Autonomous UAV Navigation in a GPS-Denied Outdoor Environment Using Discontinuous Visual Contact With Fiducial Markers

Auburn REU on SMART UAVs 2021

Technical Report #CSSE21-07

Dunn, Nicholas
Columbus State University
dunn_nicholas2@columbusstate.edu

Schallenberger, Zachary
Montana State University
z.schallenberger@student.montana.edu

Biaz, Saad
Auburn University
biazsaa@auburn.edu

Chapman, Richard
Auburn University
chapmro@auburn.edu

December 2, 2021

Abstract

Autonomous navigation of Unmanned Aerial Vehicles (UAVs) has become an increasingly popular area of study as technology has advanced over the years. One technology that has been developed for use in the autonomous navigation of UAVs is fiducial markers. Fiducial markers are a type of physical tag that can be placed in an environment to assist with UAV localization, navigation, and landing. A UAV can identify and scan a fiducial marker with an onboard camera and take appropriate action based on the data received from the marker. Previous research using only fiducial markers and onboard sensors for UAV navigation in a GPS-denied environment requires the camera to have constant visual contact with at least one marker. This paper explores the use of fiducial markers for outdoor UAV navigation in a GPS-denied environment without the restriction of constant visual contact with a marker. Physical implementation and formal methods are used to study the performance and overall viability of this solution.

1 Introduction

1.1 Context and Terminology

An **Unmanned Aerial Vehicle** (UAV) is an aircraft vehicle that does not have any pilots or passengers on board. It is often informally referred to as a drone. A UAV can be controlled in many different ways but this paper exclusively explores the autonomous control of these vehicles. UAVs typically fly autonomously using GPS-based navigation. In the case of a GPS-denied environment, UAVs cannot properly navigate autonomously without other methods of navigation. A **fiducial marker** is a known shape or pattern that is typically printed onto a flat surface. Similar to a QR code, a fiducial marker can provide data via camera recognition. The data obtained from a fiducial marker can take the form of pixel coordinates to track the four corners and center of a fiducial marker and provide a unique identification number associated with each fiducial marker. **AprilTag** is a type of fiducial marker system widely used with UAVs and UGVs (unmanned ground vehicles). AprilTags have demonstrated consistent performance (Kalaitzakis *et al.* [2])

in the past and are the fiducial marker family that was chosen for this research paper. AprilTags can be created using a normal printer and the detection software for AprilTags can compute the 3D positioning, orientation, and identification of the markers from an adequate distance relative to the size of the AprilTag. Fiducial markers can be used to provide navigational data in the event of a UAV entering a GPS-denied environment by functioning as a method of decoding predefined instructions assigned to the corresponding AprilTag identification number. A **GPS-denied environment** is an environment in which GPS technologies are unable to function due to exterior circumstances. **Autonomous navigation** is the execution of some plan or path without any human input or control along the way. Autonomous navigation can be paired with computer vision software to assist the program in its decision-making process. One computer vision library is **OpenCV** (Open Source Computer Vision Library): an open-source software library that provides access to tools essential for computer vision and machine learning. OpenCV is used in conjunction with the AprilTag software to assist a UAV in properly visualizing and decoding an AprilTag. **Pose** refers to the position and orientation of an object. Specific to this research, the pose describes the translation of a UAV (using values of all three axes) relative to a marker or particular coordinate system. It also describes the orientation of a UAV in terms of pitch, roll, and yaw. A "**dead zone**" is a term used in this paper to describe the period in which a UAV is autonomously flying without the detection of a fiducial marker or using any other form of a visual or GPS-based navigation system.

1.2 Problem

GPS-based navigation is an incredibly common form of navigation that many devices rely on. UAVs typically rely on GPS to fulfill autonomously driven missions. However, GPS-based technology faces many different risks as a chosen form of navigation. GPS is susceptible to spoofing and environmental interference. Furthermore, GPS does not function well in indoor environments and cannot be relied on for precision navigating. Due to the potential vulnerabilities of GPS-based navigation, there is a need for research into other forms of navigation for autonomously flown UAVs.

1.3 Solution

This research paper explores the idea of a new form of visual-based navigation that centers around fiducial markers. The research analyzes how a UAV could autonomously navigate a predefined course using fiducial markers (specifically AprilTags) as a form of instruction in a GPS-denied environment while relying on autonomous flying algorithms in between discontinuous visual contact "dead zones" of the fiducial markers. This solution will mitigate any concern of a failure of onboard GPS-based navigation of a UAV by implementing a discontinuous visual-based navigation system using fiducial markers.

1.4 Evaluation of Methods

This paper will be evaluating the research through a few key metrics:

1. The research will measure how well a UAV can read an AprilTag from a relative altitude through a program using OpenCV to process a live feed of a camera attached to the UAV and return data showing the detection areas of a fiducial marker. The research will measure the success by conducting tests that involve flying the drone at arbitrary locations with an AprilTag within view of the camera and measuring the success rate at which it can detect a marker in its line of sight.

2. The research will measure how well a UAV can decode a fiducial marker as a set of instructions. This will be done by setting up a test involving multiple fiducial markers placed at a variety of distances away from a camera. The measurement of success will be determined by the UAV's ability to successfully complete a set of instructions received from decoding a fiducial marker's unique identification number.

3. The research will measure how well a UAV can navigate between fiducial markers using autonomous navigation by evaluating the performance of a UAV on courses where the UAV will be required to

navigate through substantial "dead space" between AprilTags relying solely on autonomous navigation in a GPS-denied environment. The research will analyze different distances that the fiducial markers can be placed at to still guarantee a successful flight of the UAV following the course.

1.5 Organization

This paper will detail different methods and approaches taken to produce measurable results to draw a conclusion. The order in which information will be presented will be as follows: **Related Work**, **Methodology**, **Results**, **Conclusion**, and **References Cited**.

2 Related Work

Previous research has studied the use of fiducial markers for assisting UAVs in localization, navigation, and landing in both GPS-enabled and GPS-denied environments. Some of the methods used have incorporated other technologies as well, such as accelerometers, magnetometers, and gyroscopes. Both indoor and outdoor environments have been used for different applications. Different types of fiducial markers have also been developed and studies have been conducted comparing their performance.

Kalaitzakis *et al.* [2] performed a survey of different fiducial markers and their applications and conducted studies comparing four of the most widely used fiducial marker packages. The markers that were chosen for the study were ARTag, AprilTag, ArUco, and STag. A comparison of the accuracy of detection, detection rate, and computational cost of each marker was conducted. Experiments with the markers used different lighting conditions, camera angles, and distances between the camera and the markers. Three different computers (Raspberry Pi 3B+, Nvidia Jetson TX2, Intel NUC) and two different cameras (C270 Webcam, Raspberry Pi Camera Module v2) were also used to assess the performance of the markers. Each marker had strengths and weaknesses, but the AprilTag and STag markers had the overall best performance in the study.

As mentioned earlier, both indoor and GPS-denied environments have been used during autonomous UAV localization and navigation research. Along with the possibility of weak GPS signals, indoor environments provide unique navigational challenges for UAVs due to the interior features of buildings. Nahangi *et al.* [4] used fiducial markers for automated UAV localization of a GPS-denied indoor construction environment. Their experiment used a Parrot Bebop 2 drone with an onboard camera to detect AprilTag fiducial markers. The drone also used an accelerometer to assist with localization and navigation. The AprilTags were linked to previously known coordinates in the building information model (BIM), and the UAVs were localized using the information from the AprilTags and their position relative to the onboard camera. Previously computed information could then be used to successfully navigate the environment; however, the UAV was required to detect at least one AprilTag at all times.

Another study conducted by Bacik *et al.* [1] used ArUco fiducial markers to assist with UAV navigation. The researchers used a Parrot AR Drone with two cameras to detect the markers. Additional instrumentation used included an ultrasound, pressure altimeter, gyroscope, accelerometer, and magnetometer. Their method of navigation required the *n-1*th marker to be detected by the camera at the same time the *n*th marker was detected. They then calculated the pose of the *n*th marker with respect to the *n-1*th marker and used that information to keep track of the path of the UAV. A fuzzy control algorithm used the calculated pose and trajectory of the UAV to navigate it from one marker to the next. The UAV achieved a fully autonomous flight in an empty warehouse using only marker-based navigation combined with a fuzzy control algorithm. However, this is another method that requires at least one marker to be detected at all times, with the additional requirement that two markers be detected simultaneously at certain times.

Lamberti *et al.* [3] combined ARTag fiducial markers with image feature matching to navigate UAVs. A Parrot AR Drone, equipped with an ultrasound altimeter and two cameras, was used to conduct experiments. The overall method of navigation used image matching to estimate the pose of the UAV between markers and then used markers to determine the true position of the UAV and to correct any drift that occurred using

image-based navigation between the markers. The marker-less pose estimation of the UAV was performed by frame-to-frame comparisons of the features in the images captured. After an initial image was captured, each additional image underwent a feature analysis comparison against the initial image and pose estimation of the UAV. A distance threshold between the UAV and the initial image was set and after the threshold was passed, a new reference image was obtained, in which each additional image was compared for pose estimation. A control algorithm was implemented to maintain the pose of the UAV with respect to the reference image. Whenever a fiducial marker was detected, a marker-based pose estimation algorithm was executed. The marker-based algorithm, along with the UAV control algorithm, corrected the drift errors that accumulated during the marker-less navigation. The researchers concluded that this approach could maintain an acceptable level of drifting errors while navigating UAVs.

Other studies have focused on using fiducial markers solely for pose estimation. Seng *et al.* [6] used fiducial markers to estimate the pose of a Parrot AR Drone in a GPS-denied environment. Their experiments were conducted in an indoor room that had fiducial markers placed on a wall. The ARToolKit software library was used to help compute the pose of the UAV in relation to the detected markers, which could then be used to estimate the pose of the UAV in the world coordinate frame based on the known pose of each marker. At least one marker had to be detectable to estimate the pose of the UAV, however, the accuracy of pose estimation increased as the number of markers detected in a single image increased. The increased accuracy resulted from computing the average pose of the UAV based on the poses of all detected markers.

Although different technologies and environments have been used for autonomous UAV navigation, the methods that only use fiducial markers and onboard sensors require continuous detection of at least one marker by an onboard camera. The purpose of this paper is to explore autonomous UAV navigation using only fiducial markers and onboard sensors, without the requirement of continuous detection of a marker in an outdoor, GPS-denied environment.

3 Methodology

To develop a system for autonomous navigation of UAVs using only fiducial markers and onboard sensors, without the requirement of continuous detection of a marker, a suitable marker system, AprilTag, was selected. After selecting a fiducial marker system, a Raspberry Pi 3B with a Raspberry Pi Camera Module v2 was obtained and a Python program was developed to detect the AprilTags. Next, two markers were printed, and experiments were conducted to detect them with the Raspberry Pi camera. The next step was to select a UAV to use for the research: the Parrot AR.Drone 2.0. After selecting the drone, several Python programs were developed to control the drone's movements, onboard sensors, and cameras. Several flights were then successfully executed using the programs. Next, the Raspberry Pi was mounted onto the UAV and the camera was positioned towards the front of the UAV at a downward-facing angle. Python programs were later developed and tested to align the UAV with a detected fiducial marker. A program was developed to navigate the UAV from one marker to the next using only onboard sensors to minimize drift errors. Finally, the previously computed position of each marker with respect to the previous marker was stored as data with the previous marker, so that each time the UAV arrived at a marker, it read in the positional data of the next marker and adjusted its heading to navigate in a straight-line path to the next marker.

3.1 Fiducial Markers

To select a fiducial marker family for the research, some of the most widely used packages in autonomous UAV navigation were investigated and it was decided to use AprilTags. As mentioned earlier, Kalaitzakis *et al.* [2] performed a comparison of the ARTag, AprilTag, ArUco, and STag markers and found that AprilTag and STag had the overall best performance in their study. Additionally, AprilTags were also found to be used in several other papers involving UAV navigation, such as the research done by Nahangi *et al.* [4]

AprilTag fiducial markers were developed and introduced by Edwin Olson [5] to improve upon other markers that existed at that time. In the research done by Olson, they reported that AprilTags provide "a

fast and robust line detection system, a stronger digital coding system, and greater robustness to occlusion, warping, and lens distortion." [5] Since then, AprilTag has undergone several generations of development, with AprilTag 3 being the most current version at the time of writing this paper. The AprilTag fiducial marker system offers several different families of tags to choose from based on the application. The AprilTag software is also open-source and readily accessible. In addition to UAVs, AprilTags can also be used with other robotics platforms.

3.2 Unmanned Aerial Vehicle

The UAV selected for this research was the Parrot AR.Drone 2.0. The Parrot AR.Drone 2.0 is a quadcopter with a protective frame. It comes equipped with both forward and downward-facing cameras as well as several sensors, including an ultrasound altimeter, three-axis accelerometer, magnetometer, and a three-axis gyroscope. There is also a rechargeable battery capable of twelve minutes of flight time. The UAV generates its own Wi-Fi network allowing for a multitude of devices to connect and control the drone. The Parrot AR.Drone 2.0 provides a software development kit (SDK) for developers and has been used in many different studies involving UAVs, including research done by Bacik *et al.* [1], Lamberti *et al.* [3], and Seng *et al.* [6]. The application programming interface (API) enables developers to access and control the cameras, sensors, and movement of the drone. Other APIs have also been developed for the Parrot AR.Drone 2.0, such as PS-Drone. For this research, the PS-Drone API was chosen due to the functionality it provides and its ease of use. PS-Drone provides a simple library of commands that can be used in a Python program to control the movements of the drone and acquire valuable data from its onboard sensors.

3.3 Navigation

Autonomous navigation of the UAV was achieved using a combination of its onboard sensors and AprilTag fiducial markers. The position of each AprilTag with respect to the previous was computed and stored as positional data. At initial startup, the drone increased its altitude to a height greater than one meter to expand the camera's field of view and enable a wider range of AprilTag detection. After adjusting its altitude, the UAV proceeded to fly forward while using the Raspberry Pi camera to scan for an AprilTag in the environment. When a new marker was detected, the data associated with the marker was used to acquire the positional data of the next marker. The UAV then used that data to adjust its heading to navigate in a straight-line path towards the next marker. Once navigation began, the drone used only its onboard sensors to minimize drift errors until the next marker was detected by the camera. When the next marker was detected, the data associated with the marker was once again used to acquire the positional data of the next marker to enable the drone to adjust its heading to navigate in a straight-line path towards the next marker. This process was repeated until the drone fully navigated the desired path. The drone's altimeter and magnetometer were also used to record its positional data during flight.

3.4 Experimental Setup

All the programs implemented in the system were written in the Python programming language and executed on a Raspberry Pi 3B computer using the PS-Drone API to communicate with the Parrot AR.Drone 2.0 on its Wi-Fi network. Secure Shell (SSH) was used to enable control of the Raspberry Pi from a remote computer connected to the drone's Wi-Fi network. The Wi-Fi network provided by the drone functioned as a bridge between the Raspberry Pi and the remote computer. The remote computer was used to execute programs on the Raspberry Pi and initiate a safe landing sequence for the drone when necessary. Both indoor and outdoor environments were used to conduct experiments. Indoor experiments were performed to develop programs to detect AprilTags, access the UAV's sensors and cameras, and control the flight of the UAV. Outdoor experiments were performed later in the study to assist in algorithm development and testing the system in a different environment with the addition of new, external variables. Different marker placement configurations were also explored.

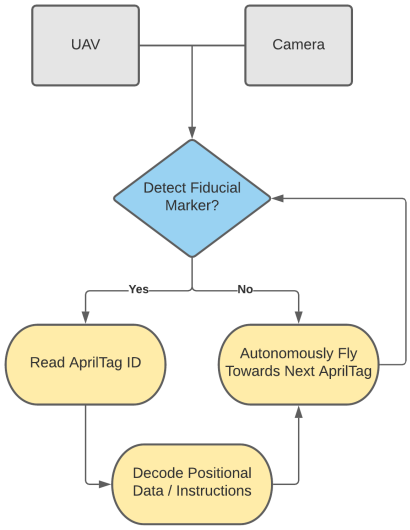

Figure 1: Flowchart outlining autonomous flight logic implemented with AprilTags

3.5 Indoor Experiments

An indoor environment was initially chosen for the algorithm development and testing of the software and hardware components of the system. To begin, a Raspberry Pi 3B with a Raspberry Pi Camera Module v2 was first obtained and configured. After configuration, the software for the AprilTag fiducial markers was installed. Programs were then developed to detect the AprilTags in an image captured by the Raspberry Pi Camera Module v2. Next, two AprilTags were printed, and experiments were conducted to detect the markers at various positions and distances. More advanced tag detection programs were then developed using OpenCV, including a program that visualizes the position of the center of the tag with respect to the center of the camera. The visual detection program overlays an 8x8 grid on the camera view and highlights grid-squares that currently detect an AprilTag pixel-corner. The program also displays a straight line from the center of the camera view to the center of the AprilTag. Next, PS-Drone was installed and programs were implemented to access and control the sensors, cameras, and movement of the Parrot AR.Drone 2.0 by importing PS-Drone and using the functions it provided. Tests were conducted to acquire real-time sensor data for the altitude and orientation of the drone as well as accessing its front and ground cameras while manually manipulating the drone. The drone's battery performance and Wi-Fi link quality were also evaluated through tests developed in Python. An external Adafruit LSM303DLHC triple-axis accelerometer/magnetometer was also incorporated into the study as a backup for the drone's onboard sensors. The LSM303DLHC was wired to the GPIO pins on the Raspberry Pi and, just as with the drone's onboard sensors, tests were performed to acquire real-time sensor data while manually manipulating the sensor. Many in-flight experiments were also conducted. Programs were then implemented on the Raspberry Pi using the PS-Drone API to demonstrate motion control of the drone. Commands were executed to control the drone's altitude, position, orientation, and speed. Override controls were also implemented to enable remote initiation of a safe landing sequence when necessary.

After demonstrating successful control of the drone's movements and sensors, the main flight program for the system was developed. Next, the Raspberry Pi, Raspberry Pi Camera Module v2, and external accelerometer/magnetometer were attached to the drone. The Raspberry Pi was attached to the top of the drone at the center of its protective hull for even weight distribution. The external accelerometer/magnetometer was attached in front of the Raspberry Pi, and the Raspberry Pi camera was attached to the bottom, towards the front of the drone at a downward-facing angle. A flight course for the UAV was then created by taping

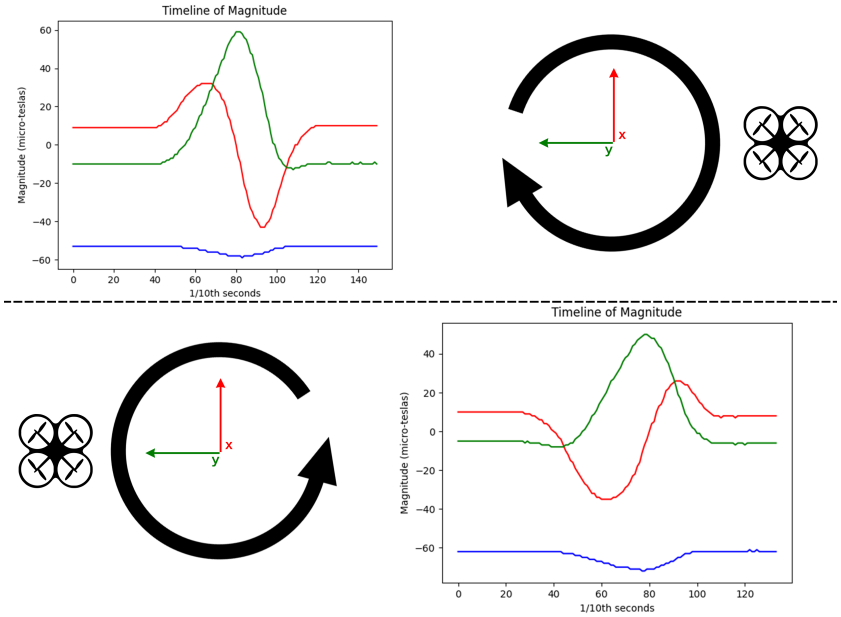

Figure 2: Recorded data of drone magnitude values relative to turning direction

two AprilTags to the floor at different locations in the lab. A compass was used to determine the angle of the second marker relative to the first. The angle was stored in the program with data for the first marker as positional data for the second marker. Finally, the flight control program was executed to test the system. Many tests were conducted to assist in refining the flight control program. Programs were also developed to read in and plot the yaw angle and magnetometer values of the drone during both real and simulated flights. Simulated flights were accomplished by manually manipulating the drone through the course.

3.6 Outdoor Experiments

An outdoor environment was chosen to conduct the final tests of the system. Two boxes were obtained and an AprilTag was taped to the top of each box. The boxes were positioned in a large grassy area to create an outdoor flight course for the UAV similar to the indoor course. The flight program was then executed to test the system. Several flights were conducted.

4 Results

4.1 Marker Detection and Decoding

Marker detection and decoding capabilities were initially evaluated before mounting the Raspberry Pi and Raspberry Pi camera to the UAV. The positions of two AprilTags were manually manipulated while a Python program to detect the AprilTags was executed on the Raspberry Pi. The position of the Raspberry Pi camera was also manipulated at different angles and the Python program successfully displayed a live feed from the camera onto a computer monitor. Successful detection and decoding of the AprilTags were easily achieved at varying angles and distances between the camera and AprilTags. Marker detection and decoding performed quite well even while the markers were in motion. The camera was able to successfully detect and decode the AprilTags under all circumstances if they were fully within the camera's field of view. The only time detection failed was when part of the AprilTag was outside of the camera's field of view.

Figure 3: Outdoor *(top)* and indoor *(bottom)* courses used during experiments

After the Raspberry Pi and Raspberry Pi Camera Module v2 were attached to the UAV, marker detection and decoding were evaluated again. Since the Raspberry Pi was attached to the UAV, the pictures taken by the camera were saved to the Raspberry Pi for viewing later. A VNC viewer was also implemented by a remote computer to access the graphical interface of the Raspberry Pi and obtain a live feed from the camera. However, the live feed did not display well using the VNC viewer, so the pictures saved to the Raspberry Pi were used instead. In-flight pictures were taken while the UAV was navigating towards the markers. Similar to the initial marker detection and decoding experiments, the camera was successfully able to detect and decode the AprilTags if they were fully within the camera's field of view. However, due to the way the Raspberry Pi camera was mounted to the UAV, the pictures were inverted. To correct the picture inversion, an OpenCV function was used to rotate the pictures 180º. Marker detection and decoding were found to be quite robust during UAV flight.

4.2 Hardware Performance

4.2.1 UAV Hardware

The evaluation of the performance of the UAV's hardware began by testing the onboard sensors, cameras, and battery life under simulated flight conditions before any external hardware was attached to the UAV. A Python program was implemented to access and display the values of the UAV's pitch, roll, yaw, altitude, magnetometer, remaining battery life (as a percentage), and Wi-Fi link quality. The UAV's front and bottom cameras were also accessed with a Python program that displayed a live feed from the cameras onto a computer monitor. The UAV was manually manipulated while executing the programs. The UAV seemed to calibrate its sensors based on its initial pose during a manual reset or while connecting its battery. Thus, the pitch, roll, yaw, altitude, and magnetometer values were relatively consistent if the pose of the UAV was consistent during a manual reset or while connecting its battery. The yaw angle and magnetometer values were also successfully acquired during a simulated flight through an indoor course and plotted on a graph. Battery performance was also found to be stable while manually manipulating the UAV and experienced only small decreases in percentage with each execution of a program. However, the UAV's cameras did not perform well. Both the forward and downward-facing cameras were successfully accessed, and a live feed was displayed on a computer monitor, yet the lag produced by the cameras was significant enough to render them unusable for this research. Additional time spent researching different methods of accessing

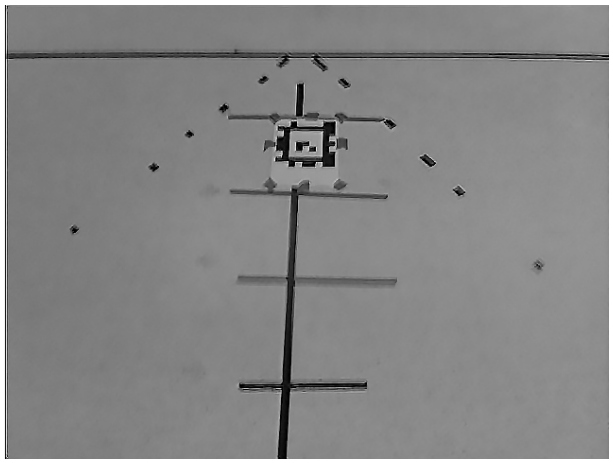

Figure 4: Picture taken from UAV of detected AprilTag during successful indoor flight

and manipulating the cameras may have yielded better results. However, the Raspberry Pi camera was used instead since only one camera was needed and it demonstrated adequate performance during earlier experiments. Wi-Fi link quality was also evaluated and found to produce adequate results.

Although the UAV's onboard sensors displayed consistent and stable performance while simulating flight, in-flight performance was found to be inconsistent and unstable. After mounting the Raspberry Pi and Raspberry Pi Camera Module v2 to the UAV, the UAV's sensors and battery life were evaluated in-flight. Of particular importance to this research was the yaw angle and magnetometer values, which seemed to be entirely arbitrary. One possible explanation for this phenomenon may be interference from the Raspberry Pi and its lithium-ion battery power supply that was connected via the GPIO pins. Another possible explanation could be interference caused by other nearby electronic devices. The UAV's battery also suffered significant decreases in performance during real flight conditions. The specifications state that the battery is capable of twelve minutes of flight time, however, only about half of the stated capability was realized. Battery performance increased after obtaining new batteries but remained inadequate for flight after falling below about forty percent. Wi-Fi link quality was stable during flight, although the UAV did experience instances of unresponsiveness to remote commands issued to initiate a safe landing sequence.

Structural components of the UAV were also indirectly tested for resiliency during flight. Many flights were conducted during the research, some were stable, while others were not. The structural integrity of the components of the UAV was maintained during stable flights. However, a substantial percentage of the minor crashes experienced by the UAV produced significant deformation to the stainless steel shafts connecting the rotor wings to the UAV and rendered the UAV flightless until the shafts were replaced as the deformity caused substantial drift. While replacing the stainless steel shafts, the E-clips holding the shafts in place were also easily damaged and had to be replaced often. The other structural components of the UAV displayed adequate resiliency.

4.2.2 External Hardware

Performance of the Raspberry Pi Camera Module v2 and external Adafruit LSM303DLHC triple-axis accelerometer/magnetometer was initially evaluated before mounting the Raspberry Pi, Raspberry Pi Camera Module v2, and LSM303DLHC to the UAV. A Python program was implemented to access and display the values obtained from the LSM303DLHC as well as a live feed from the Raspberry Pi Camera Module v2 onto a computer monitor. The values from the LSM303DLHC were successfully acquired and produced consistent results while manually manipulating the LSM303DLHC. The Raspberry Pi Camera Module v2 displayed adequate performance when evaluated by visually inspecting its live feed for lag time, field of view,

Figure 5: Parrot AR.Drone 2.0 with Raspberry Pi & Raspberry Pi Camera Module v2 mounted to the hull

and distance of AprilTag detection while manually manipulating the camera.

Performance of the Raspberry Pi Camera Module v2 and external Adafruit LSM303DLHC triple-axis accelerometer/magnetometer was reevaluated after mounting the Raspberry Pi, Raspberry Pi Camera Module v2, and LSM303DLHC to the UAV. Pictures acquired from the camera and saved to the Raspberry Pi during UAV flight revealed adequate camera performance, whereas the data acquired from the LSM303DLHC revealed compromised performance. The erratic behavior of the LSM303DLHC is thought to be a consequence of the instability of the UAV during flight, as well as possible interference from the Raspberry Pi and its lithium-ion battery power supply.

4.3 UAV Flight Performance

Similar to the performance evaluations of other components in the system, the performance of the UAV's flying capabilities was evaluated both before and after mounting external hardware to the UAV, with initial evaluations occurring before mounting external hardware. Successful flight was achieved by implementing a Python program that used the PS-Drone API to manipulate the altitude, position, orientation, and landing of the UAV. The UAV remained satisfactorily stable during flight and generally corrected most of the unsolicited positional drift that occurred due to momentum after the execution of each command.

After demonstrating satisfactory flight capabilities, the Raspberry Pi, Raspberry Pi Camera Module v2, and LSM303DLHC triple-axis accelerometer/magnetometer were mounted to the UAV, and flight performance was re-evaluated using the same Python program. Flight performance suffered from significant drift while hovering after executing commands to manipulate the pose of the UAV. This resulted in an inability

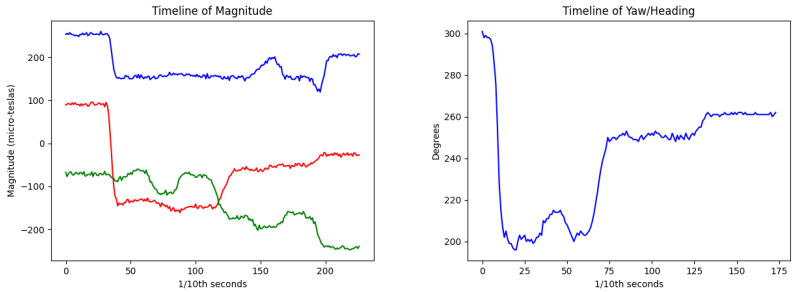

Figure 6: Recorded magnitude data of drone during a simulated indoor course run

of the UAV to achieve adequate altitude. Due to a loss in performance, a manual reset of the UAV was performed. After the manual reset, flight capabilities were partially restored, however, they remained inferior to the previously achieved levels of performance. Although inferior to previous levels, the flight capabilities were deemed to be sufficient for the research.

4.4 Autonomous UAV Navigation

After each component of the system was tested and demonstrated adequate performance, an indoor course was constructed. A Python program was developed to test the entire system and validate the proposed solution to the problem of autonomous UAV navigation in a GPS-denied outdoor environment using discontinuous visual contact with fiducial markers. Initial testing was conducted inside the lab on campus. Through many trial runs and algorithm refinement, successful autonomous UAV navigation was partially achieved. The decrease in flight capabilities that was noted after attaching the external hardware to the UAV during the flight performance evaluations was fully realized while navigating the course. Unwanted positional drift after executing commands to manipulate the pose of the UAV was magnified to the extent that the UAV could no longer correct for. The significance of the drift ultimately rendered the UAV incapable of navigating the course, however, partial course navigation was still achieved. After initiating takeoff, the UAV successfully increased its altitude and proceeded forward while taking pictures of the environment with the Raspberry Pi camera. When the first AprilTag was detected in an image, the UAV ceased forward motion, decoded the AprilTag, and used the data associated with the current AprilTag's unique identification number to acquire the previously computed positional data of the next AprilTag. The UAV then adjusted its yaw angle to navigate in a straight-line path to the next AprilTag. However, this is the step that produced significant drift. Even a rotation of just seventy degrees produced enough drift to render the UAV incapable of proceeding through the course. After conducting many indoor flights, a similar course was then constructed outside to test the system. Just as with the indoor flights, the UAV was unable to navigate the course.

5 Conclusion

Technological advancements over the years have enabled the widespread use of UAVs in many different sectors. As the number of applications for UAVs has dramatically increased, so has the need for fully autonomous navigational capabilities. Many UAVs have achieved autonomous navigation using GPS and other technologies, such as cameras and LiDAR. Another technology developed to assist in autonomous robotic navigation is fiducial markers. Fiducial markers are particularly useful in GPS-denied environments, as they can be placed in an environment and serve as a physical waypoint for a UAV. Due to the limitations of GPS, prior research has been conducted studying different methods of autonomous UAV navigation in GPS-denied environments. However, previous research using only fiducial markers and onboard sensors for

autonomous UAV navigation required an onboard camera to have constant visual contact with at least one fiducial marker. In this paper, autonomous UAV navigation in a GPS-denied outdoor environment using discontinuous visual contact with fiducial markers was explored.

The research conducted for this paper uses several different technologies including the AprilTag fiducial marker system, a Parrot AR.Drone 2.0, an Adafruit LSM303DLHC triple-axis accelerometer/magnetometer, a Raspberry Pi 3B, and a Raspberry Pi Camera Module v2. Along with the hardware components of the system, many different software packages and technologies were used including the Python programming language, OpenCV, the PS-Drone API, and the software for the Adafruit LSM303DLHC triple-axis accelerometer/magnetometer and AprilTag fiducial marker system. These technologies were evaluated both independently and in combination with each other, and both indoor and outdoor courses were constructed to evaluate the viability of autonomous UAV navigation in a GPS-denied outdoor environment using discontinuous visual contact with fiducial markers. Although autonomous UAV navigation was only partially achieved due to substantial UAV drift during flight rendering the UAV incapable of navigating the entire course, the results are promising. The technologies implemented generally met or exceeded performance and interoperability requirements. With additional research and experimentation, the performance issues with the magnetometers can likely be mitigated. Likewise, additional algorithms could be developed to minimize UAV drift, or a modern UAV with enhanced resiliency and advanced drift control mechanisms could be used with the proposed system to achieve fully autonomous UAV flight.

References Cited

[1] J. Bacik, B. Cain, F. Durovsky, P. Fedor, and D. Perdukova, "Autonomous flying with quadrocopter using fuzzy control and aruco markers," *Intel Service Robotics*, vol. 10, p. 185–194, 2017. DOI: 10.1007/s11370-017-0219-8.

[2] M. Kalaitzakis, B. Cain, S. Carroll, A. Ambrosi, C. Whitehead, and N. Vitzilaios, "Fiducial markers for pose estimation," *Journal of Intelligent & Robotic Systems*, vol. 101, no. 71, 2021. DOI: 10.1007/s10846-020-01307-9.

[3] F. Lamberti, A. Sanna, G. Paravati, P. Montuschi, V. Gatteschi, and C. Demartini, "Mixed marker-based/marker-less visual odometry system for mobile robots," *International Journal of Advanced Robotic Systems*, vol. 10, no. 5, 2013. DOI: 10.5772/56577.

[4] M. Nahangi, A. Heins, B. McCabe, and A. Schoellig, "Automated localization of uavs in gps-denied indoor construction environments using fiducial markers," *2018 Proceedings of the 35th ISARC, Berlin, Germany*, pp. 88–94, 2018. ISBN: 978-3-00-060855-1.

[5] E. Olson, "Apriltag: A robust and flexible visual fiducial system," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3400–3407, 2011. DOI: 10.1109/ICRA.2011.5979561.

[6] L. K. Seng, M. Ovinis, Nagarajan, R. Seulin, and O. Morel, "Vision-based state estimation of an unmanned aerial vehicle," *Trends in Bioinformatics*, vol. 10, no. 1, pp. 11–19, 2017. DOI: 10.3923/tb.2017.11.19.