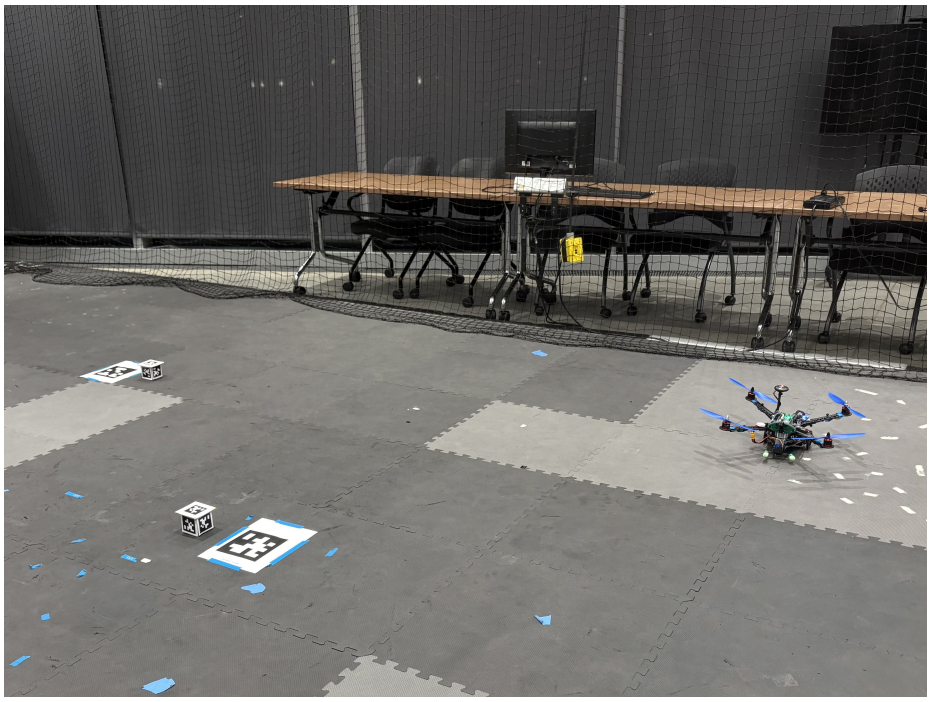


Figure 5.3: A-HAT-TRIC in laboratory test environment

arming the vehicle and as a safety measure in case the autonomous software framework failed or flight performance degraded and an emergency piloted landing was necessary.

5.1.3 Mission Profile

The mission profile designed to assess the performance of the A-HAT-TRIC vehicle is parameterized in Table 5.1. A simple visualization of the laboratory configuration is given in Figure 5.4, where the boundary of the search area is represented by the dashed-line rectangle.

Starting Coordinates	**Search Area Length**	**Search Area Width**	**Number of Samples**	**Sample 1 Location**
(1, 1)	3m	1m	1	(1.8, 2.1)

Table 5.1: Overview of the mission parameters for the A-HAT-TRIC vehicle

Multiple samples were not included in this evaluation, as it was determined that the developed algorithm was not entirely sufficient for such an operation. To implement additional samples in

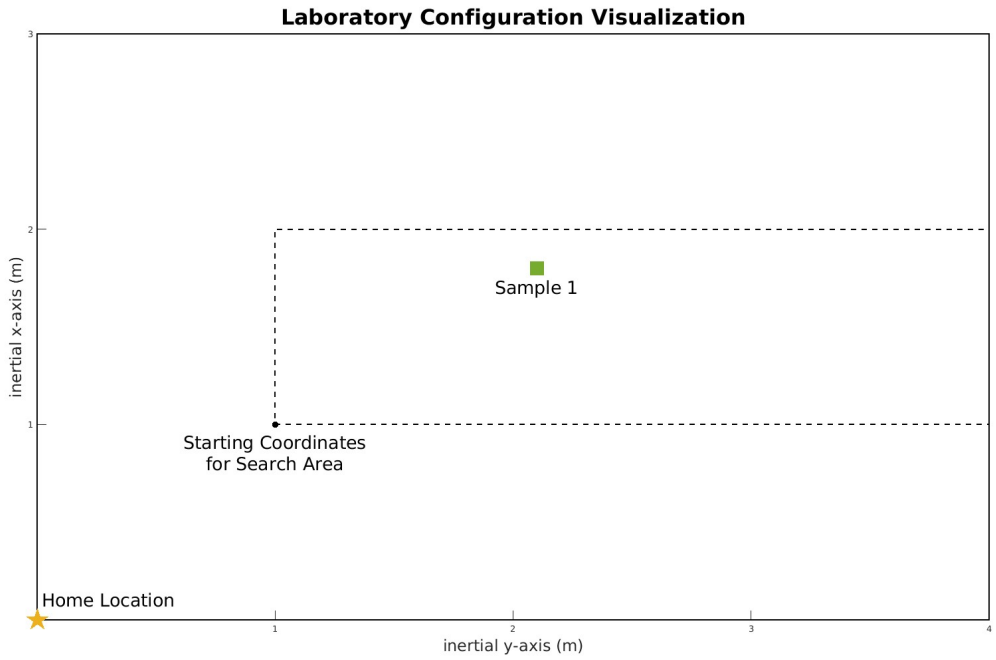


Figure 5.4: Diagram of laboratory setup for A-HAT-TRIC experimentation

the search area, the software would need to be reconfigured to interface with the VIO algorithm on the flight computer. Although this is a feasible task to complete, it was not implemented for this thesis and is considered a next step in the improvement of this autonomous framework.

5.2 A-HAT-TRIC Performance Analysis

Vehicle performance was assessed in two stages. First, the vehicle was tested for its ability to autonomously generate and follow a desired trajectory. Then, an end-to-end demonstration was performed to determine overall vehicle performance for autonomous search, detection, and collection.

5.2.1 Testing of Autonomous Trajectory Generation

To test both trajectory generation and trajectory-following of the vehicle, the parameters given in Table 5.1 were provided to the autonomous algorithm. All logic related to AprilTags in the algorithm were disabled for the purpose of collecting this information.

Utilizing the available telemetry from the vehicle, a 3-dimensional plot of the generated trajectory and its execution was constructed as shown in Figure 5.5. A planar, top-down view of this same data is also available in Figure 5.6. The rectangular outline at $z = 0$ is representative of the total area available for operations in the indoor netted facility. The dashed-line bounding box at the flight altitude, $z = -1$, is the user-defined search area. The yellow star marker at $(0, 0)$ represents the home location for the mission.

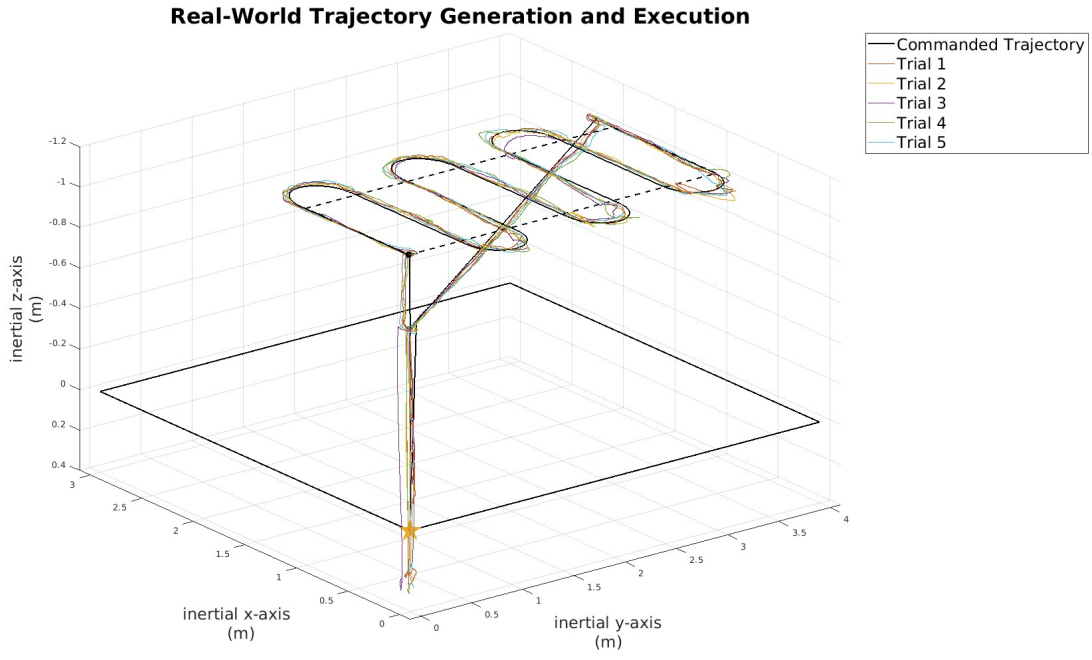


Figure 5.5: 3-dimensional view of A-HAT-TRIC trajectory-following in inertial space

The overlay of the vehicle position estimates from five trials are shown on the plots as well.

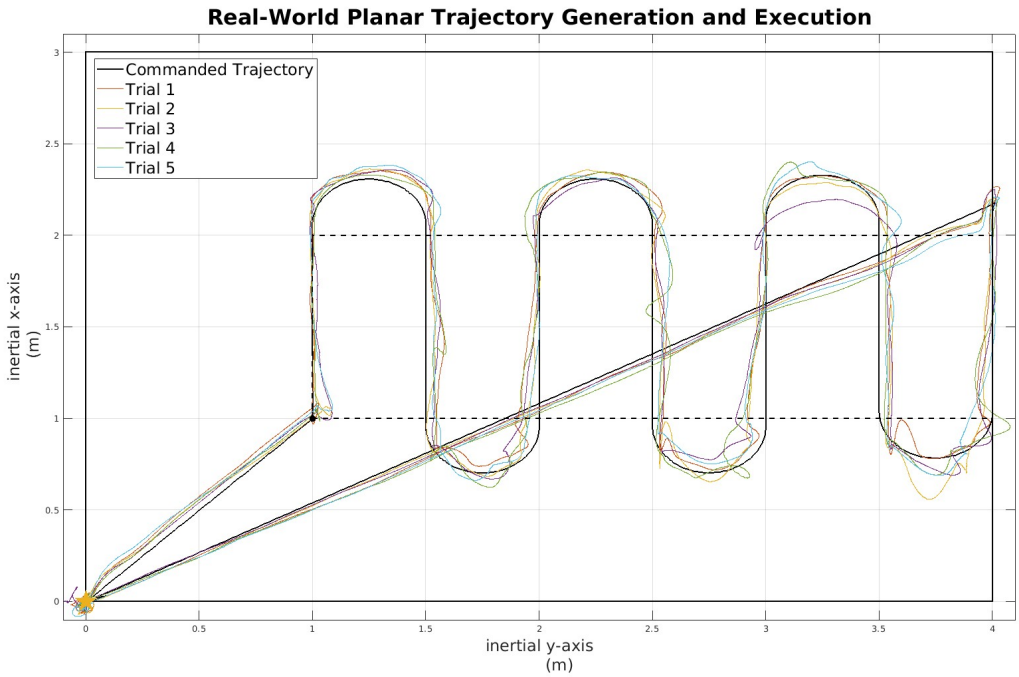


Figure 5.6: Planar view of A-HAT-TRIC trajectory-following in inertial space

Each trial is exceptionally consistent in planar linear motion. Deviations in position for linear motion can largely be attributed to the VIO-based position estimate utilized for computing the endpoints for the straight-line trajectory. The position estimates are collected at a data sample rate of about 100 Hz, significantly higher than that of the trajectory setpoint commanding which is published at a rate of around 10 Hz. This allows for small differences in position at any given moment to propagate through the remaining trajectory generation framework.

The estimated time required for A-HAT-TRIC to complete the desired trajectory in each of the five trials is given in Table 5.2. The vehicle was relatively consistent in its execution, with an average time of 130.6 seconds.

This data demonstrates that the autonomous algorithm devised for trajectory planning and following for the A-HAT-TRIC vehicle is successful. The consistency and accuracy of the data points across all five trials conducted, for both path-following and time elapsed, emphasizes the

Trial	Completion Time
1	131 s
2	133 s
3	128 s
4	129 s
5	132 s

Table 5.2: Time to complete generated trajectory

satisfactory performance of the vehicle at this reduced level of autonomy.

5.2.2 End-to-End Autonomous Sample Collection Mission Demonstration

To demonstrate the vehicle's ability to autonomously detect and collect a sample from the laboratory environment, the AprilTag detection framework was implemented back into the vetted trajectory generation code. For this test, the same parameters given in Table 5.1 were utilized.

The vehicle's position estimates in inertial space from the sample collection test are plotted in Figure 5.7. In this plot, the location of sample 1 is denoted by a green square marker. The plot shows all steps in the autonomous algorithm process (Figure 4.1) required for collecting and depositing a single sample.

It is observed that the vehicle follows the desired search trajectory, encountering the sample before its second pass across the width of the search area. The reduced altitude for more accurate repositioning with respect to the AprilTags is also apparent in this plot, with the cluster of position estimates just prior to landing near the sample representing the vehicle's final attempt to successfully align its heading with the sample. The position estimates at $z = 0$ represent the terrestrial motion commanded for actually collecting the sample. Then, the vehicle performs a vertical takeoff to navigate back to the home location and deposit the sample, as pictured in Figure 5.8.

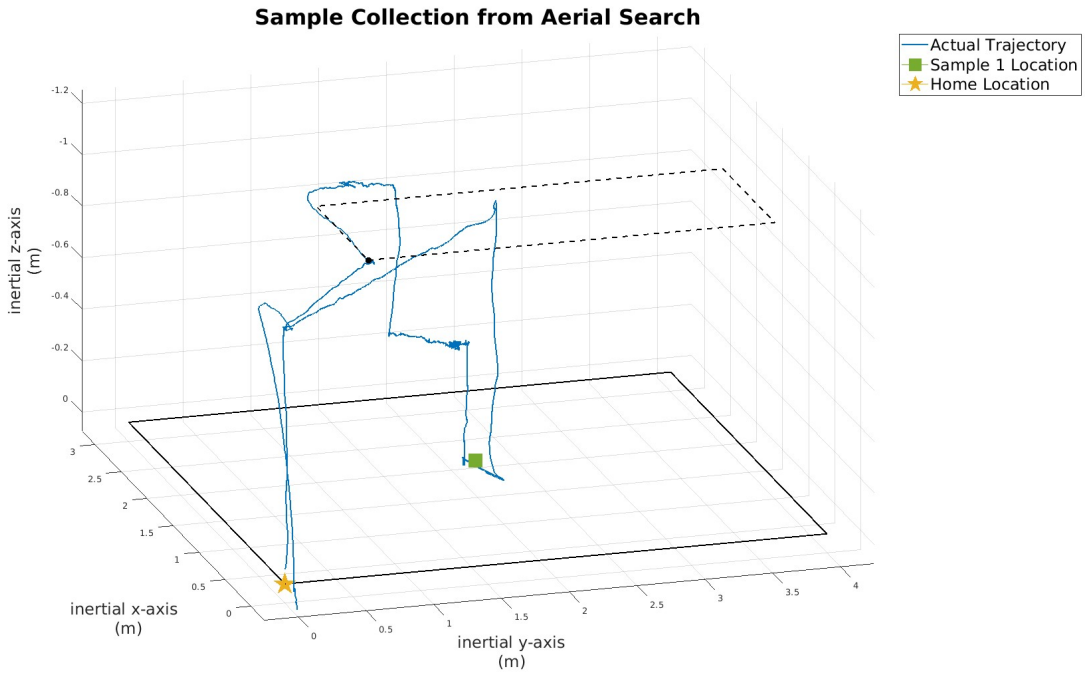
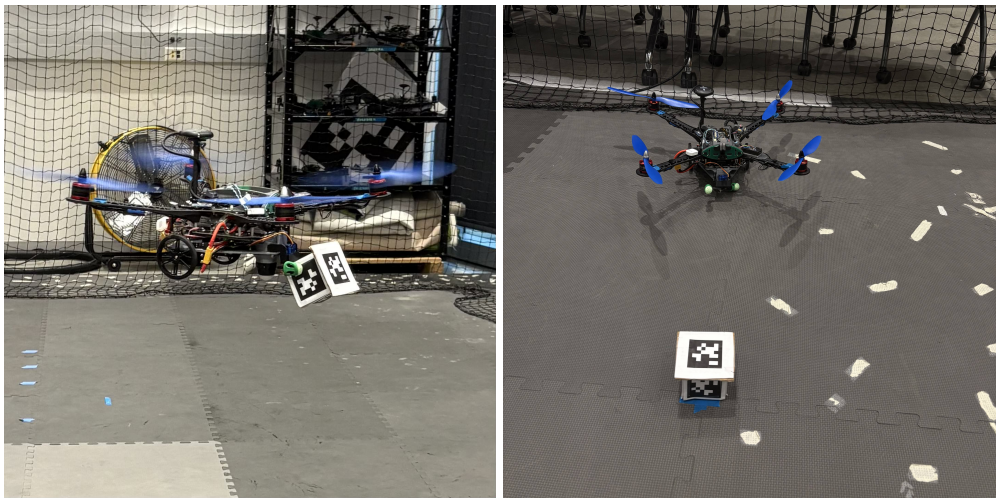


Figure 5.7: 3-dimensional view of A-HAT-TRIC sample collection trajectory in inertial space



(a) A-HAT-TRIC in flight with sample (b) A-HAT-TRIC with deposited sample

Figure 5.8: A-HAT-TRIC demonstrating sample collection capabilities

Apart from visual observations, sample collection can also be ascertained by the telemetry received from the vehicle. Comparing the vehicle's position to the camera estimate of the AprilTag's position relative to the A-HAT-TRIC body frame can determine whether the sample

has been successfully grasped or not. If a sample is contained by the gripper, it is expected that the coordinates estimating its position relative to the body frame would be constant. Otherwise, that would indicate the sample is not steadily grasped by the vehicle. This concept is shown by the plotted telemetry in Figure 5.9.

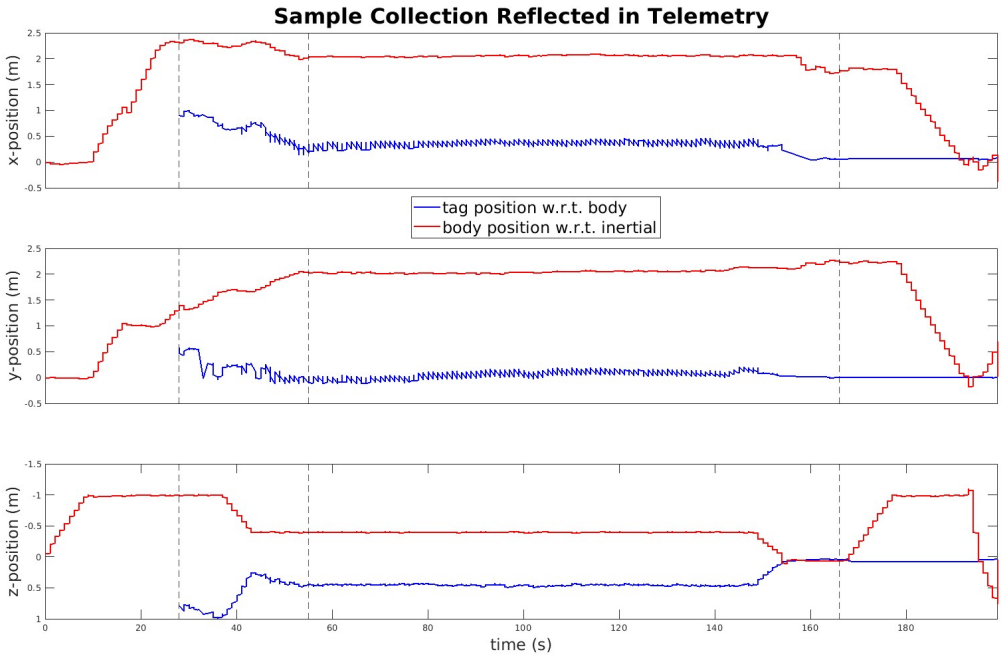


Figure 5.9: Position of sample and position of vehicle over time

The left-most dashed line in the plot represents the time at which the first detection of the landing site AprilTag (ID 1) occurred. The next vertical line on the plot is indicative of when the algorithm switches to utilizing coordinate estimates for the sample's AprilTag (ID 0) to properly align the vehicle's heading with the sample prior to landing. This phase of the mission is extremely time consuming to achieve an optimal orientation with respect to the sample, as it requires both high accuracy and high precision for successful sample collection.

The third vertical line on the plot represents the moment at which the sample was grasped. After this instance in time, it is observed that the sample's estimated position with respect to

the body frame is relatively constant while the vehicle returns to the home location. This is the indication that the sample has successfully been grasped by the vehicle. After landing and depositing the sample, the cameras lose sight of the AprilTag, so no updated measurement of its position in the body frame can be made. This makes it difficult to track the successful deposit of the sample with available telemetry alone. However, in this test, the algorithm was able to accurately ascertain it had indeed released the sample.

To assess vehicle performance, five trial runs were executed. The results of each trial, displayed analogous to the previously-explored plots in this section, are given in Appendix D. These demonstrations were performed using the parameters in Table 5.1. All trials were successful in retrieving the sample, as can be ascertained by reading the telemetry in the manner highlighted previously. The approximate times for completing the mission in each trial are given in Table 5.3. The mean time for the A-HAT-TRIC vehicle to complete the mission was 146.4 seconds.

Trial	**Completion Time**
1	137 s
2	106 s
3	196 s
4	123 s
5	170 s

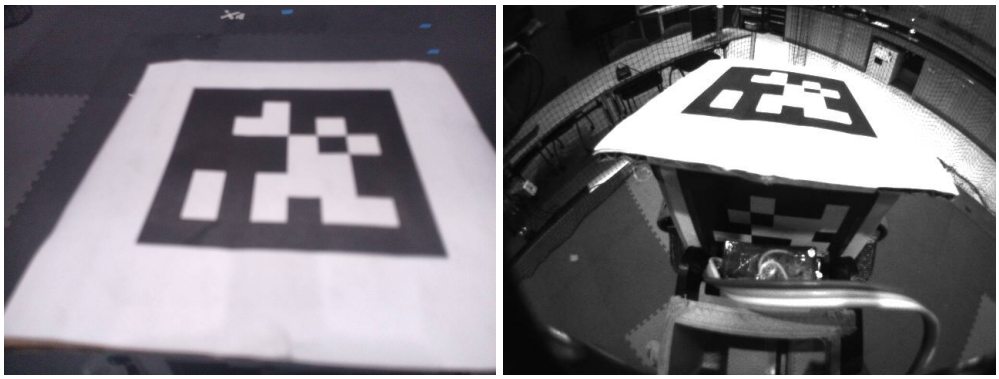
Table 5.3: Time to complete sample retrieval mission

It is observed that two trials, trial 3 and trial 5, have a significantly longer time to completion than the others. This can be attributed to the fact that the vehicle did not detect the landing AprilTag until after the vehicle had completed its first pass along the width of the search area. All other trials resulted in the initial landing AprilTag detection being made at the start of the aerial search. This can be observed in the telemetry in Appendix D.

A notable result from the testing of the autonomous algorithm was that of trial 5. In this particular test, the sample was retrieved such that an AprilTag was not immediately visible to either camera on the front of the vehicle. This unique configuration is shown in a still taken from the hires camera during the execution of this test in Figure 5.10. It was expected that this particular orientation of the sample with respect to the front of the vehicle would result in the algorithm inaccurately flagging the sample as not having been collected. However, it was still successful in cataloging the sample as having been successfully retrieved. For reference, the nominal positioning of the sample after having been retrieved with the cameras at the front of the flight deck is shown in Figure 5.11.

Figure 5.10: Irregularly retrieved sample in trial 5 of end-to-end demonstration

The success of the A-HAT-TRIC vehicle and its autonomous framework for sample detection and collection underscores its viability for further exploration as a concept for real-world applications. The vehicle's ability to detect and collect the sample in the defined mission framework demonstrates



(a) Hires camera field of view (b) Tracking camera field of view

Figure 5.11: Nominal orientation of sample with respect to detection cameras after retrieval

the possibility for research that can expand this concept into a more realistic mission concept.

Chapter 6: Conclusion

6.1 Summary of Contributions

This thesis investigated the concept of a hybrid aerial-terrestrial vehicle for the purpose of sample detection and collection. First, a custom design for a terrestrial platform was created for mounting to a commercial aerial platform. This design, while simple, provided the basic capabilities necessary to demonstrate terrestrial navigation and grasping of a sample at the surface. For the aerial and terrestrial platforms to be compatible with one another, a wireless communication protocol was established to enable seamless commanding from the flight computer to the microcontroller. To execute the sample collection mission, a framework that only required the operator to arm the vehicle was constructed. The autonomous algorithm utilized logical states to command vehicle performance. Finally, indoor experimental analysis was conducted to assess the feasibility of the vehicles similar to A-HAT-TRIC for real-world sample collection missions.

6.2 Future Work

To improve the practical utility of the A-HAT-TRIC vehicle, future research should be allocated to the following areas:

- **Enhancement of Terrestrial Locomotive Capabilities:** The maneuverability of the terrestrial

platform did not provide the level of precision necessary for proper orientation with respect to the sample for collection. Improving operability on the ground would greatly improve the capabilities of the vehicle as a whole. Aerial localization of fiducial markers can be negatively impacted by the vibration due to flight as well as changes in lighting conditions as the vehicle navigates. Terrestrial navigation, however, is not as susceptible to these issues and would prove a much more reliable method of orienting the vehicle for sample collection.

- **Increasing Gripper Utility:** The simple manipulator designed for this vehicle is not optimal for use with irregularly shaped objects. Creating a gripper capable of handling a large variety of objects would increase its usefulness. To increase potential applications of this vehicle concept, a universal gripper mount could be constructed such that the end effector can be easily changed for the collection of a sample of a different shape or size.

- **Utilization of Object Detection in Place of AprilTags:** For actual sample collection missions, it is highly unlikely that samples and landing sites would be adorned with fiducial markers. As such, the versatility of this algorithm can be increased by implementing an object detection framework. Custom models can train the algorithm to detect objects of interest, whether they be samples for collection or landmarks for landing sites.

- **Implementation in an Outdoor Environment:** With necessary hardware design changes, this vehicle should be operated outdoors to increase the fidelity of the testing performed. This would also necessitate use of the vehicle's global positioning sensor system and magnetometer, as visual inertial odometry may not always be as reliable outdoors as in an indoor environment. Increasing the capabilities of the vehicle for outdoor use would

result in a major design iteration that progresses the concept towards its intended use case for field sample collection.

Appendix A: Customization of Built-In AprilTag Detection Framework

In the native ModalAI infrastructure for AprilTag detection, `voxl-tag-detector`, the published AprilTag data does not denote which camera the localization estimate was made with respect to. Since A-HAT-TRIC employs two cameras for AprilTag detection, the tracking camera and hires image sensor, it is useful to attribute each estimate to the correct camera. Knowledge of the detecting camera allows for the appropriate transformation out of the camera frame to occur.

Accessing the AprilTag data innately collected by the flight computer requires interfacing with the Modal Pipe Architecture framework [45]. Publicly available code [46] from the developers at ModalAI was leveraged as a guide for exploiting this structure to obtain the necessary AprilTag data. The software written for this research created two ROS 2 publishers, one with AprilTag information from the hires sensor and another containing AprilTag data from the tracking camera. These messages specifically contained the AprilTag position and orientation with respect to the camera frame.

To properly utilize the localized AprilTags for the autonomous algorithm, the coordinates had to be transformed out of the camera frame. Ultimately, it was decided to publish the AprilTag position with respect to the body frame since it was a simple transform. The autonomous state machine included the inertialization of the AprilTag coordinates, as described in Chapter 2.

A second ROS 2 node was written for providing AprilTag localization in the body frame to

the autonomous algorithm. This node subscribes to the publishers containing AprilTag information with respect to the camera frame, and utilizes each cameras' extrinsics to transform the coordinates into their respective body frames. It publishes this data so that the autonomous algorithm can access it during operations.

A graphic showcasing this methodology is given in Figure A.1.

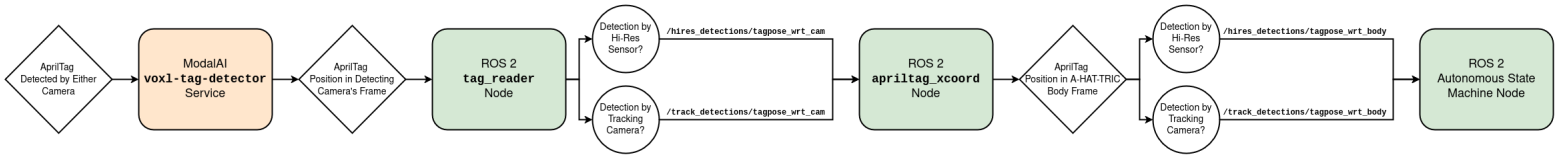


Figure A.1: Overview of modified AprilTag detection framework

Appendix B: Bill of Materials

These lists omit fasteners, wires, and other miscellaneous parts needed for assembly.

B.1 Aerial Platform

Item	Quantity	Manufacturer	Part Number
VOXL 2 Flight Deck	1	ModalAI	MDK-F0006-2-03
VOXL 2 I/O Expander Board	1	ModalAI	MDK-M0065-1-02
VOXL 2 USB3.0/UART Expansion Adapter	1	ModalAI	MDK-M0151-1-00
S500 Quadcopter Frame	1	Readytosky	20175171502
2212 920KV Brushless Motor	2 CW, 2 CCW	Readytosky	2017621103826
1045 Propellers	2 CW, 2 CCW	Readytosky	2017930142950
BL-Heli 20A Brushless Speed Controller	4	FLASH HOBBY	N/A
R9 SX Receiver	1	FrSky	N/A
M9N GPS	1	Holybro	12029

Table B.1: Aerial platform parts list

B.2 Terrestrial Platform

Item	Quantity	Manufacturer	Part Number
Arduino ESP32 Nano	1	Arduino	ABX00092
5V, 5A Step-Down Voltage Regulator	1	Pololu	2851
Continuous Rotation Servo	2	SpringRC	SM-S4303R
Wheel for Standard Servo Splines	2	Pololu	4925
Digital Micro Servo	1	FEETECH	FT90B
Ball Caster with 1” Plastic Ball and Plastic Rollers	1	Pololu	2691

Table B.2: Terrestrial platform parts list

Appendix C: VOXL 2 to Arduino Nano ESP32 Communication Protocol

The Arduino Nano ESP32 receives motor commands from the VOXL 2 via a wireless UDP socket connection. This necessitates proper setup in both ROS 2, on the VOXL 2, and in the script running on the Arduino Nano ESP32. For successful communication, both devices must be connected to the same Wi-Fi network and the Internet Protocol (IP) address of the Arduino Nano ESP32 must be known.

The Arduino Nano ESP32 leverages the `WiFi.h` [47] and `WiFiUDP.h` [48] libraries for communication. The former is used for connecting to the desired Wi-Fi network, and the latter is needed for receiving the UDP packets from the VOXL 2. The packet received is converted from a byte array to a float, which is then utilized to send a command to the correct motor. This script also prints the device's IP address and UDP port to the serial monitor, which assists in setup for the ROS 2 script on the VOXL 2.

The ROS 2 node exploits the `Boost.Asio` library [49] for establishing its connection to the Arduino Nano ESP32. This library allows for the endpoint of the communication network to be defined using the receiving device's IP address and UDP port. After the algorithm determines the proper motor commands to be sent to the terrestrial platform, it encodes this information as a single float value. The float is converted to a byte array so that it can properly be sent to the Arduino Nano ESP32. The mapping from float value transmitted to motor commands is given in

Table C.1. This table denotes the behavior of the locomotive motors and gripper motor separately, as they are not all active at the same time.

Float Value	Locomotive Motor Behavior	Gripper Motor Behavior
0	no motion	no change
1	drive forward	no change
2	drive backward	no change
3	turn right	no change
4	turn left	no change
5	no motion	close gripper
6	no motion	open gripper
7	zero-point clockwise rotation	no change
8	zero-point counter-clockwise rotation	no change

Table C.1: Mapping transmitted float value to motor behavior

Appendix D: End-to-End Laboratory Demonstration Plots

Trial 1

Trial 1 resulted in a successful collection of the sample and returned to the home location to deposit in 137 seconds.

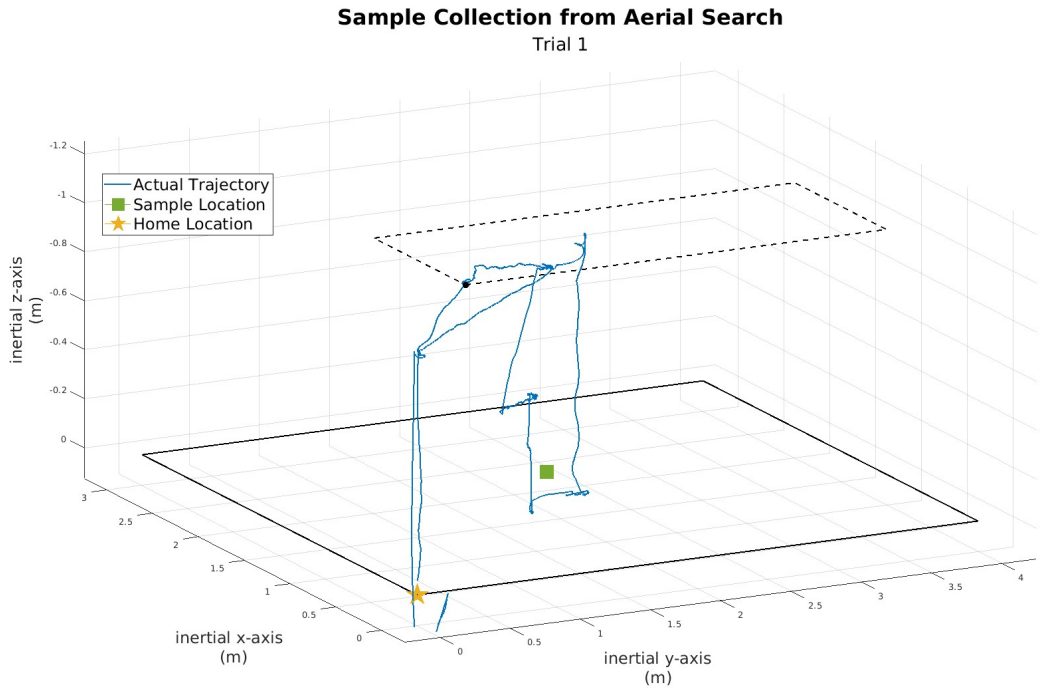


Figure D.1: End-to-end test inertial trajectory - trial 1

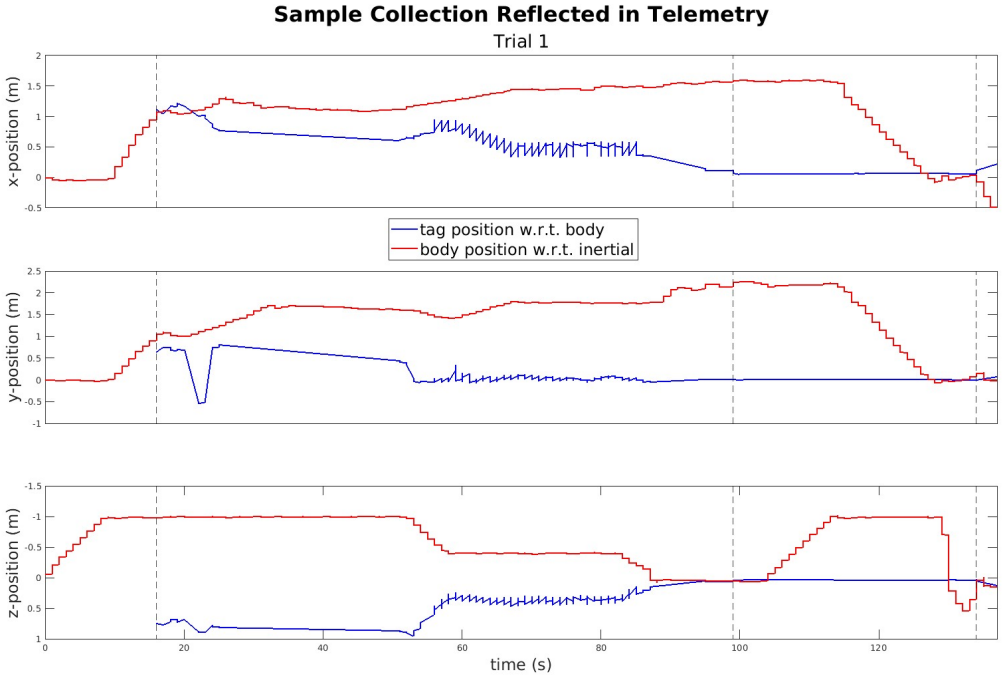


Figure D.2: Position of sample and position of vehicle over time - trial 1

Trial 2

Trial 2 resulted in a successful collection of the sample and returned to the home location to deposit in 106 seconds.

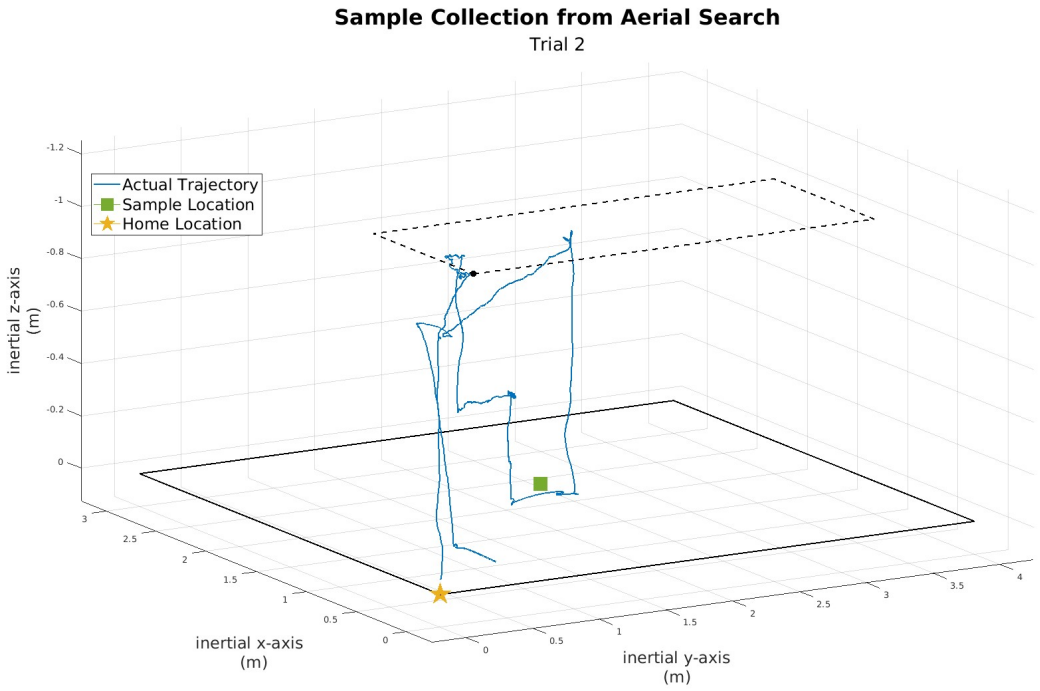


Figure D.3: End-to-end test inertial trajectory - trial 2

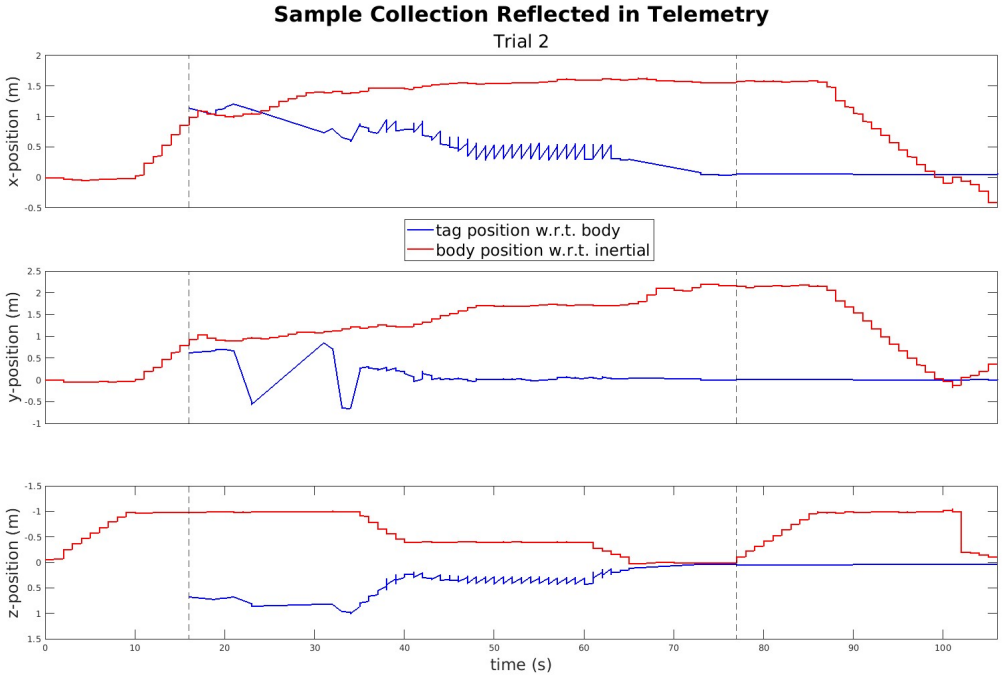


Figure D.4: Position of sample and position of vehicle over time - trial 2

Trial 3

Trial 3 resulted in a successful collection of the sample and returned to the home location to deposit in 113 seconds.

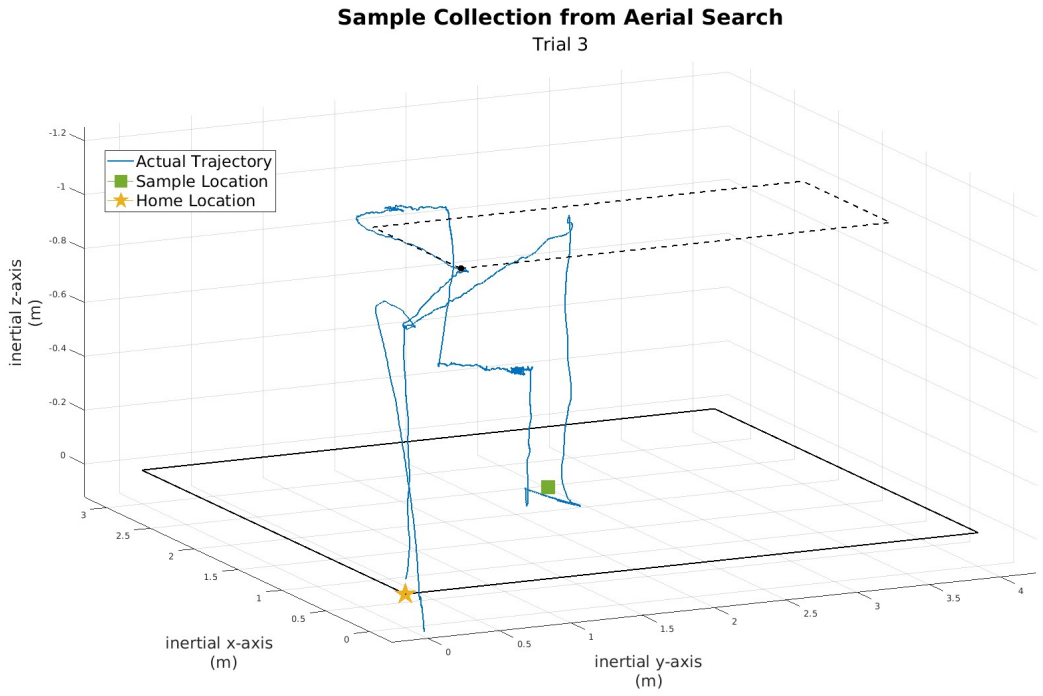


Figure D.5: End-to-end test inertial trajectory - trial 3

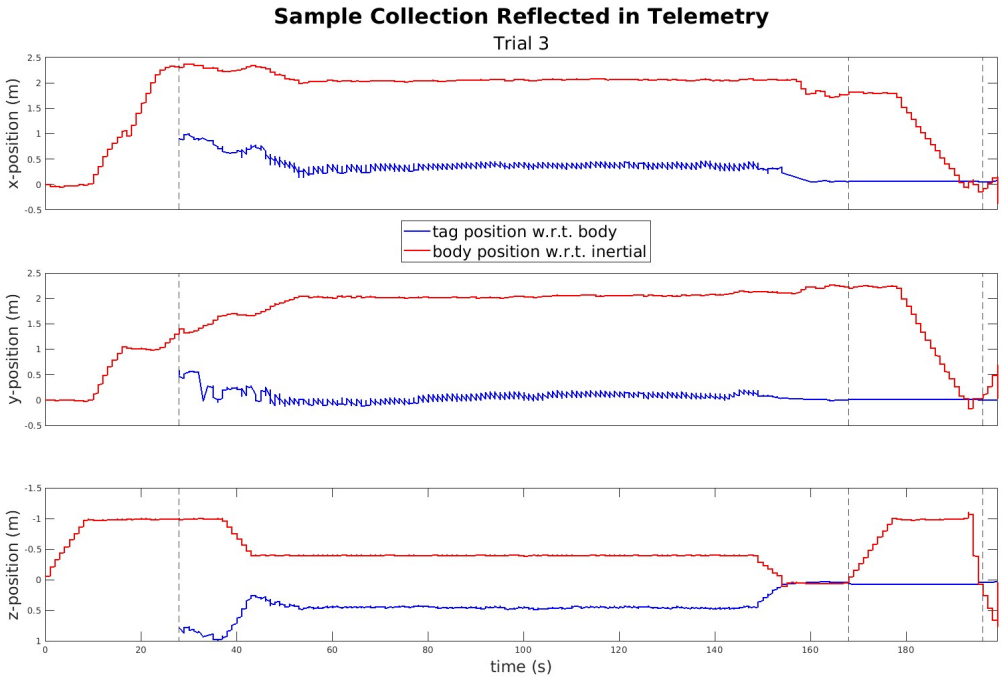


Figure D.6: Position of sample and position of vehicle over time - trial 3

Trial 4

Trial 4 resulted in a successful collection of the sample and returned to the home location to deposit in 170 seconds.

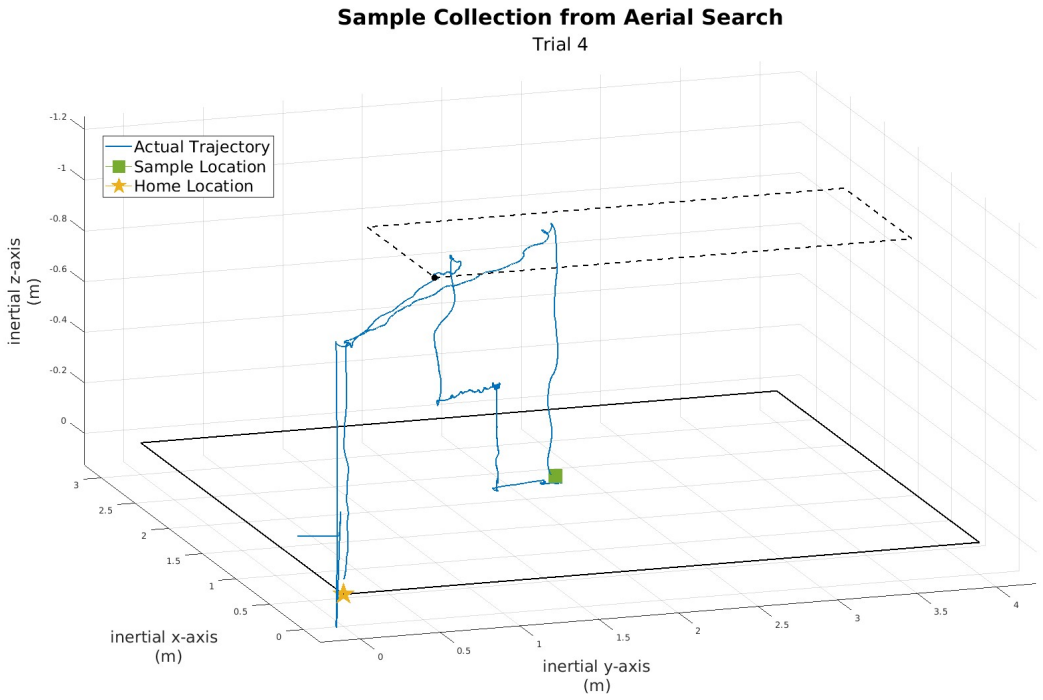


Figure D.7: End-to-end test inertial trajectory - trial 4

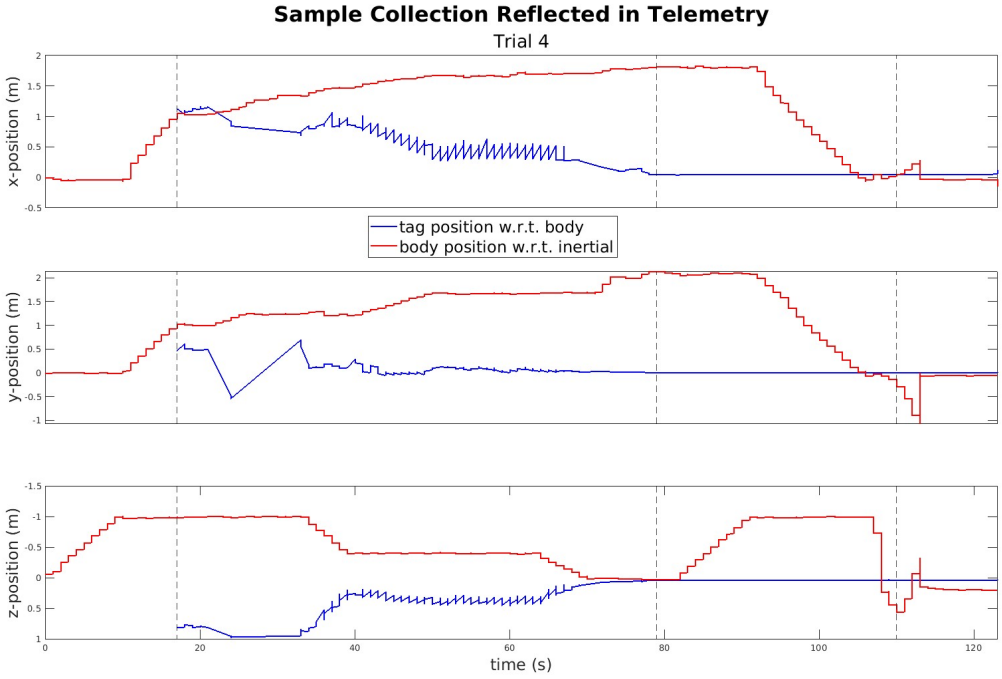


Figure D.8: Position of sample and position of vehicle over time - trial 4

Trial 5

Trial 4 resulted in a successful collection of the sample and returned to the home location to deposit in 148 seconds. In this trial, the sample was collected in a sub-optimal configuration and an AprilTag was not able to be detected by either camera. However, the algorithm was still able to determine that a sample had been collected and successfully deposited.

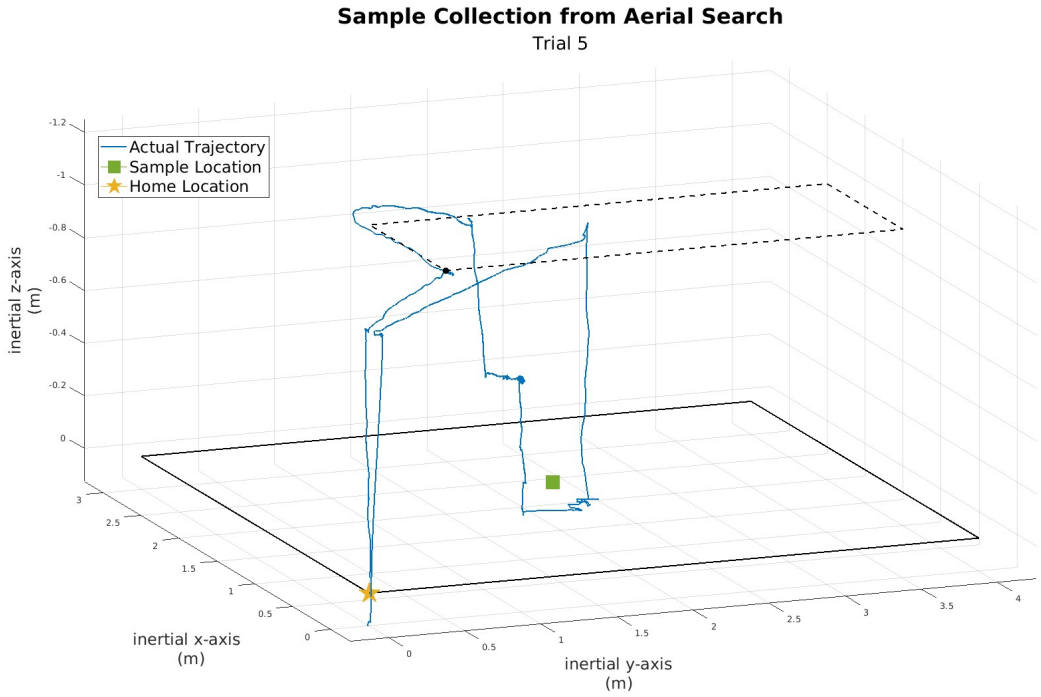


Figure D.9: End-to-end test inertial trajectory - trial 5

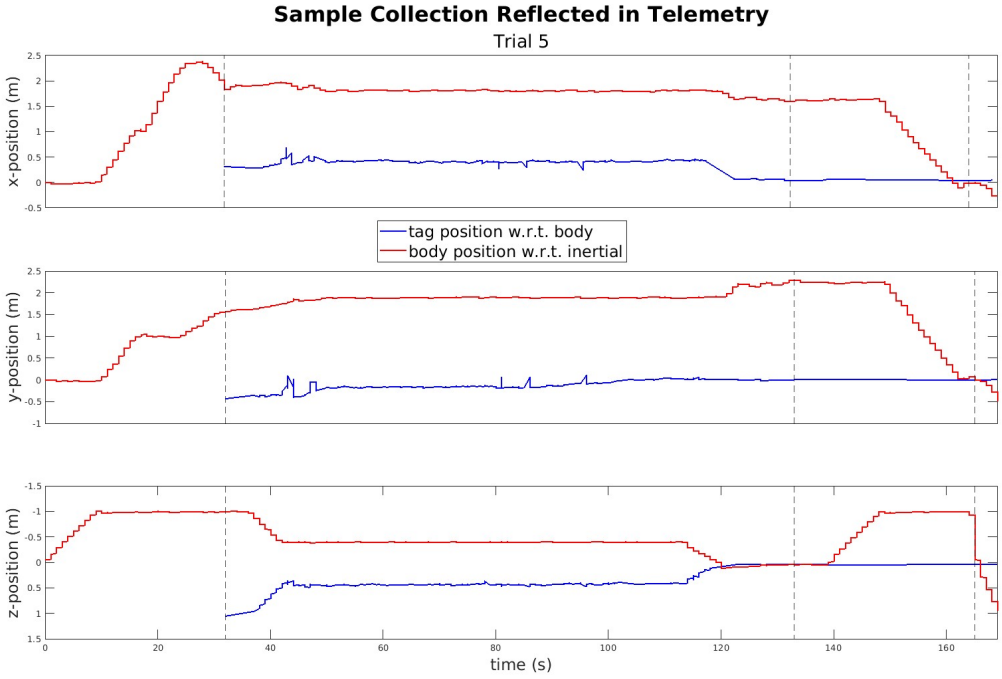


Figure D.10: Position of sample and position of vehicle over time - trial 5

Bibliography

[1] Mars Sample Return - Mars Missions - NASA Jet Propulsion Laboratory — jpl.nasa.gov. https://www.jpl.nasa.gov/missions/mars-sample-return-msr/. [Accessed 18-02-2025].

[2] S. Withrow-Maser, W. Johnson, N. Schatzman, L. Young, H. Cummings, C. Malpica, L. Meyn, B. Allan, T. Tzanetos, H. Grip, W. Koning, A. Chan, A. Ruan, B. Pipenberg, and M. Keennon. Mars Sample Recovery Helicopter: Rotorcraft to Retrieve the First Samples from the Martian Surface. *Proceedings of the Vertical Flight Society 79th Annual Forum*, May 2023.

[3] NASA-JPL Sample Recovery Helicopter Concept Art. https://mars.nasa.gov/system/downloadable_items/47754_PIA25338.jpg. [Accessed 27-02-2025].

[4] G. Muscato, F. Bonaccorso, L. Cantelli, D. Longo, and C.D. Melita. Volcanic Environments: Robots for Exploration and Measurement. *IEEE Robotics Automation Magazine*, 19(1):40–49, 2012.

[5] J. E. Bares and D. S. Wettergreen. Dante II: Technical Description, Results, and Lessons Learned. *The International Journal of Robotics Research*, 18(7):621–649, 1999.

[6] C. R. Stoker, N. A. Cabrol, T. R. Roush, J. Moersch, J. Aubele, N. Barlow, E. A. Bettis III, J. Bishop, M. Chapman, S. Clifford, C. Cockell, L. Crumpler, R. Craddock, R. De Hon, T. Foster, V. Gulick, E. Grin, K. Horton, G. Hovde, J. R. Johnson, P. C. Lee, M. T. Lemmon, J. Marshall, H. E. Newsom, G. G. Ori, M. Reagan, J. W. Rice, S. W. Ruff, J. Schreiner, M. Sims, P. H. Smith, K. Tanaka, H. J. Thomas, G. Thomas, and R. A. Yingst. The 1999 Marsokhod rover mission simulation at Silver Lake, California: Mission overview, data sets, and summary of results. *Journal of Geophysical Research*, 106(E4):7639–7663, 2001.

[7] M. R. James, B. B. Carr, F. D'Arcy, A. K. Diefnbach, H. R. Dietterich, A. Fornaciai, E. Lev, E. J. Liu, D. C. Pieri, M. Rodgers, B. Smets, A. Terada, F. W. von Aulock, T. R. Walter, K. T. Wood, and E. U. Zorn. Volcanological applications of unoccupied aircraft systems (UAS): Developments, strategies, and future challenges. *Volcanica*, 3(1):67–114, 2020.

[8] E. Eiden, M. E. Pritchard, and P. R. Lundgren. Spatial and Temporal Resolution Needs for Volcano Topographic Change Data Sets Based on Past Eruptions (1980–2019). *Earth and Space Science*, 10(10):e2023EA003054, 2023.

[9] H. V. Abeywickrama, B. A. Jayawickrama, Y. He, and E. Dutkiewicz. Empirical Power Consumption Model for UAVs. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–5, 2018.

[10] A. Kalantari and M. Spenko. Modeling and Performance Assessment of the HyTAQ, a Hybrid Terrestrial/Aerial Quadrotor. *IEEE Transactions on Robotics*, 30(5):1278–1285, October 2014.

[11] D. Zhang, M. Xu, P. Zhu, C. Guo, Z. Zhong, H. Lu, and Z. Zheng. The development of a novel terrestrial/aerial robot: autonomous quadrotor tilting hybrid robot. *Robotica*, 42(1):118–138, 2024.

[12] K. Shi, Z. Jiang, L. Ma, L. Qi, and M. Jin. MTABot: An Efficient Morphable Terrestrial-Aerial Robot With Two Transformable Wheels. *IEEE Robotics and Automation Letters*, 9(2):1875–1882, 2024.

[13] S. Yu, B. Pu, K. Dong, S. Bai, and P. Chirarattananon. A Hybrid Quadrotor With a Passively Reconfigurable Wheeled Leg Capable of Robust Terrestrial Maneuvers. *IEEE Robotics and Automation Letters*, 10(4):3486–3493, 2025.

[14] P. Wang, Z. Man, Z. Cao, J. Zheng, and Y. Zhao. Dynamics modelling and linear control of quadcopter. In *2016 International Conference on Advanced Mechatronic Systems (ICAMechS)*, pages 498–503, 2016.

[15] PX4 Autopilot User Guide. `https://docs.px4.io/main/en/`. [Accessed 08-03-2025].

[16] Controller Diagrams. `https://docs.px4.io/main/en/flight_stack/controller_diagrams.html`. [Accessed 08-03-2025].

[17] Relocalization — ModalAI Technical Docs. `https://docs.modalai.com/voxl-vision-hub-apriltag-relocalization/#frames-of-reference`. [Accessed 09-03-2025].

[18] Apriltag Detection — ModalAI Technical Docs. `https://docs.modalai.com/voxl-tag-detector/#tag-coordinate-frame`. [Accessed 09-03-2025].

[19] AprilTag. `https://april.eecs.umich.edu/software/apriltag`. [Accessed 09-03-2025].

[20] ROS2 Installation VOXL 2 — ModalAI Technical Docs. `https://docs.modalai.com/ros2-installation-voxl2/#mpa-to-ros2`. [Accessed 09-03-2025].

[21] VehicleLocalPosition (UORB message) — PX4 Guide (main). `https://docs.px4.io/main/en/msg_docs/VehicleLocalPosition.html`. [Accessed 09-03-2025].

[22] Configure Extrinsics — ModalAI Technical Docs. https://docs.modalai.com/configure-extrinsics/. [Accessed 09-03-2025].

[23] VehicleAttitude (UORB message) — PX4 Guide (main). https://docs.px4.io/main/en/msg_docs/VehicleAttitude.html. [Accessed 09-03-2025].

[24] Distributions - ROS Wiki. http://wiki.ros.org/Distributions. [Accessed 11-03-2025].

[25] ROS 2 Documentation - ROS 2 Documentation: Rolling Documentation. https://docs.ros.org/en/rolling/index.html#. [Accessed 12-03-2025].

[26] ROS 2 middleware interface. http://design.ros2.org/articles/ros_middleware_interface.html. [Accessed 12-03-2025].

[27] Getting Started with Arduino IDE 2 — Arduino Documentation. https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/. [Accessed 13-03-2025].

[28] Use ROS 2 to interact with Gazebo - Gazebo ionic documentation. https://gazebosim.org/docs/latest/ros2_integration/. [Accessed 13-03-2025].

[29] ROS 2 User Guide — PX4 Guide (main). https://docs.px4.io/main/en/ros2/user_guide.html. [Accessed 13-03-2025].

[30] uXRCE-DDS (PX4-ROS 2/DDS Bridge) — PX4 Guide (main). https://docs.px4.io/main/en/middleware/uxrce_dds.html. [Accessed 13-03-2025].

[31] Gazebo Vehicles — PX4 Guide (main). https://docs.px4.io/main/en/sim_gazebo_gz/vehicles.html. [Accessed 13-03-2025].

[32] Gazebo Worlds — PX4 Guide (main). https://docs.px4.io/main/en/sim_gazebo_gz/worlds.html. [Accessed 13-03-2025].

[33] Sentinel Drone Functional Description. https://docs.modalai.com/sentinel-functional-description/. [Accessed 27-02-2025].

[34] RS2212 920KV CWCCW Brushless Motor. https://www.readytosky.com/e_productshow/?787-RS2212-920KV-CW&CCW-Brushless-Motor-787.html. [Accessed 27-02-2025].

[35] 3D printing with PLA vs. ABS: What's the difference? https://www.hubs.com/knowledge-base/pla-vs-abs-whats-difference/. [Accessed 27-02-2025].

[36] Pololu Wheel for Standard Servo Splines (25T, 5.8mm) - 70x8mm, Black, 2-Pack. https://www.pololu.com/product/4925/specs. [Accessed 27-02-2025].

[37] Z. X. Zhang, T. Zhang, D. Wang, X. Zhang, X. Zhenxiang, and K. Prakashan. Physicomechanical, friction, and abrasion properties of EVA/PU blend foams foamed by supercritical nitrogen. *Polymer Engineering & Science*, 58(5):673–682, 2018.

[38] SpringRC SM-S4303R Continuous Rotation Servo. https://www.pololu.com/product/1248. [Accessed 27-02-2025].

[39] FEETECH Continuous Rotation Servo FS5106R. https://www.pololu.com/product/3430. [Accessed 27-02-2025].

[40] 3V Digital Micro Servo, Standard (FT90B). https://www.parallax.com/product/3v-digital-micro-servo-standard-ft90b/. [Accessed 27-02-2025].

[41] trajectory_setpoint (UORB message) — PX4 User Guide (v1.13). https://docs.px4.io/v1.13/en/msg_docs/trajectory_setpoint.html. [Accessed 17-03-2025].

[42] VehicleCommand (UORB) — PX4 Guide (main). https://docs.px4.io/main/en/msg_docs/VehicleCommand.html. [Accessed 24-03-2025].

[43] VehicleStatus (UORB) — PX4 Guide (main). https://docs.px4.io/main/en/msg_docs/VehicleStatus.html. [Accessed 24-03-2025].

[44] QGroundControl - Drone Control - Ground Control Station for Small Air - Land - Water Autonomous Unmanned Systems. https://qgroundcontrol.com/. [Accessed 01-04-2025].

[45] Modal Pipe Architecture — ModalAI Technical Docs. https://docs.modalai.com/mpa/. [Accessed 03-04-2025].

[46] clients/voxl-inspect-tags.c · master · voxl / VOXL SDK / Services / voxl-tag-detector · GitLab. https://gitlab.com/voxl-public/voxl-sdk/services/voxl-tag-detector/-/blob/master/clients/voxl-inspect-tags.c. [Accessed 03-04-2025].

[47] WiFi — Arduino Documentation. https://docs.arduino.cc/libraries/wifi/. [Accessed 16-03-2025].

[48] WiFiUDP — Arduino Documentation. https://docs.arduino.cc/language-reference/en/functions/wifi/udp/. [Accessed 16-03-2025].

[49] Boost.Asio - master. https://www.boost.org/doc/libs/master/doc/html/boost_asio.html. [Accessed 16-03-2025].