AIAA SciTech Forum
8–12 January 2018, Kissimmee, Florida
2018 AIAA Information Systems-AIAA Infotech @ Aerospace

10.2514/6.2018-0506

Visual Anchoring: Orbiting a Target with a UAS Using Vision as the Primary Sensor Modality

Christopher W. Lum*, Ryan Grimes†, Dai Tsukada‡, and Jonathon Winde§
Autonomous Flight Systems Laboratory
University of Washington, Seattle, WA, 98195, USA

Tadej Kosel¶
Laboratory of Aeronautics
Faculty of Mechanical Engineering, University of Ljubljana, Slovenia

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.

Nomenclature

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

*Research Assistant Professor, William E. Boeing Department of Aeronautics and Astronautics, University of Washington; Guggenheim Hall Room 211, Box 352400, Seattle, WA 98195-2400, AIAA member.
†Researcher Assistant, William E. Boeing Department of Aeronautics and Astronautics, University of Washington; Guggenheim Hall Room 211, Box 352400, Seattle, WA 98195-2400.
‡Formerly Research Assistant at UW, now Autonomous Control Systems Laboratory, Ltd., Marive West 32F, 2-6-1, Nakase, Mihama-ku, Chiba, 261-7132, Japan.
§Formerly Research Assistant at UW, now Insitu, 118 E Columbia River Way, Cook, WA 98605, USA.
¶Research Associate Professor, Faculty of Mechanical Engineering, University of Ljubljana, Askerceva 6, SI-1000 Ljubljana, Slovenia, EU

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_ψ}	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
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

I. Introduction

I.A. 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. 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.

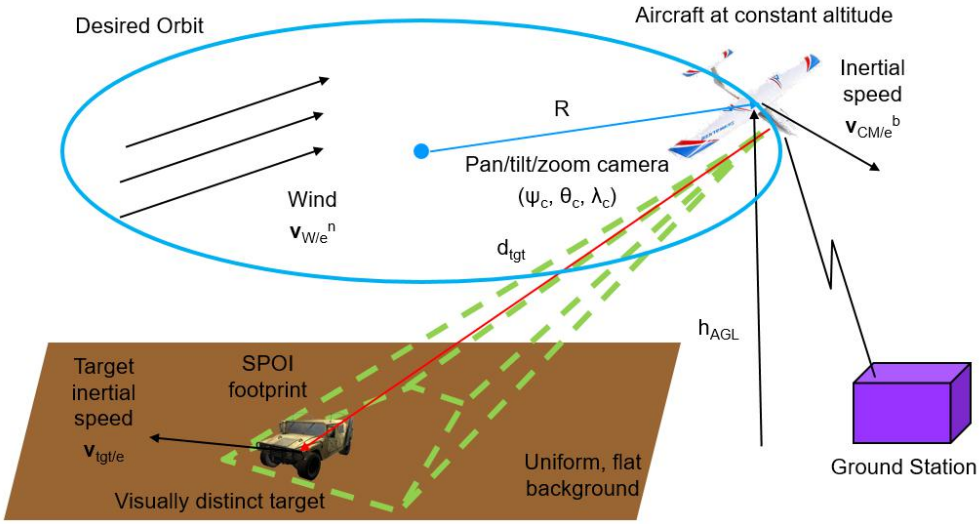


Figure 1. Project vision showing a UAS orbiting a target in a wind field while streaming imagery to the GCS.

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.[1] 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[2].

I.B. Prior Work

The problem of establishing an orbit[3] or otherwise tracking a moving target in wind[4] has been studied before[5]. 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 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[6,7]. Others have looked at localizing targets[8] or finding unique features such as wildfires[9] using UAS. Recently, image processing from UAS has become popular in the realm of precision agriculture[10], collision avoidance[11], and searching[12–14].

II. Controller Description

The overall block diagram of the system is shown in Figure 2. The control law for establishing an orbit is executed onboard the aircraft. However, inputs to the controller are computed on the GCS via the vision algorithm (described in Section III). Further details on the UAS and associated hardware is discussed in Section V.A.

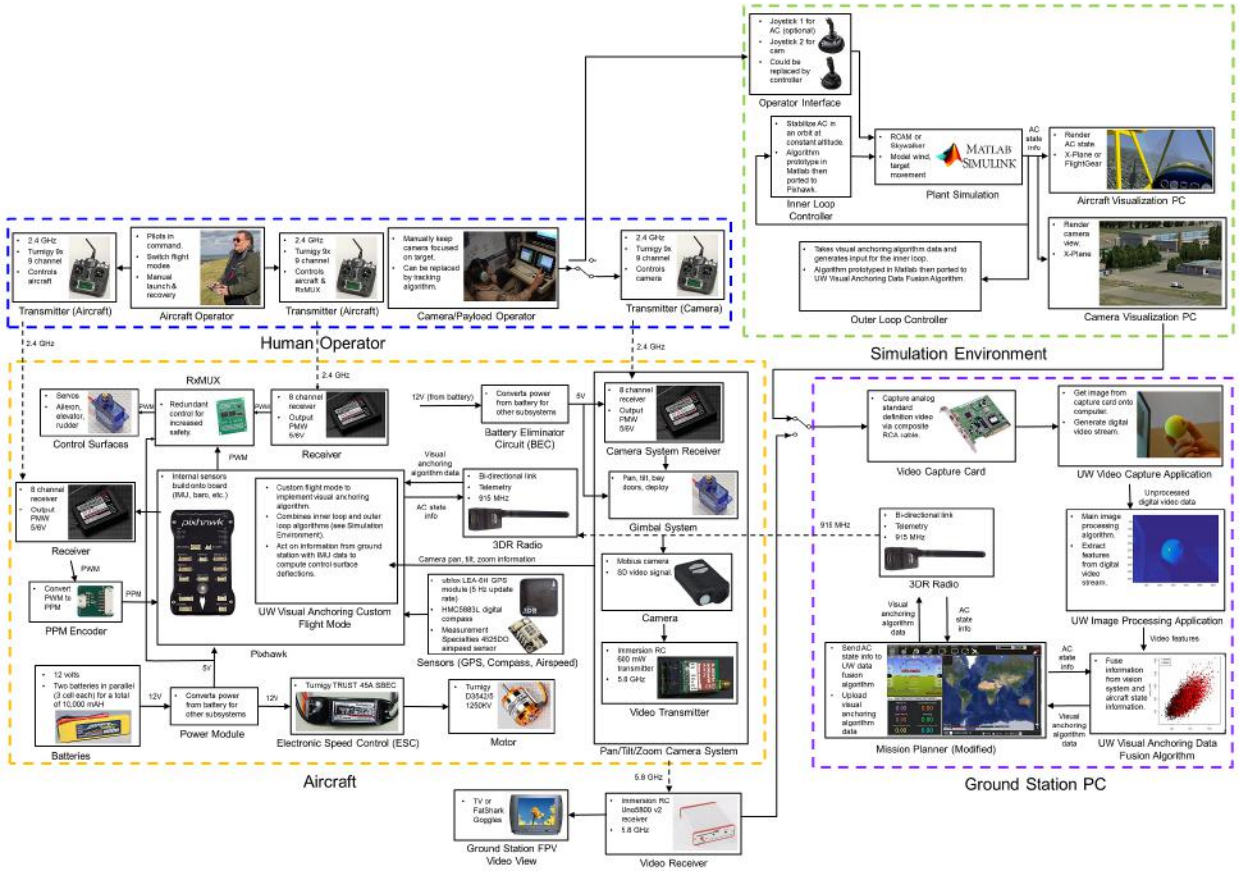


Figure 2. Block diagram of entire system including aircraft, GCS, and simulation environment.

II.A. Control Law

The architecture of the orbit controller is shown in Figure 3. 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 aileron, elevator and rudder to perform a steady state coordinated turn. The typical approach for such control problem is to decouple the lateral and directional axes and design controllers to command appropriate control inputs[15].

II.A.1. Coordinated Turn

In an ideal coordinated turn, net forces in the body y-axis direction are zero[15]. 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

$$\tan\phi = \frac{W}{F_{center}} \tag{1}$$

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

$$V_A = \dot{\psi}\, R \tag{2}$$

$$F_{center} = \frac{m\, V_A^2}{R} = m\, V_A\, \dot{\psi} \tag{3}$$

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

$$\phi = \arctan\frac{V_A\, \dot{\psi}}{g} \tag{4}$$

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}.

$$\phi_{ref} = \arctan\frac{V_A\, \dot{\psi}_{ref}}{g} \tag{5}$$

The inner loop controller uses a reference heading rate, $\dot{\psi}_{ref}$, with respect to 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}.

II.A.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 ground radius information obtained by the vision system (Section III). 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

$$\dot{\psi}_{err} = -(K_{P_{outer}}\,(R - R_{des}) - K_{D_{outer}}\dot{R}) \tag{6}$$

II.A.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 attitudes 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 yaw damper 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 the altitude above ground, h_{AGL}, of the aircraft specified by the operator.

Roll Inner Loop The roll inner loop controller is responsible of producing the aileron command component to satisfy the turn coordination constraint. The combination of the roll and roll rate feedback control helps to achieve 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).

$$\dot{\psi}_{ref} = \dot{\psi}_{err} + \frac{V_A}{R} \tag{7}$$

Bank angle error, ϕ_{err}, is derived from reference bank angle, ϕ_{ref}, from Eq. (5) and measured bank angle, ϕ.

$$\phi_{err} = \arctan\left(\frac{V_A\,\dot{\psi}_{ref}}{g}\right) - \phi \tag{8}$$

The control algorithm to compute aileron deflection, ΔA, is a typical proportional-derivative scheme that consumes bank angle error, ϕ_{err}, from Eq. (8) and bank rate, $\dot{\phi}$, from the inertial measurement.

$$\Delta A = -\left(K_{P_\phi}\,\phi_{err} - K_{D_\phi}\,\dot{\phi}\right) \tag{9}$$

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.

$$\Delta R = K_{P_{\dot{\psi}}}(\dot{\psi}_{ref} - \dot{\psi}) \tag{10}$$

Altitude Hold The altitude hold controller serves to hold a constant orbit altitude specified by the operator regardless of orbit radius and flight conditions. The controller first computes the reference pitch angle, θ_{ref}, and it is then consumed by the pitch inner loop where the elevator surface deflection, ΔE, is calculated.

$$\theta_{ref} = K_{P_h}\,(h_{des} - h_{AGL}) + K_{I_h}\int_0^t (h_{des} - h_{AGL})\,dt \tag{11}$$

$$\Delta E = -\left(K_{P_\theta}\,(\theta_{ref} - \theta) - K_{D_\theta}\dot{\theta}\right) \tag{12}$$

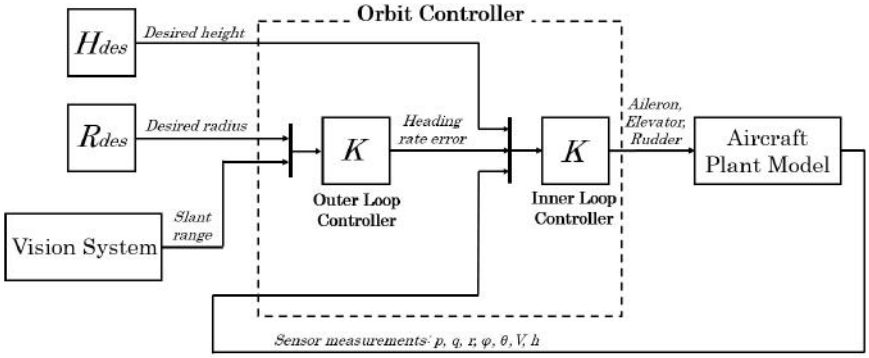


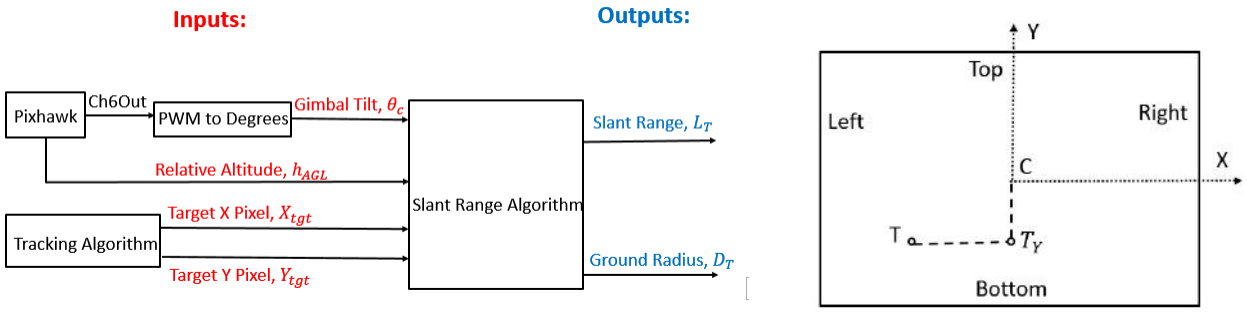
Figure 3. Orbit controller block diagram.

III. Vision Algorithm

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 controllers as the input actual radius for the system. This section describes the algorithm that the vision system utilizes to perform these functions.

III.A. Computing Ground Distance between Aircraft and Target

As seen in Fig. 4(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 III.B.



(a) Block diagram of the slant range algorithm.

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

Figure 4. Computing slant range from the image frame.

The image frame can be visualized in Fig. 4(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.

III.A.1. Y-Axis Geometry

Through inspection of the Y-Axis physical geometry, seen in Fig. 5(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.

$$L_C = \frac{h_{AGL}}{\tan(\theta_C)} \tag{13}$$

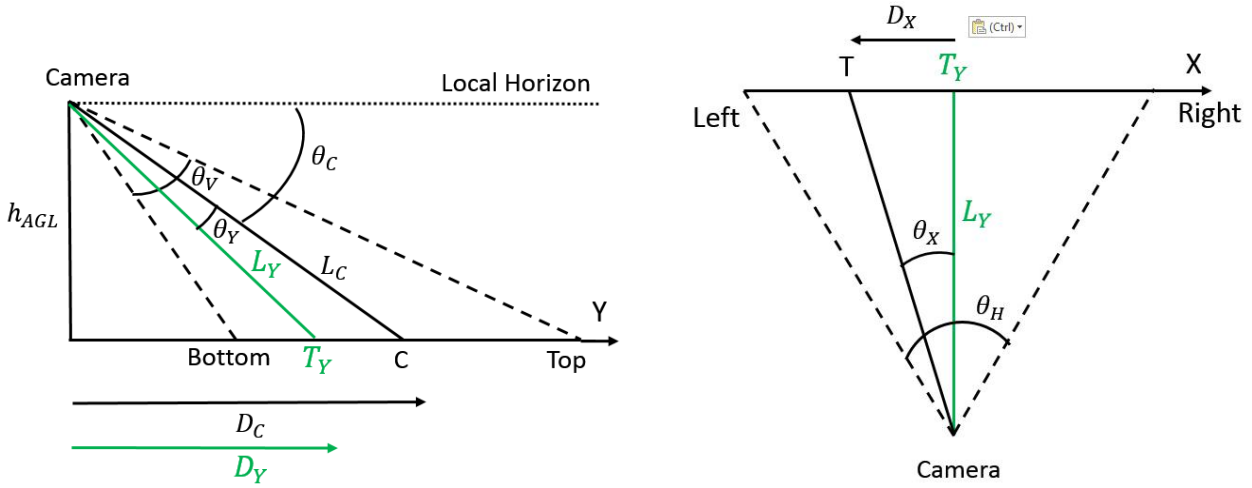
$$D_C = \frac{h_{AGL}}{\sin(\theta_C)} \tag{14}$$

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.

$$L_Y = \frac{h_{AGL}}{\sin(\theta_C + \theta_Y)} \tag{15}$$

$$D_Y = \frac{h_{AGL}}{\tan(\theta_C + \theta_Y)} \tag{16}$$

$$\theta_Y = \theta_V \frac{Y_{tgt}}{S_y} \tag{17}$$



(a) Y-axis geometry.

(b) X-axis geometry.

Figure 5. Physical geometry associated with x and y axis.

III.A.2. X-Axis Geometry

Through inspection of the X-Axis physical geometry, seen in Fig. 5(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 value of interest, the absolute value of θ_X is used to ground distance values positive.

$$D_X = L_Y \tan(|\theta_X|) \tag{18}$$

$$\theta_X = \theta_H \frac{X_{tgt}}{S_x} \tag{19}$$

III.A.3. Combined Axes Geometry

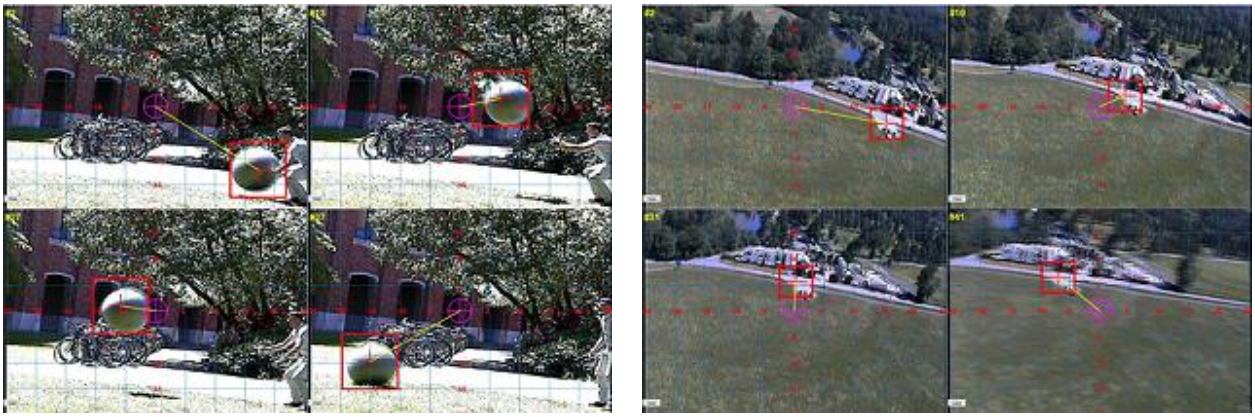
By combining the information from the X and Y axes, the total slant range between the aircraft and the target, L_T, and the total ground radius between the aircraft and the target, D_T, can be found in the following equations.

$$D_T = \sqrt{D_X^2 + D_Y^2} \tag{20}$$

$$L_T = \sqrt{D_T^2 + h_{AGL}^2} \tag{21}$$

III.B. Tracking Algorithm

The vision system utilizes a real-time compression tracking scheme[16] 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 object tracking algorithm is very useful for the application of visual anchoring. Tests were performed both on the ground and in the air. On the ground a large exercise



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

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

Figure 6. Tracking an object

ball was filmed and the algorithm worked very well as shown in Fig. 6(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. 7(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 6(b) shows example of tracking the parking lot where the ground crew was located during a flight test. Figure 6(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.

III.B.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.

IV. 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. Different environments were used to simulate these two components separately from one another.

IV.A. Orbit Controller Simulation

The orbit controller is a simple inner/outer loop control scheme and as such, there are many tools available to simulate this system in order to validate performance.

IV.A.1. Matlab/Simulink Simulation

The primary controller design took place in Matlab/Simulink. The popular Research Civil Aircraft Model (RCAM)[17] 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 slant range measured from the anchor point defined by the user. Actual simulation results of this system are discussed in Section VI.A.

IV.A.2. JSBSim, Mission Planner, Arduplane Simulation

Once the orbit control algorithm was validated in the Matlab/Simulink environment, the algorithm was ported onto the Arduplane firmware so it can run on a UAS. The Arduplane implementation of the flight controller was primarily tested in a Software-in-the-Loop (SITL) simulated environment[18]. 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 V.A.1). A modified version of Mission Planner (Section V.A.2) was used as the ground control station (GCS) to communicate and send commands to the simulator. Results of this simulation are discussed in Section VI.B.

IV.B. Image Processing Simulation

Simulating and testing the image processing portion of this system in a laboratory environment required an environment capable of interacting with a dynamic aircraft simulation while simultaneously rendering the physical environment in a realistic fashion. The popular X-Plane[19] flight simulator environment was used but only as an engine to rendering scenery. A series of plugins[20] was used to override the internal dynamic aircraft model in X-Plane and instead use data from the Matlab/Simulink simulation as the state of the aircraft. In addition, the Matlab/Simulink environment contains systems to simulate a gimbaled camera. A user is able to manipulate the gimbal with joystick and the X-Plane environment is used to render the viewpoint of the camera on a separate computer (referred to as the 'Camera Visualization PC' in Figure 2). This video can be streamed to the image processing algorithm in lieu of the actual imagery from a UAS video feed.

V. Flight Test Environment

Once sufficient simulation and analysis was completed, the algorithms were implemented on aircraft hardware for flight testing.

V.A. 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. 7(a). The airframe is upgraded with systems for autonomous flight and video transfer. The control surfaces consist of ailerons (no flaps), rudder and split elevator. A split elevator is used for safety purposes (redundant servos). An additional safety feature is an RxMUX unit (8-Channel Servo Multiplexer) device which protects the airplane in case of total flight computer failure. It allows the pilot to take manual control of the airplane during flight and has direct connection between RC receiver and 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 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 LiPo battery used for main aircraft/avionics power. The camera gimbal system is composed of a camera integrated in an off-the-shelf gimbal system, video link transmitter, RC receiver and a seperate LiPo battery to power the camera gimbal. The GCS as shown in Fig. 7(b) is contained in a customized 16' trailer and is composed of a main computer with running Mission Planner[21], data link modem, and video link receiver. CONDOR on-board, ground systems, and connections between components are shown in Fig. 8. There are also two RC transmitters, one for aircraft manual control and the second one for camera gimbal control. Three people are required to perform flight; a GCS operator, a camera gimbal operator and a pilot.

(a) CONDOR in flight with retracted camera gimbal

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

Figure 7. CONDOR unmanned aerial system.

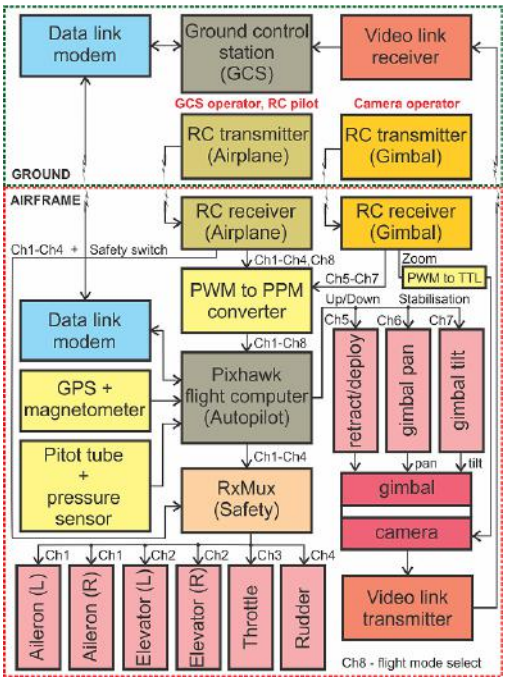


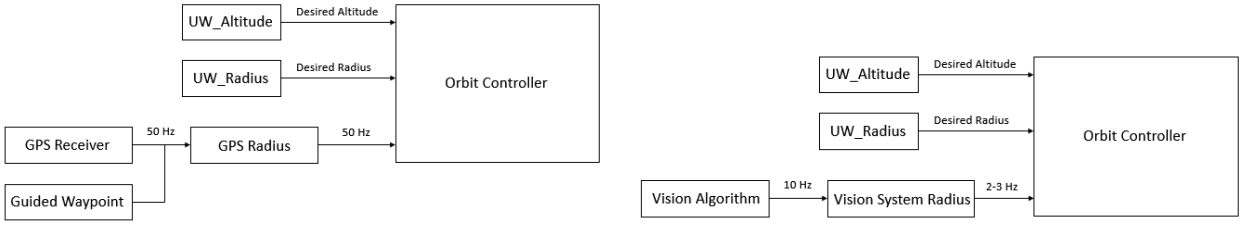
Figure 8. Block diagram of CONDOR flight and video systems.

V.A.1. Arduplane Modification

Arduplane 3.5.3 was forked from the GitHub ArduPilot repository[22]. The flight controller described in Section II was converted from the Simulink model and ported to a C++ library. The library was added to the Arduplane project and referenced within two custom flight modes, UW_Mode_2 and UW_Mode_3. These two custom flight modes are essentially identical, except for the methods they use to receive the actual radius measurements for the orbit controller. When activated on the Pixhawk, both flight modes operate at 50 Hz. Two custom parameters, UW_Altitude and UW_Radius were also added to the Arduplane project to provide the orbit controller with desired altitude and radius values.

UW_Mode_2: As seen in Fig. 9(a), UW_Mode_2 is the initial testing mode for the orbit controller that does not allow for use of the vision system radius estimates. This mode is designed to allow for testing of the orbit controller with only GPS radius estimates. This allows for testing of the orbit controller's performance without the influence of the vision system. The distance between the current GPS location of the aircraft and the GPS location of the target is used as the input actual radius for the orbit controller. The GPS location of the target is provided by the user when selecting a desired point of orbit for guided mode. Guided mode is a flight mode in Arduplane that computes control inputs that cause the aircraft to orbit around a selected orbit point (defined by a GPS waypoint). The position update speed for this mode is 50 Hz; a limitation of the GPS receiver onboard the aircraft. This means that the input actual radius information used by the orbit controller is updated at 50 Hz.

UW_Mode_3: As seen in Fig. 9(b), UW_Mode_3 is designed to be the demonstration mode for the visual anchoring method. The input actual radius measurements for this mode are obtained over the Pixhawk's RC Channel 8. The embedded python script in Mission Planner can either send GPS radius estimates or vision system radius estimates over RC Channel 8. If the script is sending vision system radius estimates, Mission Planner receieves the estimates from the vision system code implemented in Matlab via UDP communication. If the script is sending GPS radius estimates, Mission Planner receives the estimates from the telemetry radio link. Therefore, this mode does not inherently rely on the GPS receiver for input actual radius measurements, but it can be tested with GPS radius estimates to isolate the orbit controller from the vision system. This is useful when comparing the performance of UW_Mode_2 to UW_Mode_3 in order to gauge the impact of position update speed on orbit controller performance. The position update speed for this mode in simulation is variable between 1 and 10 Hz. In SITL simulation, the python script is only able to send GPS radius estimates because the vision system is not simulated in the environment. Since the telemetry radio link does not exist physically in the simulation, the position update speed can be varied by settings in Mission Planner. The position update speed for this mode in flight is 3 Hz; a limitation of the telemetry radio link between the aircraft and the GCS.



(a) UW_Mode_2. (b) UW_Mode_3.

Figure 9. Block diagrams of custom Arduplane flight modes.

V.A.2. Mission Planner Modification

Mission Planner 1.3.38 was forked from the GitHub MissionPlanner repository[23]. A menu was altered in the Mission Planner source code to accommodate the custom flight modes added to Arduplane (see Section V.A.1). An embedded Python script was created to bridge the communication between the aircraft firmware (Arduplane), the GCS (Mission Planner), and the image processing software (Matlab). The embedded python script runs in Mission Planner and communicates with Matlab via UDP Port communication. The image processing software combines the tracking algorithm and the slant range algorithm defined in Section III and is implemented in Matlab.

Table 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)

V.B. Flight Testing Infrastructure

The aircraft is registered as N589WC. 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.[24] Risk assessments[25–27] are conducted prior to flights to ensure safe operation of the UAS.

(a) Base infrastructure.

(b) Performing a hand launch of the CONDOR UAV.

Figure 10. Flight testing of the CONDOR UAV near Seattle, WA.

VI. Results

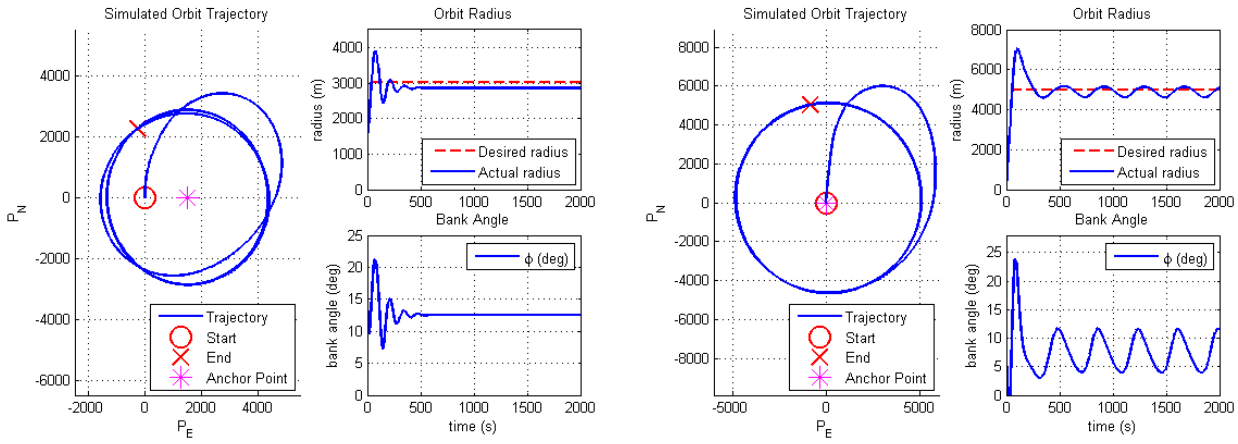
Major results are presented in this section. These are presented in an order or increasing complexity and fidelity. Matlab simulation results serve to validate the orbit controller algorithms. Later, SITL simulation results validate the customized firmware. Finally, flight testing demonstrates viability of the system with actual aircraft.

VI.A. 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 1 summarizes the set of variables being considered to create various flight conditions. Each column in Table 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 a several extreme scenarios to completely characterize the aircraft performance.

Figure 11(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 11(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 attribute to the steady wind causing the trajectory to be biased in one direction.



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

Figure 11. Simulated Orbit Trajectory.

Table 2. SITL matrix of test conditions.

Mode	Radius	Initial Position	Wind Condition	Position Update Speed
UW_Mode_2	200 m	Outside Orbit	Wind 10 kt / 000°	50 Hz
UW_Mode_3	200 m	Outside Orbit	Wind 10 kt / 000°	10 Hz
UW_Mode_3	200 m	Outside Orbit	Wind 10 kt / 000°	3 Hz

VI.B. SITL Results

The next level of simulation fidelity involves the SITL simulation described previously in Section IV.A.2. The SITL simulation involved testing the orbit controller while implemented into the Arduplane custom flight modes UW_Mode_2 and UW_Mode_3. 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 UW_Mode_2 or UW_Mode_3 with given GPS relative position between the aircraft and the waypoint. Since the vision system will be tested separately, the SITL simulation uses GPS information to validate the orbit controller. UW_Mode_2 receives position updates at 50 Hz, while UW_Mode_3 allows for a variable rate position update speed that can be changed to mirror what the actual flight test position update speed will be. This flight test position update speed is limited by the telemetry radio used to transmit information between the Pixhawk and the GCS.

Scenario variables shown in Table 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 frame for easier tracking.

Figure 12(a) shows the performance of UW_Mode_2 at a 200 m radius. UW_Mode_2 was initialized slightly outside the desired orbit trajectory with a steady 10 kt of 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 12(b) and Figure 12(c) show the performance of UW_Mode_3 at a 200 m radius with 10 Hz position update speed and 3 Hz position update speed respectively. UW_Mode_3 was initialized slightly outside the desired orbit trajectory with a steady 10 kt of 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) UW_Mode_2 at 50 Hz. (b) UW_Mode_3 at 10 Hz. (c) UW_Mode_3 at 3 Hz.

Figure 12. Simulated SITL results with various controllers and scenarios showing convergence to a stable orbit centered on a waypoint.

VI.C. Flight Testing Results

UW_Mode_2 and UW_Mode_3 were tested separately in flight using the firmware that was simulated in the SITL testing. UW_Mode_2 still received position updates at 50 Hz, while UW_Mode_3 was limited to around 3 Hz of position update speeds from the telemetry radio link. The same procedure as defined in the SITL results section was used to prepare the aircraft to enter the custom modes.

Figure 13(a) shows the performance of UW_Mode_2 at a 200 m radius. UW_Mode_2 was initialized slightly outside the desired orbit trajectory with around 6 kt of 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.

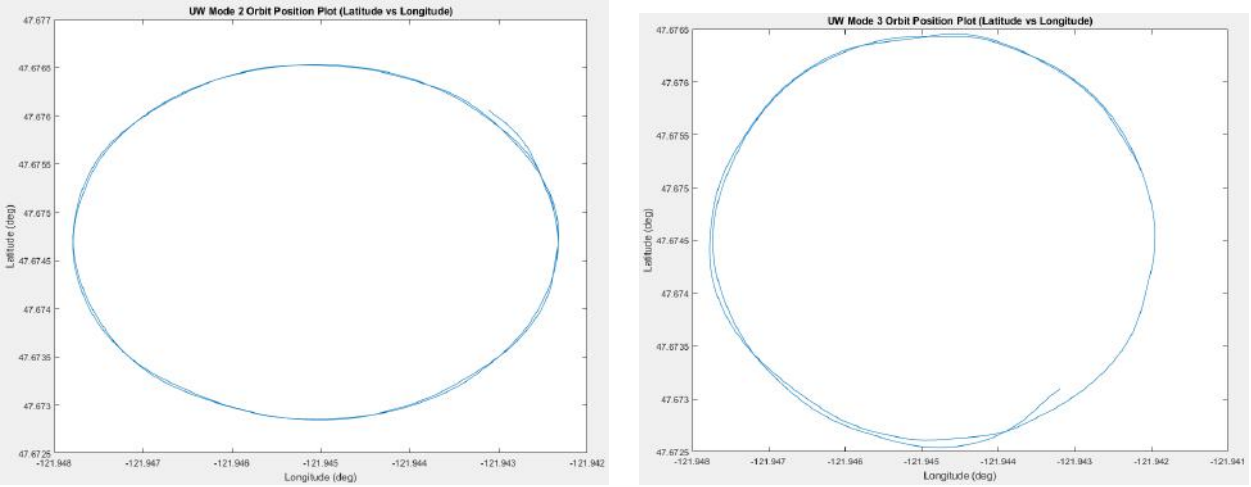
Figure 13(b) shows the performance of UW_Mode_3 at a 200 m radius. UW_Mode_3 was initialized slightly inside the desired orbit trajectory with around 4 kt of wind from the northeast. The orbit controller was able to successfully command the aircraft to a stable orbit in significant wind with relatively smooth bank commands. The affect of lower position update speeds can be seen in the actuated orbit. Small perturbations from the changing wind environment took longer to decay than in UW_Mode_2.

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

VI.D. Vision System Testing Results

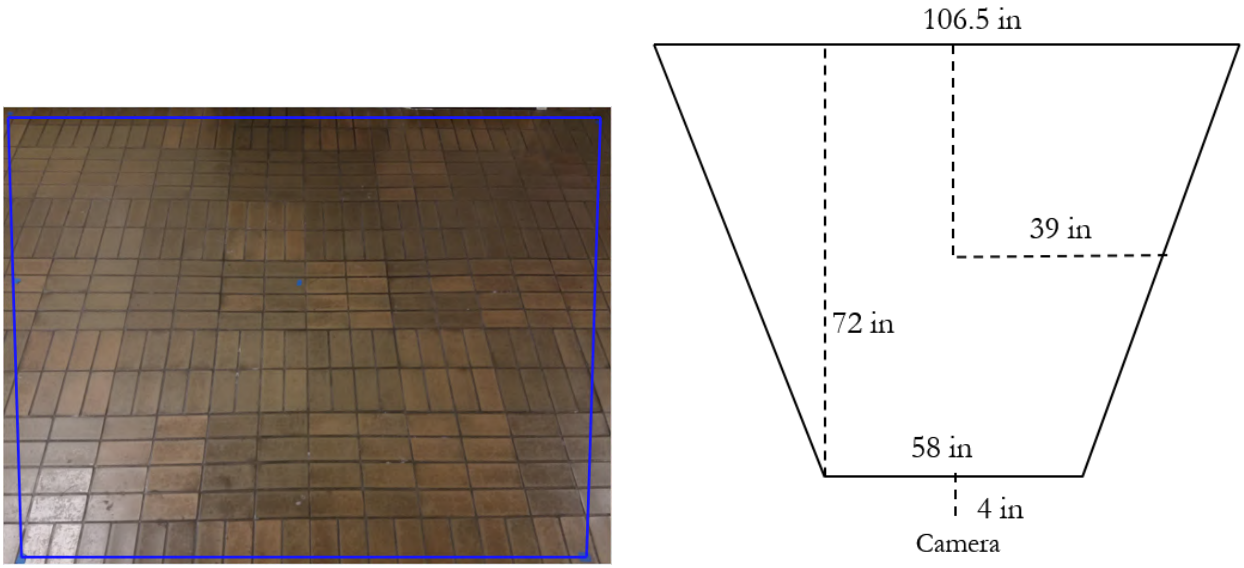
The vision algorithm described in Section III was initially tested by performing a ground test of the algorithm given data from a smart phone camera pointed at the ground at a known height and tilt angle relative to the local horizon. The corners of the camera image were marked on the ground and measured to create the physical projection of the image onto the ground. For a relative altitude of $h_{AGL} = 53$ in and a tilt angle of $\theta_C = -57°$, the dimensions of the physical projection of the image can be seen in Figure 14(b).

The ground distance from the camera to several points on the edge of the physical projection were estimated using the vision algorithm. These estimates were generated by assuming the camera has a resolution of 720×480 and inputting the coordinates of the edge pixels. These estimations were then compared against



(a) UW_Mode_2 at 50 Hz (b) UW_Mode_3 at 3Hz

Figure 13. Real aircraft converging to a stabilized orbit.



(a) Actaul image from the ground test (b) Physical projection of the image frame on the ground.

Figure 14. Vision system ground test camera image and its projection onto the ground.

Table 3. Vision system ground test results.

Point	Coordinate X	Coordinate Y	Estimated Ground Distance (in)	Measured Ground Distance (in)	Relative Error (%)
Center	0	0	34.42	40.00	13.93
Bottom Left Corner	-360	-240	31.79	29.27	8.61
Bottom Right Corner	360	-240	31.79	29.27	8.61
Top Right Corner	360	240	102.61	92.80	10.57
Top Left Corner	-360	240	102.61	92.80	10.57
Left Center	-360	0	50.16	55.87	10.22
Left Right	360	0	50.16	55.87	10.22

the physical measurements of ground distance to gauge the accuracy of the algorithm. The results of this comparison are in Table 3. The relative error for each test case was around 10%, which is non-negligible. However, the ground projection was only an estimate of the true projection because the markings of the corners of the image were approximations. This would introduce error into the measured ground distance, which the ground test assumed to be accurate. In addition, the tilt angle of the camera was measured from the internal IMU on the smart phone. The camera was only stabilized by hand when the IMU measurement and corners of the image were recorded, so inaccuracies in the tilt angle used by the algorithm account for additional error. Overall, the ground test provides worst-case accuracy estimates of the vision algorithm at around 10% of the estimated ground distance.

VII. Conclusions

This paper described 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 provided simulation and flight test results of the proposed visual anchoring orbit controller with GPS radius estimates, while demonstrating that the GPS radius estimates could be replaced with the proposed vision system radius estimates. These results demonstrated the ability of the orbit controller to successfully stabilize an orbit using GPS radius estimates at varying update speeds and in real wind conditions. The ability of the orbit controller to function at update speeds as low as 3 Hz allows for the vision system to be implemenented offboard of the aircraft and provided through the telemtry radio link. Ground test results of the proposed vision system were provided to estimate the accuracy of the vision system. These initial results provide worst case estimates of the vision system at around 10% of the estimated ground radius. Future work in this area will focus on obtaining flight test results of the proposed vision system and flight test results of the combined orbit controller and vision system to demonstrate the entire visual anchoring method.

References

[1]Brandon, J., "GPS Jammers Illegal, Dangerous, and Very Easy to Buy," *Fox News Technology*, 2010.

[2]Volpe, J. A., "Vulnerability Assessment of the Transportation Infrastructure Relying on the Global Positioning System," Tech. rep., National Transporation Systems Center, 2001.

[3]Rysdyk, R. T., Lum, C. W., and Vagners, J., "Autonomous Orbit Coordination for Two Unmanned Aerial Vehicles," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, San Francisco, CA, August 2005.

[4]Rysdyk, R. T., "Unmanned Aerial Vehicle Path Following for Target Observation in Wind," *Journal of Guidance, Control, and Dynamics*, September 2006, pp. 1092–1100.

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

[6]Sinopoli, B., Micheli, M., Donato, G., and Koo, T.-J., "Vision Based Navigation for an Unmanned Aerial Vehicle," *Proceedings of the 2001 ICRA Conference*, 2001.

[7]Blosch, M., Weiss, S., Scaramuzza, D., and Siegwart, R., "Vision Based MAV Navigation in Unknown and Unstructured Environments," *Proceedings of the 2010 ICRA Conference*, 2010.

[8]Stachura, M. and Frew, E. W., "Cooperative Target Localization with a Communication-Aware Unmanned Aircraft

System," *Guidance, Control, and Dynamics*, Vol. 34, 2011, pp. 1352–1362.

[9]Lum, C. W., Summers, A., Carpenter, B., Rodriguez, A., and Dunbabin, M., "Automatic Wildfire Detection and Simulation Using Optical Information from Unmanned Aerial Systems," *Proceedings of the 2015 SAE Aerotec Conference*, Seattle, WA, September 2015.

[10]Lum, C. W., Mackenzie, M., Shaw-Feather, C., Luker, E., and Dunbabin, M., "Multispectral Imaging and Elevation Mapping from an Unmanned Aerial System for Precision Agriculture Applications," *Proceedings of the 13th International Conference on Precision Agriculture*, St. Louis, MO, August 2016.

[11]Ueunten, K. K., Lum, C. W., Creigh, A. A., and Tsujita, K., "Conservative Algorithms for Automated Collision Awareness for Multiple Unmanned Aerial Systems," *Proceedings of the 2015 IEEE Aerospace Conference*, Big Sky, MO, March 2015.

[12]Lum, C. W. and Vagners, J., "A Modular Algorithm for Exhaustive Map Searching Using Occupancy Based Maps," *Proceedings of the 2009 Infotech@Aerospace Conference*, Seattle, WA, April 2009.

[13]Lum, C. W., Vagners, J., and Rysdyk, R. T., "Search Algorithm for Teams of Heterogeneous Agents with Coverage Guarantees," *AIAA Journal of Aerospace Computing, Information, and Communication*, Vol. 7, January 2010, pp. 1–31.

[14]Lum, C. W. and Rysdyk, R. T., "Feature Extraction of Low Dimensional Sensor Returns for Autonomous Target Identification," *Proceedings of the 2008 Guidance, Navigation, and Control Conference*, Honolulu, HI, August 2008.

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

[16]Zhang, K., Zhang, L., and Yang, M.-H., "Real-Time Compressive Tracking," *Proceedings of the 2012 ECCV Conference*, Florence, Italy, October 2012, Part III, LNCS 7574, pp. 866879, 2012.

[17]Lambrechts, P., Bennani, S., Looye, G., and Helmersson, A., "Robust Flight Control Design Challenge Problem Formulation and Manual: the Reserach Civil Aircraft Model (RCAM)," Tech. rep., Group for Aeronautical Research and Technology in Europe, Europe, 1997.

[18]"Setting up SITL on Windows," ArduPilot Dev Team, 2016, `http://ardupilot.org/dev/docs/sitl-native-on-windows.html`.

[19]"X-Plane Website," `http://www.x-plane.com/`, Accessed December 1, 2017.

[20]"X-Plane Plugins," `https://github.com/clum/XPlanePlugins`, Accessed December 1, 2017.

[21]"Mission Planner Overview," ArduPilot `http://ardupilot.org/planner/docs/mission-planner-overview.html/`, Accessed 2016.

[22]"GitHub ArduPilot Repository," GitHub Repository, `https://github.com/ArduPilot/ardupilot`.

[23]"GitHub MissionPlanner Repository," GitHub Repository, `https://github.com/ArduPilot/MissionPlanner`.

[24]Lum, C. W., Larson, R. S., Handley, W., Lui, S., and Caratao, Z., "Flight Testing an ADS-B Equipped sUAS in GPS-Denied Environments," *Proceedings of the AIAA Flight Testing Conference*, Denver, CO, June 2017.

[25]Lum, C. W. and Waggoner, B., "A Risk Based Paradigm and Model for Unmanned Aerial Vehicles in the National Airspace," *Proceedings of the 2011 Infotech@Aerospace Conference*, St. Louis, MO, March 2011.

[26]Lum, C. W. and Tsukada, D. A., "UAS Reliability and Risk Analysis," *Encyclopedia of Aerospace Engineering*, July 2016.

[27]Lum, C. W., Gauksheim, K., Kosel, T., and McGeer, T., "Assessing and Estimating Risk of Operating Unmanned Aerial Systems in Populated Areas," *Proceedings of the 2011 AIAA Aviation Technology, Integration, and Operations Conference*, Virginia Beach, VA, September 2011.