Visual Anchoring: Fixed-Wing UAS Orbit Stabilization About a Visual Anchor Point Without GPS Dependence

Ryan J Grimes

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Aeronautics & Astronautics

University of Washington

2018

Committee:

Christopher Lum, Chair

Juris Vagners

Program Authorized to Offer Degree:
William E. Boeing Department of Aeronautics and Astronautics

University of Washington

Abstract

Visual Anchoring: Fixed-Wing UAS Orbit Stabilization About a Visual Anchor Point Without GPS Dependence

Ryan J Grimes

Chair of the Supervisory Committee:
Research Assistant Professor Christopher Lum
William E. Boeing Department of Aeronautics and Astronautics

This paper describes a system for stabilizing an orbit of an unmanned aircraft system (UAS) around a target. The system utilizes an inner/outer loop controller architecture to achieve a stabilized, coordinated turn orbit about a point. The innovation is that the primary inputs to this controller are obtained from a vision system that computes the slant range to the target based on images streamed from the aircraft making vision the primary sensor modality for achieving a stabilized orbit. Both the vision system for estimating slant range and the associated controller for achieving a stabilized orbit are discussed. Varying levels of simulation including software-in-the-loop (SITL) are presented before discussing flight testing of the custom built UAS.

The views expressed are those of the author and do not reflect the official policy or position of the US Air Force, Department of Defense or the US Government.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Abbreviations

AFSL	Autonomous Flight Systems Laboratory
CONDOR	Camera Operated Navigation Done Outside (GPS) Ranges
FAA	Federal Aviation Administration
GCS	Ground Control Station
GPS	Global Positioning System
LiPo	Lithium Polymer
PPM	Pulse Position Modulation
PWM	Pulse Width Modulation
RC	Remote Control
RCAM	Research Civil Aircraft Model
RxMUX	Servo Multiplexer Unit
SITL	Software In The Loop
SPOI	Sensor Point of Interest
UDP	User Datagram Protocol
UAS	Unmanned Aerial System

Symbols

d_{tgt}	Euclidean distance to target (slant range)
D_C	Ground radius between aircraft and center of image
D_X	Ground distance between Y-Axis intercept and target
D_Y	Ground radius between aircraft and Y-Axis intercept
D_T	Ground radius between aircraft and target

F_b	Body frame attached to aircraft
F_c	Camera frame attached to camera
F_{center}	Centripetal force
F_n	North/East/Down frame (inertial)
g	Earth gravitational acceleration
h_{AGL}	Altitude above target (above ground level)
h_{des}	Desired altitude of orbit
K_{D_ϕ}	Roll inner loop derivative gain
K_{D_θ}	Pitch inner loop derivative gain
$K_{D_{outer}}$	Outer loop derivative gain
K_{I_h}	Altitude hold integral gain
K_{P_h}	Altitude hold proportional gain
$K_{P_{outer}}$	Outer loop controller proportional gain
$K_{P_{\dot{\psi}}}$	Turn coordinator proportional gain
K_{P_ϕ}	Roll inner loop proportional gain
K_{P_θ}	Pitch inner loop proportional gain
L_C	Slant range between aircraft and center of image
L_Y	Slant range between aircraft and Y-Axis intercept
L_T	Slant range between aircraft and target
m	Mass of aircraft
N	Nonlinear coordinate correction factor
P_E	Position east
P_N	Position north
p	Bank rate of aircraft
q	Pitch rate of aircraft
r	Heading (yaw) rate of aircraft

R	Actual radius of orbit
$\dot{R}$	Actual radius rate
R_{des}	Desired radius of orbit
S_x	Camera horizontal resolution
S_y	Camera vertical resolution
t	Time
V_A	Airspeed of aircraft
$V_{CM/e}$	Velocity of aircraft w.r.t. inertial frame
$V_{W/e}$	Velocity of wind w.r.t. inertial frame
$V_{tgt/e}$	Velocity of target w.r.t. inertial frame
W	Weight of aircraft
X_{tgt}	X cooridinate of target
Y_{tgt}	Y cooridinate of target

Greek symbols

ε_R	Radius error
ϕ	Euler angle of bank of aircraft
ϕ_{err}	Bank angle error
ϕ_{ref}	Reference bank angle
$\dot{\phi}$	Bank rate
θ	Euler angle of pitch of aircraft
θ_{err}	Pitch angle error
θ_{ref}	Reference pitch angle
$\dot{\theta}$	Pitch rate
ψ	Euler angle of yaw of aircraft
$\dot{\psi}$	Heading (yaw) rate of aircraft
$\dot{\psi}_{err}$	Heading rate error

$\dot{\psi}_{ref}$	Reference heading rate of aircraft
θ_c	Euler angle of pitch of camera (tilt)
θ_X	Angular position of the target X-Axis intercept
θ_Y	Angular position of the target Y-Axis intercept
θ_V	Camera vertical field of view
θ_H	Camera horizontal field of view
ψ_c	Euler angle of yaw of camera (pan)
λ_c	Zoom factor of camera
ΔA	Aileron deflection
ΔE	Elevator deflection
ΔR	Rudder deflection

ACKNOWLEDGMENTS

I would like to thank Dr.Christopher Lum for allowing me to work on the Visual Anchoring project and for his support and guidance during my time as a member of the Autonomous Flight Systems Laboratory (AFSL). I would also like to thank Tadej Kosel for his dedication and support to the project. This project would not have been possible without support and funding from Insitu and the University of Washington Royalty Research Fund. In addition, I would like to thank additional AFSL members who contributed to the project : Hannah Rotta, Helen Kuni, Ravi Patel, Simeng Zhu, Connor Kafka, Alex Banh, Selena Lui, and Joshua Brockschmidt. Finally, I would like to thank my friends and family for their tremendous support.

DEDICATION

For my parents Cathy and Terry Grimes

Thank you for your unwavering love and support

Chapter 1
INTRODUCTION

1.1 Problem Statement

This work focuses on developing a control system to stabilize an UAS in an orbit above a target that is either moving or in a wind field as shown in Figure 1.1. The aircraft is equipped with a standard definition camera that can stream video imagery to a ground control station (GCS) in real-time. The system will compute appropriate control inputs such that the system stabilizes an orbit around the target using this streamed imagery as the primary sensor modality.

The novelty in this approach is that GPS is not used to establish the location of the aircraft and geo-locating the target is not required. A major driver for this design is the fact that the GPS signal is subject to a variety of potential degradations, such as loss of path (indoors or under foliage), multi-path solutions (in between buildings or in difficult terrain), sunspots, or direct interference. GPS jamming is possible and accessible[6]. Active jamming of GPS signals is a major concern of the FAA and has led to disruption of aviation services on more than one occasion[24].

1.2 Prior Work

The problem of establishing an orbit[19] or otherwise tracking a moving target in wind[18] has been studied before[7]. Many of these works focus on the control law to achieve this orbit and assume that the position of both the aircraft and the target are known. However, if GPS is denied then other sensors are required to obtain a relative measurement from the aircraft to the target. This work combines the control problem of establishing a stable orbit

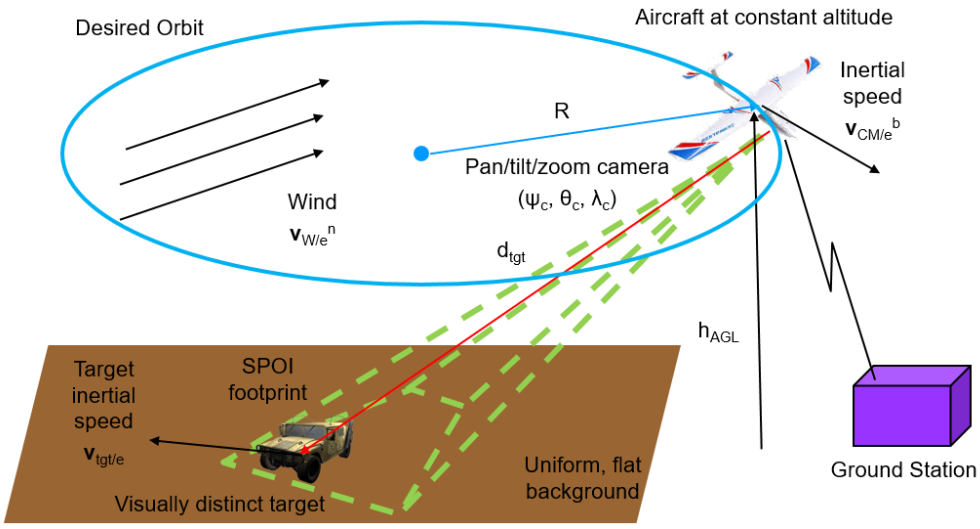


Figure 1.1: Project vision showing a UAS orbiting a target in a wind field while streaming imagery to the GCS.

with the machine vision problem of extracting the target's relative position from the aircraft via video imagery.

Vision based navigation for UAS has been studied by several groups in the past[5, 20]. Others have looked at localizing targets[21] or finding unique features such as wildfires[15] using UAS. Recently, image processing from UAS has become popular in the realm of precision agriculture[16], collision avoidance[23], and searching[9, 11, 13].

Chapter 2

ORBIT CONTROLLER

The control law for establishing an orbit is executed onboard the aircraft. However, the actual radius input to the controller is computed on the GCS via the vision system (described in Chapter 3). Further details on the UAS and associated hardware is discussed in Section 5.1.

2.1 Control Law

The architecture of the orbit controller is shown in Figure 2.1. The orbit controller is composed of two separate controllers working in series. The outer loop controller serves to detect the deviation from the desired orbit radius and compute the appropriate input signal for the inner loop controller. The inner loop controller then consumes this signal as well as aircraft sensor measurements to stabilize the aircraft on the desired orbit and altitude. Under the assumption that the autopilot includes an airspeed hold function, the control problem becomes how to command the aileron, elevator and rudder to perform a steady coordinated turn. The typical approach for this control problem is to decouple the lateral and directional axes and design controllers to command appropriate control inputs[22]

2.1.1 Coordinated Turn

In an ideal coordinated turn, net forces in the body y-axis direction are zero[22]. If it is assumed that the aircraft maintains a constant altitude, or zero vertical acceleration while orbiting, the relation between centripetal force, F_{center}, aircraft weight, W, and bank angle, ϕ, can be written as

$$\tan\phi = \frac{W}{F_{center}} \tag{2.1}$$

We also know that the tangential velocity, V_A, is related to the heading rate by

$$V_A = \dot{\psi}\, R \tag{2.2}$$

$$F_{center} = \frac{m\, V_A^2}{R} = m\, V_A\, \dot{\psi} \tag{2.3}$$

Finally, the approximated expression of bank angle, ϕ, to perform a coordinated turn is kinematically related to the heading rate, $\dot{\psi}$, as

$$\phi = \arctan\frac{V_A\, \dot{\psi}}{g} \tag{2.4}$$

In order to keep the axis of the aircraft smoothly aligned with the direction of motion during a steady rate turn, the aircraft requires both lateral and directional inputs to command banking and yawing simultaneously. To enter and maintain a steady rate turn, the lateral and directional command can be coupled so that a reference heading rate, $\dot{\psi}_{ref}$, can be converted into a reference bank angle, ϕ_{ref}.

$$\phi_{ref} = \arctan\frac{V_A\, \dot{\psi}_{ref}}{g} \tag{2.5}$$

The inner loop controller uses a reference heading rate, $\dot{\psi}_{ref}$, with respect to a user defined orbit radius, R_{des}, as well as a heading rate error, $\dot{\psi}_{err}$, computed by the outer loop controller to command the lateral axis by converting the heading rate error, $\dot{\psi}_{err}$, into a reference bank angle, ϕ_{ref}.

2.2 Outer Loop Controller

The primary goal of the outer loop controller is to detect the deviation from the desired orbit radius, R_{des}, and minimize its error to zero. The guidance law can be expressed as, $\varepsilon_R = R - R_{des}$, $\varepsilon_R \to 0$.

In order to achieve this goal, the outer loop controller utilizes a proportional-derivative controller to compute the heading rate error, $\dot{\psi}_{err}$, from the actual radius information obtained by the vision system (Chapter 3). By using information from the vision system, the outer loop controller does not require GPS information to determine relative position of the aircraft to the target. The output from the outer loop controller becomes

$$\dot{\psi}_{err} = -(K_{P_{outer}}(R - R_{des}) - K_{D_{outer}}\dot{R}) \tag{2.6}$$

2.3 Inner Loop Controller

As an integral part of the orbit controller, the inner loop controller is responsible for the stability augmentation of the aircraft's lateral-directional axes while tracking the desired orbit and maintaining a constant altitude specified by the operator. The inner loop controller consumes heading rate error, $\dot{\psi}_{err}$, calculated from the outer loop controller combined with sensor measurements such as the aircraft attitude and airspeed to produce the primary control surface commands for pitch, roll and yaw.

There are three control loops for the vertical, lateral and directional axes respectively. The lateral control loop works together with the directional control loop to correct and compensate for the aircraft's deviation from the desired orbit and help maintain a coordinated turn. The vertical control loop helps maintain a user specified desired altitude, h_{des}.

2.3.1 Roll Inner Loop

The roll inner loop controller is responsible for producing the aileron command component to satisfy the turn coordination constraint. The combination of the roll and roll rate feedback control helps to achieve the proper lateral transient response and define the control bandwidth.

Reference heading rate, $\dot{\psi}_{ref}$, can be calculated from the heading rate error, $\dot{\psi}_{err}$, from the outer loop controller, airspeed, V_A, and actual radius, R, using Eq. (2.2).

$$\dot{\psi}_{ref} = \dot{\psi}_{err} + \frac{V_A}{R} \tag{2.7}$$

Bank angle error, ϕ_{err}, is derived from reference bank angle, ϕ_{ref}, from Eq. (2.5) and measured bank angle, ϕ.

$$\phi_{err} = \arctan\left(\frac{V_A\,\dot{\psi}_{ref}}{g}\right) - \phi \tag{2.8}$$

The control algorithm to compute aileron deflection, ΔA, is a typical proportional-derivative scheme that consumes bank angle error, ϕ_{err}, from Eq. (2.8) and bank rate, $\dot{\phi}$, from the inertial measurement.

$$\Delta A = -\left(K_{P_\phi}\,\phi_{err} - K_{D_\phi}\,\dot{\phi}\right) \tag{2.9}$$

2.3.2 Turn Coordinator

The goal of the turn coordinator is to provide a rudder command as part of the turn coordination constraint. The turn coordinator is a proportional controller with an adjustable gain. As the aircraft banks to maintain a desired heading rate, the turn coordinator provides rudder commands that keep the aircraft heading tangential to the orbit path. Rudder deflection, ΔR, is calculated as follows.

$$\Delta R = K_{P_{\dot{\psi}}}(\dot{\psi}_{ref} - \dot{\psi}) \tag{2.10}$$

2.3.3 Altitude Hold

The altitude hold controller uses aircraft pitch to hold a constant orbit altitude specified by the operator independent of orbit radius. The altitude hold controller first computes the reference pitch angle, θ_{ref}, through a proportional-integral controller using the altitude error signal as seen in Eq. 2.11. Next, the elevator surface deflection, ΔE, is computed by a proportional-derivative controller using the pitch error signal as seen in Eq. 2.12.

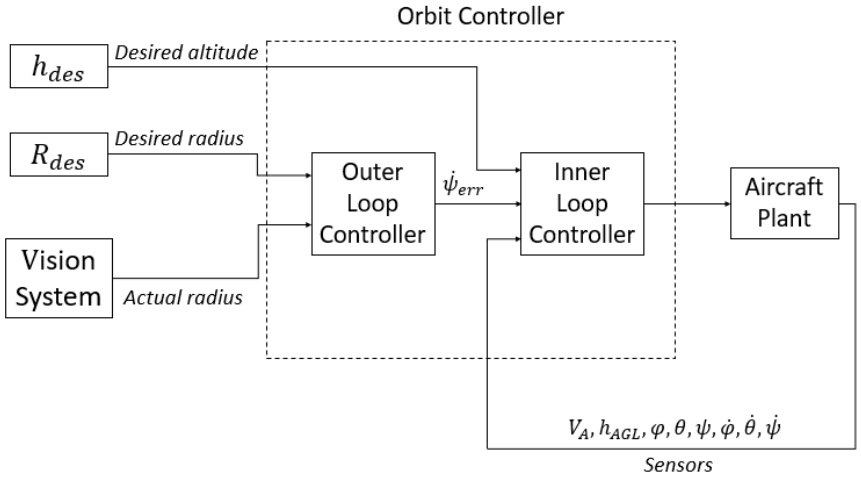


Figure 2.1: Orbit controller block diagram.

$$\theta_{ref} = K_{P_h}\left(h_{des} - h_{AGL}\right) + K_{I_h}\int_0^t \left(h_{des} - h_{AGL}\right) dt \tag{2.11}$$

$$\Delta E = -\left(K_{P_\theta}\left(\theta_{ref} - \theta\right) - K_{D_\theta}\dot{\theta}\right) \tag{2.12}$$

Chapter 3

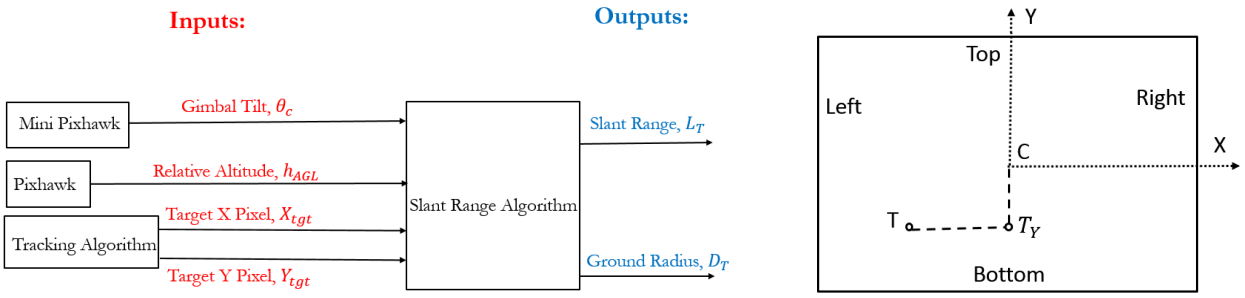
VISION SYSTEM

The primary goal of the vision system is to perform real-time image processing on video streamed from the aircraft to the ground station in order to compute the slant range and ground radius between the aircraft and the target of interest. The computed ground radius can then be used by the previously defined orbit controller as the actual radius input. This section describes the algorithm that the vision system utilizes to perform these functions.

3.1 Slant Range Algorithm

As seen in Fig. 3.1(a), the slant range or ground radius from the camera to the target can be calculated using the Slant Range Algorithm. The inputs to the algorithm are camera gimbal tilt angle, θ_c (relative to the local horizon), the relative altitude of the aircraft, h_{AGL}, and the X and Y coordinates of the target within the image frame, X_{tgt} and Y_{tgt}. The algorithm requires the assumption that the ground being observed by the camera is flat. Since this algorithm is designed for use without GPS information, h_{AGL} is provided to the system as a pressure altitude measurement from a barometer sensor. The image frame coordinates of the target are provided from the tracking algorithm described in Section 3.2.

The image frame can be visualized in Fig. 3.1(b), where point C is the center of the image, point T is the target in the image, and point T_Y is the Y-Axis intercept of the target in the image. The origin of the axes is in the center of the image with positive X to the right and positive Y upward.



(a) Slant Range Algorithm block diagram

(b) Origin and positive directions in the image frame.

Figure 3.1: Slant Range Algorithm diagrams.

3.1.1 Y-Axis Geometry

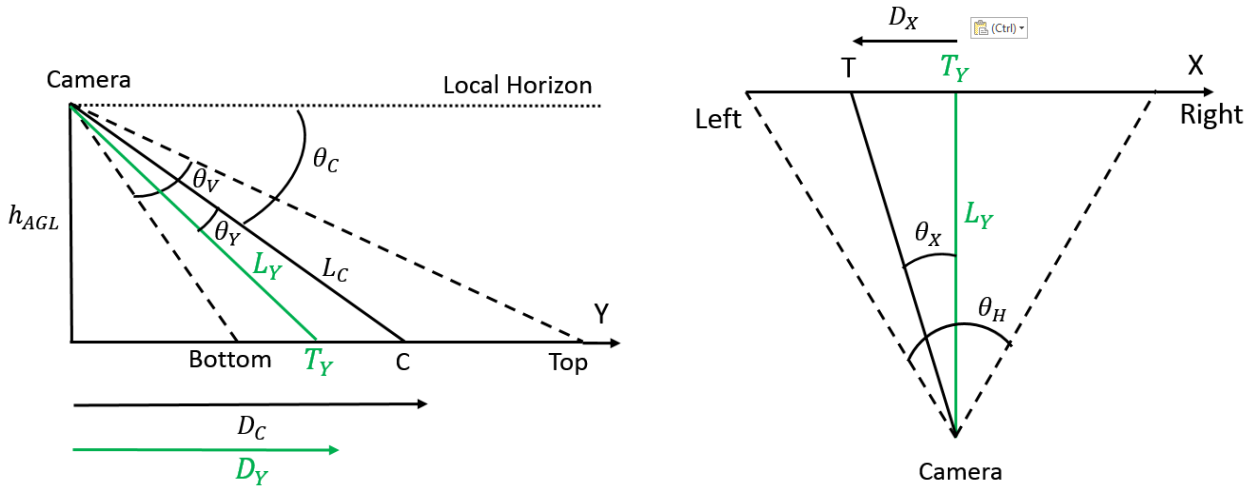
Through inspection of the Y-Axis physical geometry, seen in Fig. 3.2(a), the slant range to the center of the image, L_C and the ground radius to the center of the image, D_C can be computed from the following equations.

$$L_C = \frac{h_{AGL}}{\tan(\theta_C)} \tag{3.1}$$

$$D_C = \frac{h_{AGL}}{\sin(\theta_C)} \tag{3.2}$$

Similarly, if the angular position of the target Y-Axis intercept, θ_Y, is taken into account, the slant range to the target Y-Axis intercept, L_Y, and the ground radius to the target Y-Axis intercept, D_Y, can be computed from the following equations, where θ_V is the vertical field of view of the camera, Y_{tgt} is the Y coordinate of the target, and S_y is the vertical resolution of the image.

$$L_Y = \frac{h_{AGL}}{\sin(\theta_C + \theta_Y)} \tag{3.3}$$



(a) Y-axis geometry.

(b) X-axis geometry.

Figure 3.2: Physical geometry associated with x and y axis.

$$D_Y = \frac{h_{AGL}}{\tan(\theta_C + \theta_Y)} \tag{3.4}$$

$$\theta_Y = \theta_V \frac{Y_{tgt}}{S_y} \tag{3.5}$$

3.1.2 X-Axis Geometry

Through inspection of the X-Axis physical geometry, seen in Fig. 3.2(b), the ground distance from the target Y-axis intercept to the target, D_X, can be found through the following equations, where θ_X is the angular position of the target X-Axis intercept, θ_H is the horizontal field of view of the camera, X_{tgt} is the X coordinate of the target, and S_x is the horizontal resolution of the image. Since the magnitude of the ground distance from the target Y-Axis intercept to the target is the value of interest, the absolute value of θ_X is used to ensure positive ground distance values.

$$D_X = L_Y \tan(|\theta_X|) \tag{3.6}$$

$$\theta_X = \theta_H \frac{X_{tgt}}{S_x} \tag{3.7}$$

3.1.3 Combined Axes Geometry

By combining the information from the X and Y axes, the total slant range between the aircraft and the target, L_T, and the total ground radius between the aircraft and the target, D_T, can be found in the following equations.

$$D_T = \sqrt{D_X^2 + D_Y^2} \tag{3.8}$$

$$L_T = \sqrt{D_T^2 + h_{AGL}^2} \tag{3.9}$$

3.1.4 Nonlinear Coordinate Correction Factor

The Slant Range Algorithm assumes that the pixel to field of view ratio throughout the image is constant. Specifically, it assumes that you can linearly relate pixel location within the image to an effective field of view location within the image. As shown in Section 7.1, this assumption is reasonable when objects portrayed in the image are at similar distances away from the camera. However, as objects in the image begin to vary in distance away from camera significantly, objects further away from the camera appear smaller in the image than objects closer to the camera. This will cause the pixel to field of view ratio in the image to vary depending on the scaling of the objects. This effect is particulary pronounced in the Y axis specifically for tilted images. As seen in Figure 3.3, a nonlinear approximation can be used to correct for the dynamic pixel to field of view ratio throughout the Y axis of the image. It is important to note that if the linear approximation is assumed when significant nonlinearity exists, the pixel location of objects will appear higher up on the image than they should. This effect causes tilted images to appear as if they are being viewed at a different tilt angle than the current actual gimbal tilt angle.

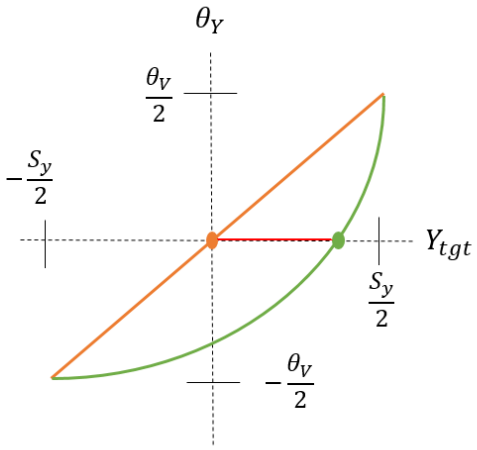


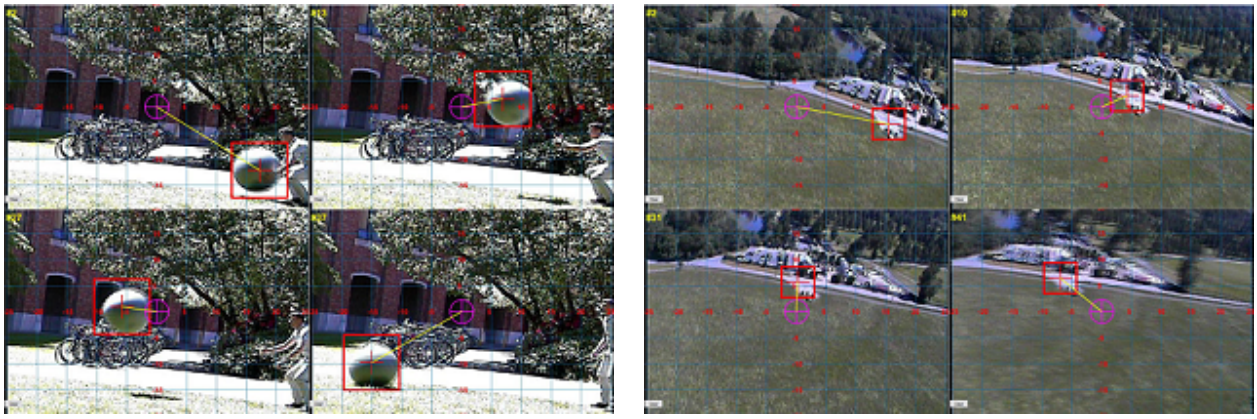
Figure 3.3: Plot of nonlinear coordinate correction factor effect.

This nonlinear coordinate correction factor can be implemented through the following equation, where N is the nonlinear coordinate correction factor. The value of the nonlinear coordinate correction factor can be determined by testing the Slant Range Algorithm and fitting the correction factor to known truth data.

$$\theta_Y = \theta_V \left[\left(\frac{Y_{tgt} + \frac{S_y}{2}}{S_y} \right)^N - 0.5 \right] \tag{3.10}$$

3.2 Tracking Algorithm

The vision system utilizes a real-time compressive tracking scheme[25] formulated as a binary classification problem using a naive Bayes classifier with online update in the compressed domain. The tracking algorithm runs in real-time and performs favorably against state-of-the-art algorithms on challenging sequences in terms of efficiency, accuracy and robustness. The tracking algorithm is very useful for the application of visual anchoring because it provides the current location of the target within the image for the Slant Range Algorithm. Tests were performed both on the ground and in the air to ensure the tracking



(a) Tracking an exercise ball as it bounces across the scene

(b) Tracking the parking lot during a flight test.

Figure 3.4: Initial tracking algorithm tests

algorithm was able to perform properly in real time. On the ground a large exercise ball was filmed and the algorithm worked very well as shown in Fig. 3.4(a). The red rectangular frame (target frame) in the image is driven by the tracking algorithm.

Tests were also conducted in the air using the custom built CONDOR aircraft (Fig. 5.1(a)). From the air, the system was tasked with following objects on the ground such as a tent, car and some other ground features such as large trees, parking lots, etc. Figure 3.4(b) shows an example of tracking the parking lot where the ground crew was located during a flight test. Figure 3.4(b) shows four frames from the image with different positions of the parking lot inside the image frame. The algorithm works very well on these ground features for visual anchoring tracking purposes. The following parameter values were used for the tracking algorithm: the number of trained negative samples (20), radical scope of positive samples (5) and size of search window (50) for ground and flight tests.

3.2.1 Target Tracking Limitations

While the tracking algorithm performs well, there are a few limitations for the visual anchoring application. It is important that the object being tracked remains in the camera field of view at all times. If the tracked target leaves the field of view, the tracking algorithm will not be able to re-identify the target when it returns in the field of view. Therefore, it is paramount that the camera gimbal is pointed so that the target never leaves the current tracked image. In addition, the tracked target cannot move too quickly between image frames. If there is a sudden change in position of the target between image frames, the tracking algorithm may not be able to keep tracking the target successfully. Therefore, the camera gimbal must be commanded to pan and tilt at reasonable rates. Sudden changes in pan and tilt angles of the camera gimbal could cause sudden shifts in the target position in the image frames.

Chapter 4
SIMULATION ENVIRONMENT

Due to the complex nature of the control system, a simulation environment is necessary to validate proper operation of the various components of the system before moving to flight testing. As described previously, the two main components of the system are the orbit controller and the vision system. Two simulation enviornments were used to validate the orbit controller design before moving to flight testing. The vision system was initially validated through ground testing that is described in Secton 7.1.

4.1 Matlab/Simulink Simulation

The primary controller design took place in Matlab/Simulink. The popular Research Civil Aircraft Model (RCAM)[8] was used as a plant model since the actual modeling parameters of the CONDOR UAV were not available at the controller design phase. RCAM is a six-degree of freedom, nonlinear aircraft model that represents a medium sized, two engine transport jet aircraft. Navigation equations were incorporated with the RCAM to track the absolute position of the aircraft in the simulation environment. The vision system was also simulated to output the actual radius measured from the anchor point defined by the user. Actual simulation results of this system are discussed in Section 6.1.

4.2 JSBSim, Mission Planner, Arduplane Simulation

Once the orbit control algorithm was validated in the Matlab/Simulink environment, the algorithm was ported into the Arduplane firmware so it could perform on a UAS. The Arduplane implementation of the flight controller was primarily tested in a Software-in-the-Loop (SITL) simulated environment[4]. The SITL is a special build of the Arduplane

firmware that can emulate an autopilot system without the hardware. Utilizing JSBSim as the core flight dynamics model, the SITL simulated a small UAS running the modified version of Arduplane (Section 5.2). A modified version of Mission Planner (Section 5.3) was used as the ground control station (GCS) to communicate and send commands to the simulator. Results of this simulation are discussed in Section 6.2.

Chapter 5
FLIGHT TEST ENVIRONMENT

Once sufficient simulation and analysis was completed, the algorithms were implemented on aircraft hardware for flight testing. This section discusses the aircraft hardware, the GCS hardware, and how they interact during flight testing.

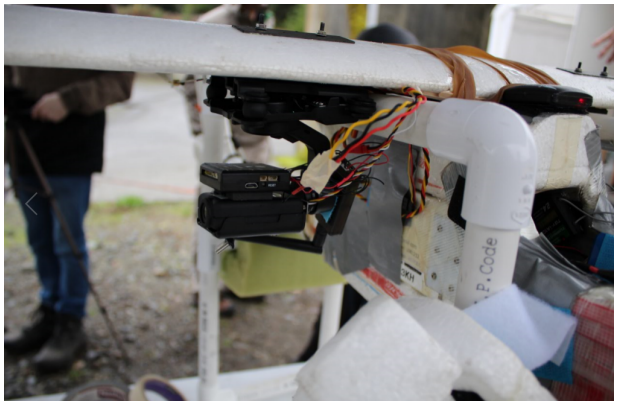
5.1 The CONDOR UAS

The Camera Operated Navigation Done Outside (GPS) Ranges (CONDOR) UAS is a highly customized system based on the popular Skywalker 1900 airframe as shown in Fig. 5.1(a). The airframe is upgraded with systems for autonomous flight and video transfer. The control surfaces consist of ailerons (no flaps), rudder and elevator. An additional safety feature is an RxMUX unit (8-Channel Servo Multiplexer) device which protects the aircraft in case of total flight computer failure. It allows the pilot to take manual control of the aircraft during flight and has direct connection between the RC receiver and the control surfaces (not routed through the flight computer). This was added as a precaution given that many flight tests are running customized versions of the Arduplane firmware.

The on-board control system is composed of a Pixhawk flight computer (autopilot), data link modem for telemetry, GPS receiver with magnetometer (magnetic compass), pitot tube with dynamic pressure sensor, RC receiver, PWM (Pulse Width Modulation) to PPM (Pulse Position Modulation) converter and a LiPo battery used for main aircraft/avionics power.

The on-board vision system is composed of a Mobius Action Camera integrated in a Quanum 3 Axis Brushless Gimbal, video link transmitter, RC receiver and a seperate LiPo battery to power the camera gimbal. A Pixhawk Mini flight computer (with separate telemetry radio) is also mounted on top of the Mobius Action Camera to provide accurate gimbal

(a) CONDOR ready for flight



(b) CONDOR gimbal setup

Figure 5.1: CONDOR Unmanned Aerial System (UAS).

tilt angle measurements. This flight computer is only used to provide accurate attitude measurements for the gimbal and is not used as an autopilot. The gimbal setup can be seen in Fig. 5.1(b).

The GCS as shown in Fig. 5.5(b) is contained in a customized 16' trailer and is composed of a main computer with running Mission Planner [3], data link modems, and a video link receiver. CONDOR on-board systems and connections between components are shown in Fig. 5.2. CONDOR ground systems and connections between components are shown in Fig. 5.4. There are also two RC transmitters, one for aircraft manual control and the second one for camera gimbal control. Three operators are required to perform flight: a GCS operator, a camera gimbal operator and a pilot. Due to the current limitations of the vision system discussed in Section 3.2.1, the camera gimbal operator must ensure that the visual anchor point remains within the field of view of the streamed imagery.

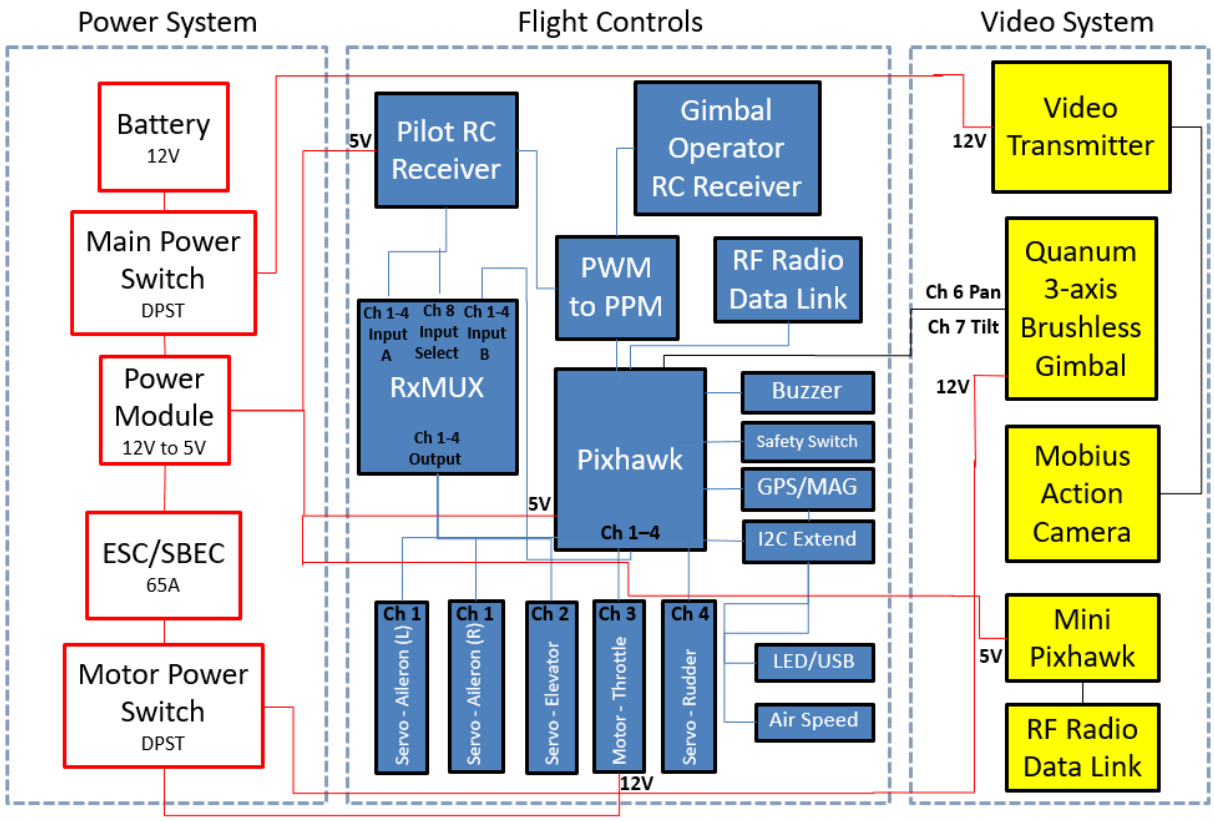


Figure 5.2: Block diagram of CONDOR on-board systems.

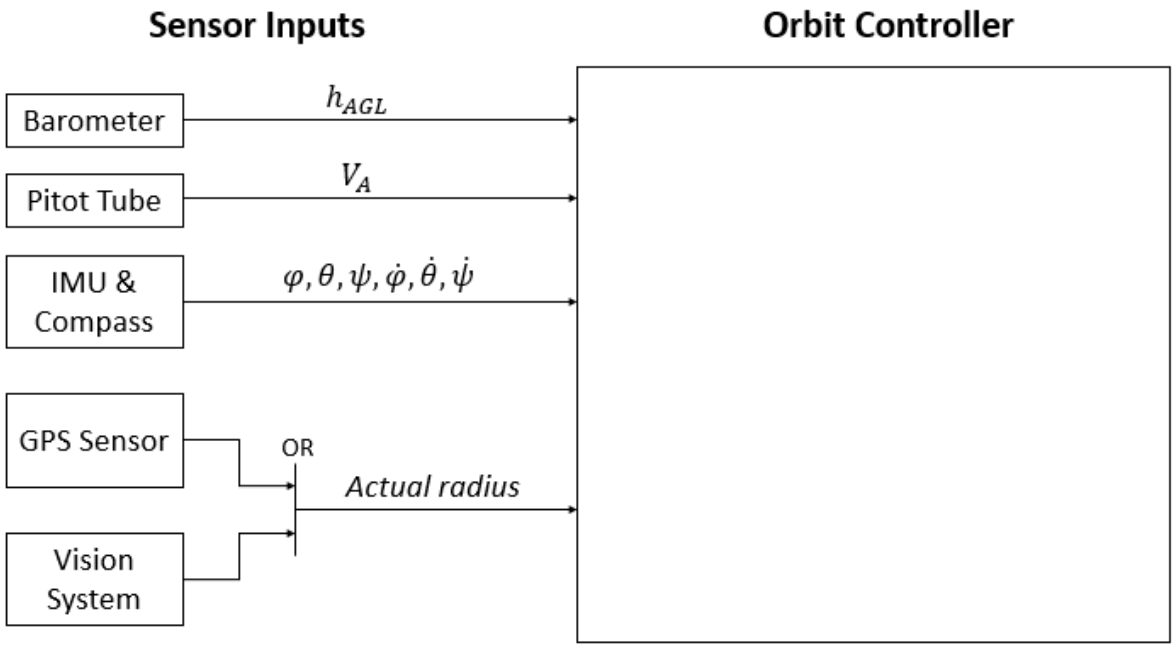


Figure 5.3: Block diagram of Visual Anchoring custom flight mode.

5.2 Arduplane Modification

Arduplane 3.5.3 was forked from the GitHub ArduPilot repository[1]. The orbit controller described in Chapter 2 was converted into a C++ library. The library was added to the Arduplane project and referenced within a custom flight mode. As shown in Fig. 5.3, The Visual Anchoring flight mode provides the inputs required for the orbit controller from the on-board sensors integrated with the Pixhawk. In addition, the flight mode is able to provide the actual radius input from either the on-board GPS sensor or the vision system. This is designed to allow for initial orbit controller testing independent of the vision system. The flight mode also allows for the actual radius input to be provided at variable update frequencies. This is important because the vision system can only provide actual radius updates to the Pixhawk at around 3 Hz due to the telemetry radio link. However, the on-board GPS sensor can provide actual radius updates to the Pixhawk at 50 Hz.

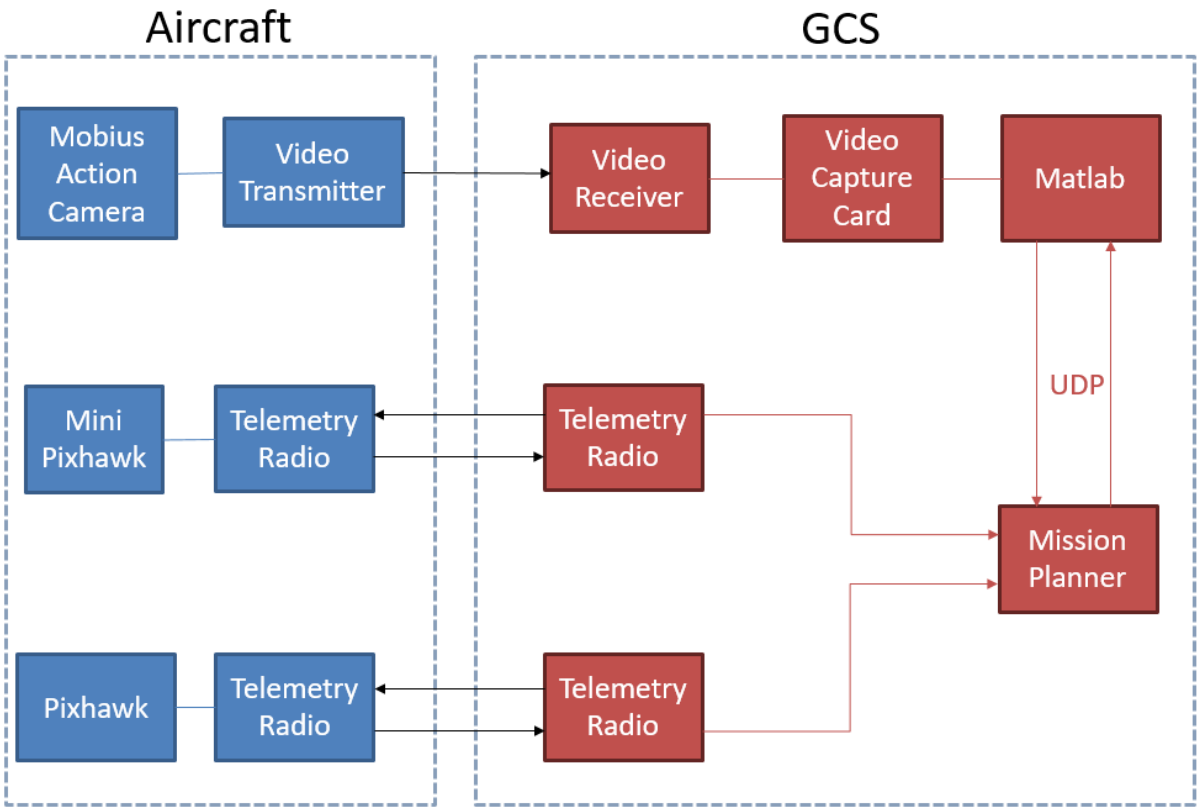


Figure 5.4: Block diagram of CONDOR ground systems.

5.3 Mission Planner Modification

Mission Planner 1.3.38 was forked from the GitHub MissionPlanner repository[2]. A menu was altered in the Mission Planner source code to accommodate the custom flight modes added to Arduplane (see Section 5.2). An embedded Python script was created to bridge the communication between the aircraft firmware (Arduplane), the GCS (Mission Planner), and the vision system software (Matlab). The embedded python script runs in Mission Planner and communicates with Matlab via UDP Port communication. The vision system software combines the tracking algorithm and the slant range algorithm defined in Chapter 3 and is implemented in Matlab. A video capture card is used to digitize the analog video signal from the video reciever so that the digital video signal can be read in Matlab.

(a) Base infrastructure

(b) CONDOR GCS housed inside the UW AFSL Mobile Flight Operations Center (MFOC)

Figure 5.5: Flight testing of the CONDOR UAS near Seattle, WA.

5.4 Flight Testing Infrastructure

The aircraft is registered as N632UW. Early flights were conducted under FAA Certificate of Authorization (COA) 2016-WSA-23-COA at a test site in class G airspace in Washington State. Current flight tests operate under FAA Part 107 rules[17]. Risk assessments[10, 12, 14] are conducted prior to flights to ensure safe operation of the UAS.

Chapter 6

ORBIT CONTROLLER RESULTS

Major results for the orbit controller are presented in this section. They are presented in order of increasing complexity and fidelity. Matlab simulation results serve to validate the orbit controller algorithm. Next, SITL simulation results validate the customized firmware. Finally, flight testing demonstrates viability of the firmware with the actual aircraft.

6.1 Matlab/Simulink Results

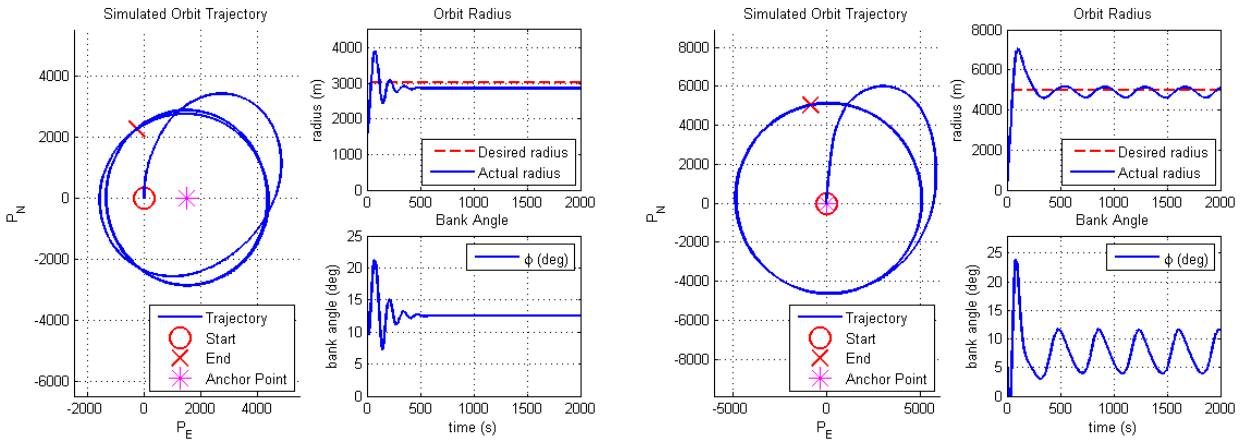
The orbit controller simulation was subject to a set of test scenarios designed to evaluate the performance and robustness of the control algorithm. Table 6.1 summarizes the set of variables being considered to create various flight conditions. Each column in Table 6.1 lists all test conditions for a specific parameter, and each test scenario is a combination of the 4 test condition parameters. The simulation loops through all 81 combinations of test condition parameters and records time history data for each test case. Each test condition consists of a nominal scenario and several extreme scenarios to completely characterize the aircraft performance.

Figure 6.1(a) shows results from the test case where initial position is off orbit by half a radius and with no wind/noise disturbances added. In this case, the aircraft takes approximately 400 s to settle in the desired orbit. Once the orbit was captured, the aircraft maintains the radius while keeping its attitude stabilized.

Figure 6.1(b) shows results from the test case where initial position is off orbit by a full radius and wind disturbances of 10 m/s is added. In this somewhat extreme test scenario, the aircraft is still able to capture the orbit. The oscillatory behavior in the radius is due to the steady wind causing the trajectory to be biased in one direction.

Table 6.1: Matlab/Simulink matrix of test conditions.

Orbit Radius	**Initial Position**	**Wind Condition**	**Vision System Noise**
Small (2000 m)	On Orbit	No Wind	No Noise
Medium (3000 m)	Off Orbit (half radius)	Moderate Wind (5 m/s)	Moderate Noise (mag. = 0.5)
Large (5000 m)	Off Orbit (full radius)	Strong Wind (10 m/s)	Strong Noise (mag. = 1.0)



(a) $R = 3000\,\text{m}$. (b) $R = 5000\,\text{m}$.

Figure 6.1: Simulated Orbit Trajectory.

6.2 SITL Results

The next level of simulation fidelity involves the SITL simulation described previously in Section 4.2. The SITL simulation involved testing the orbit controller while implemented into the Visual Anchoring flight mode. For these tests, the aircraft was commanded to autonomously take off in simulation and enter an orbit about the home waypoint. The aircraft was then commanded into the Visual Anchoring flight mode while given the actual radius input from the GPS Sensor.

Table 6.2: SITL matrix of test conditions.

Radius	**Initial Position**	**Wind Condition**	**Radius Update Frequency**
200 m	Outside Orbit	Wind 10 kt / 000°	50 Hz
200 m	Outside Orbit	Wind 10 kt / 000°	10 Hz
200 m	Outside Orbit	Wind 10 kt / 000°	3 Hz

Scenario variables shown in Table 6.2 were used to generate several flight conditions in the SITL environment to validate that the modified flight control firmware operates as desired. Smaller radii were tested in the SITL environment than the Matlab environment to ensure that smaller radii orbits could be stabilized. This is important when implementing the vision system because the tracking algorithm will not perform well on very small targets. Reducing the radius of the orbit will allow for targets to appear larger in the image for better tracking.

Figure 6.2(a) shows the performance of the orbit controller at a 200 m radius and a radius update frequency of 50 Hz. The Visual Anchoring flight mode was initialized slightly outside the desired orbit trajectory with a steady 10 kt wind from the north (000°). The orbit controller was able to successfully command the aircraft to a stable orbit in significant wind with smooth bank commands.

Figure 6.2(b) and Figure 6.2(c) show the performance of the orbit controller at a 200 m radius with a 10 Hz radius update frequency and a 3 Hz radius update frequency respectively.The Visual Anchoring flight mode was initialized slightly outside the desired orbit trajectory with a steady 10 kt wind from the north for both cases. Lowering the position update speed had a slight impact on the actuated orbit, but the main impact was on the smoothness of the bank commands. The bank commands became more sharp and noisy with less frequent radius information, but the orbit result was still sufficient.

(a) Simulation of VA mode at 50 Hz.

(b) Simulation of VA mode at 10 Hz.

(c) Simulation of VA mode at 3 Hz.

Figure 6.2: Simulated SITL results with various radius update frequencies showing convergence to a stable orbit centered on a waypoint.

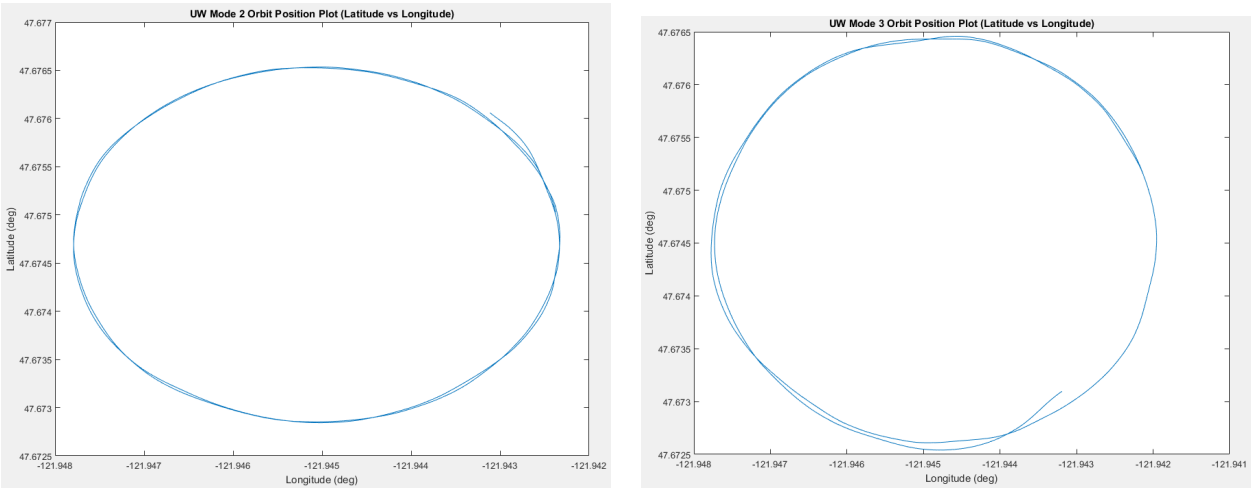
6.3 Flight Testing Results

The Visual Anchoring flight mode was tested at both 50 Hz and approximately 3 Hz radius update frequencies using the GPS sensor for the actual radius inputs. The reason the radius update frequency was approximately 3 Hz is because the actual radius input was limited by the telemetry radio link between Mission Planner and the Pixhawk. In simulation, the radius update frequency for the 3 Hz test is manually set to exactly 3 Hz to simulate the telemetry radio link. The custom firmware used for flight testing was identical to the custom firmware used in the SITL testing. The same procedure as defined in the SITL results section was used to prepare the aircraft to enter the custom flight mode.

Figure 6.3(a) shows the performance of the orbit controller at a 200 m radius and a radius update frequency of 50 Hz. The Visual Anchoring flight mode was initialized slightly outside the desired orbit trajectory with around a 6 kt wind from the southeast. The orbit controller was able to successfully command the aircraft to a stable orbit in significant wind with smooth bank commands as expected from the SITL simulation.

Figure 6.3(b) shows the performance of orbit controller at a 200 m radius and a radius update frequency of approximately 3 Hz. The Visual Anchoring flight mode was initialized slightly inside the desired orbit trajectory with around a 4 kt wind from the northeast. The orbit controller was able to successfully command the aircraft to a stable orbit in significant wind with relatively smooth bank commands. The affect of lower radius update frequency can be seen in the actuated orbit. Small perturbations from the dynamic wind environment took longer to decay than with the 50 Hz radius update frequency.

Note that these flight tests used the GPS sensor to provide the actual radius input for the orbit controller. The vision system was not being used to estimate the radius to the target.



(a) Flight Test of VA mode at 50 Hz

(b) Flight Test of VA mode at 3 Hz

Figure 6.3: CONDOR aircraft converging to a stabilized orbit in the VA flight mode (GPS).

Chapter 7
VISION SYSTEM RESULTS

Major results for the vision system are preseneted in this section. Initially, the vision system was ground tested to ensure the Slant Range Algorithm was estimating ground distances properly. Next, the vision system was flight tested on the CONDOR aircraft to ensure that the Slant Range Algorithm worked properly at higher altitudes and the tracking algorithm was able to successfully track while in flight.

7.1 Ground Testing Results

The Slant Range Algorithm described in Chapter 3 was initially tested by performing a ground test of the vision system. The CONDOR aircraft was kept stationary at a relative altitude of $h_{AGL} = 1.041m$ while the gimbal was held at a tilt angle of $\theta_C = -44.5°$. A grid was placed in front of the camera and distances between the aircraft and the grid intersection points were known. The X and Y ground position coordinates of the grid were visually estimated using the vision system without a nonlinear correction factor ($N = 1$). The coordinates were then visually estimated again using a nonlinear correction factor of $N = 1.3$. Figure 7.1 shows a comparison of the visually estimated and the actual coordinates from this ground test. The CONDOR aircraft was located 1.041 meters above the origin of the plot shown in Figure 7.1. The vision system consistently overestimated the Y axis distance estimations for all the grid points when using a nonlinear correction factor of $N = 1$. However, the vision system estimated the X axis distance estimations for all the grid points much better. This suggests that the pixel to field of view ratio in the X axis direction of the image is much closer to linear than the Y axis direction. The average error between the actual and estimated grid points for the $N = 1$ case was 6.3% or 81.5 mm. The average error

between the actual and estimated gird points for the $N = 1.3$ case was 3.1% or 43.5 mm.

7.2 Flight Testing Results

In order to flight test the vision system, the CONDOR aircraft was set into an autonomous orbit about a visual anchor point using the ArduPlane guided mode. A large blue tent was used as the visual anchor point and placed in the center of a grass field. The blue color of the tent was very distinct from the green grass field and allowed for successful tracking. The ArduPlane guided mode allows for the user to select a GPS waypoint in Mission Planner and the aircraft will begin to orbit around that point at a desired altitude and radius. This mode is particularly useful for flight testing the vision system because the GPS measured radius of the orbit is recorded and can be compared to the visually estimated radius calculated by the vision system. Figure 7.2 shows the results of the vision system when implemented in guided mode at a desired altitude of 100 m and a desired radius of 100 m. As seen in Figure 7.2, when a nonlinear correction factor of $N = 1$ was used to estimate the orbit radius, the vision system significantly overestimated the actual orbit radius. However, when a nonlinear correction factor of $N = 5$ was used to estimate the orbit radius, the vision system estimated the actual orbit radius much more accurately. The maximum and average error for the radius estimations for $N = 1$ was 76.30% and 54.91% respectively. The maximum and average error for the radius estimations for $N = 5$ was 12.83% and 5.94% respectively.

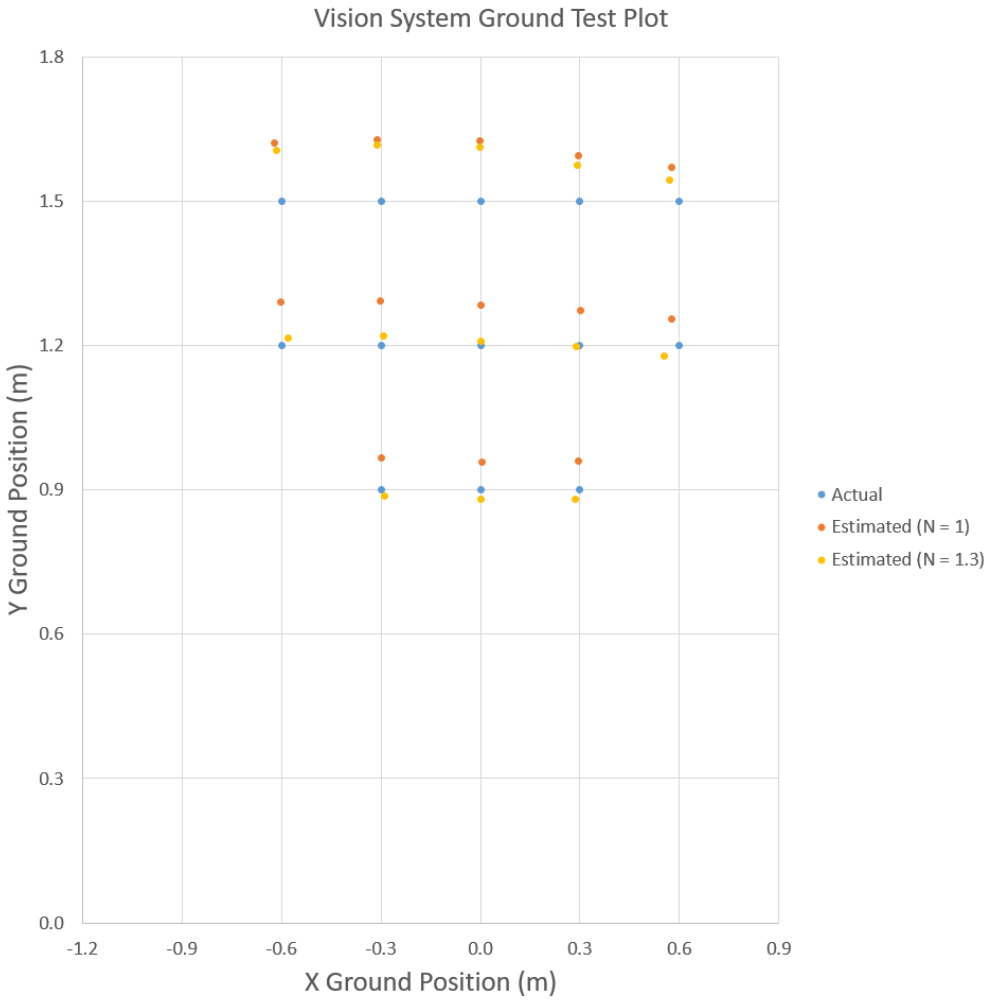


Figure 7.1: Comparison of visually estimated and actual grid coordinates from vision system ground test.

Figure 7.2: Comparison of visually estimated and GPS measured orbit radius from vision system flight test.

Chapter 8
CONCLUSIONS

This paper describes a method to stabilize an orbit about a target using vision as the primary sensor modality, thereby making this type of operation feasible for UAS in a GPS-denied environment. It also provides simulation and flight test results of the proposed visual anchoring orbit controller with GPS radius estimates, while demonstrating that the GPS radius estimates can be replaced with the proposed vision system radius estimates. These results demonstrate the ability of the orbit controller to successfully stabilize an orbit using GPS radius estimates at varying radius update frequencies and in real wind conditions. The ability of the orbit controller to function at update frequencies as low as 3 Hz allows for the vision system to be implemenented offboard of the aircraft and provided through the telemetry radio link. Ground and flight test results of the proposed vision system were provided to estimate the accuracy of the vision system. These initial results suggest that the vision system can estimate the ground radius between an aircraft and a target at an altitude of 100 meters with an accuracy of around 10% of the actual ground radius. Future work in this area will focus on obtaining flight test results of the combined orbit controller and vision system to demonstrate the entire visual anchoring flight mode. Once the merged flight test results have been obtained, the vision system code running on the GCS should be moved on-board the aircraft. This would allow for better orbit controller performance because the input actual radius would be able to be provided to the orbit controller at a greater frequency than 3 Hz. In addition, future work should focus on developing a general forumulation to calculate the required nonlinear correction factor depending on aircraft altitude and gimbal tilt angle.

BIBLIOGRAPHY

[1] Github ardupilot repository. GitHub Repository. https://github.com/ArduPilot/ardupilot.

[2] Github missionplanner repository. GitHub Repository, . https://github.com/ArduPilot/MissionPlanner.

[3] Mission planner overview. ArduPilot http://ardupilot.org/planner/docs/mission-planner-overview.html/, . Accessed 2016.

[4] Setting up sitl on windows. ArduPilot Dev Team, 2016. http://ardupilot.org/dev/docs/sitl-native-on-windows.html.

[5] M. Blosch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision based mav navigation in unknown and unstructured environments. In *Proceedings of the 2010 ICRA Conference*, 2010.

[6] J. Brandon. Gps jammers illegal, dangerous, and very easy to buy. *Fox News Technology*, 2010.

[7] D. J. Klein. *Coordinated Control and Estimation for Multi-agent Systems: Theory and Practice*. PhD thesis, University of Washington, Seattle, WA, September 2008.

[8] P. Lambrechts, S. Bennani, G. Looye, and A. Helmersson. Robust flight control design challenge problem formulation and manual: the reserach civil aircraft model (rcam). Technical report, Group for Aeronautical Research and Technology in Europe, Europe, 1997.

[9] C. W. Lum and R. T. Rysdyk. Feature extraction of low dimensional sensor returns for autonomous target identification. In *Proceedings of the 2008 Guidance, Navigation, and Control Conference*, Honolulu, HI, August 2008.

[10] C. W. Lum and D. A. Tsukada. Uas reliability and risk analysis. *Encyclopedia of Aerospace Engineering*, July 2016.

[11] C. W. Lum and J. Vagners. A modular algorithm for exhaustive map searching using occupancy based maps. In *Proceedings of the 2009 Infotech@Aerospace Conference*, Seattle, WA, April 2009.

[12] C. W. Lum and B. Waggoner. A risk based paradigm and model for unmanned aerial vehicles in the national airspace. In *Proceedings of the 2011 Infotech@Aerospace Conference*, St. Louis, MO, March 2011.

[13] C. W. Lum, J. Vagners, and R. T. Rysdyk. Search algorithm for teams of heterogeneous agents with coverage guarantees. *AIAA Journal of Aerospace Computing, Information, and Communication*, 7:1–31, January 2010.

[14] C. W. Lum, K. Gauksheim, T. Kosel, and T. McGeer. Assessing and estimating risk of operating unmanned aerial systems in populated areas. In *Proceedings of the 2011 AIAA Aviation Technology, Integration, and Operations Conference*, Virginia Beach, VA, September 2011.

[15] C. W. Lum, A. Summers, B. Carpenter, A. Rodriguez, and M. Dunbabin. Automatic wildfire detection and simulation using optical information from unmanned aerial systems. In *Proceedings of the 2015 SAE Aerotec Conference*, Seattle, WA, September 2015.

[16] C. W. Lum, M. Mackenzie, C. Shaw-Feather, E. Luker, and M. Dunbabin. Multispectral imaging and elevation mapping from an unmanned aerial system for precision agricul-

ture applications. In *Proceedings of the 13th International Conference on Precision Agriculture*, St. Louis, MO, August 2016.

[17] C. W. Lum, R. S. Larson, W. Handley, S. Lui, and Z. Caratao. Flight testing an ads-b equipped suas in gps-denied environments. In *Proceedings of the AIAA Flight Testing Conference*, Denver, CO, June 2017.

[18] R. T. Rysdyk. Unmanned aerial vehicle path following for target observation in wind. *Journal of Guidance, Control, and Dynamics*, pages 1092–1100, September 2006.

[19] R. T. Rysdyk, C. W. Lum, and J. Vagners. Autonomous orbit coordination for two unmanned aerial vehicles. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, San Francisco, CA, August 2005.

[20] B. Sinopoli, M. Micheli, G. Donato, and T.-J. Koo. Vision based navigation for an unmanned aerial vehicle. In *Proceedings of the 2001 ICRA Conference*, 2001.

[21] M. Stachura and E. W. Frew. Cooperative target localization with a communication-aware unmanned aircraft system. *Guidance, Control, and Dynamics*, 34:1352–1362, 2011.

[22] B. L. Stevens and F. L. Lewis. *Aircraft Control and Simulation*. John Wiley and Sons, Hoboken, NJ, 2nd edition, 2003.

[23] K. K. Ueunten, C. W. Lum, A. A. Creigh, and K. Tsujita. Conservative algorithms for automated collision awareness for multiple unmanned aerial systems. In *Proceedings of the 2015 IEEE Aerospace Conference*, Big Sky, MO, March 2015.

[24] J. A. Volpe. Vulnerability assessment of the transportation infrastructure relying on the global positioning system. Technical report, National Transporation Systems Center, 2001.

[25] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *Proceedings of the 2012 ECCV Conference*, Florence, Italy, October 2012. Part III, LNCS 7574, pp. 866879, 2012.