A Gimbal-Supported, Mono Camera, Relative Position Measurement System of a Visually Distinct Object for UAV Guidance

Rostyk Svitelskyi

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

Master of Science in Aeronautics & Astronautics

University of Washington

2019

Committee:

Christopher Lum, Chair

Juris Vagners

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

University of Washington

Abstract

A Gimbal-Supported, Mono Camera, Relative Position Measurement System of a Visually Distinct Object for UAV Guidance

Rostyk Svitelskyi

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

This thesis describes a vision-based system for measuring distance and bearing between an unmanned aerial vehicle (UAV) and a ground object. The system is built of a single camera, mounted on a gimbal, which stabilizes camera attitude. The novelty of this system is that it uses a single camera and tracks a single object on the ground, hence requiring less equipment compared to other types of vision-based navigation systems. Vision system data estimates are fed into the controller to stabilize aircraft on a circular orbit and to maintain fully automated flight, without GPS data. The system can be used as a backup in the case of GPS signal loss. In that case, the UAV will orbit a target until the signal is restored or manual input is received. We describe the hardware chosen to solve this problem and the image-processing and rectification algorithms that determine the position of the target. The position data is used to control the autopilot which was optimized for circling the target. The real flight test results for the system, installed on a fixed-wing UAV, are presented.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

Abbreviations

AFSL	Autonomous Flight Systems Laboratory
AHRS	Attitude and Heading Reference System
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
RxMUX	Servo Multiplexer Unit
SITL	Software In The Loop
UDP	User Datagram Protocol
UAV	Unmanned Aerial Vehicle

Symbols

D_X	Ground distance between X-Axis intercept and target
D_Y	Ground radius between aircraft and Y-Axis intercept
D_T	Ground radius between aircraft and target
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_Y	Slant range between aircraft and Y-Axis intercept
L_T	Slant range between aircraft and target
p	Bank rate of aircraft in aircraft body frame
q	Pitch rate of aircraft in aircraft body frame
r	Yaw rate of aircraft in aircraft body frame
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
V_{GS}	Ground speed of aircraft
V_{IAS}	Indicated airspeed of aircraft
X_{tgt}	X coordinate of target in pixels from center
Y_{tgt}	Y coordinate of target in pixels from center

Greek symbols

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	Aircraft heading
$\dot{\psi}$	Heading rate of aircraft
$\dot{\psi}_{err}$	Heading rate error
$\dot{\psi}_{ref}$	Reference heading rate of aircraft
θ_c	Pitch angle of camera (tilt)
θ_X	Angular position of the target on the image
θ_Y	Angular position of the target on the image
θ_V	Camera vertical field of view
θ_H	Camera horizontal field of view
ψ_c	Pan angle of camera (yaw)
ϕ_c	Roll angle of camera
ϕ_B	Vision bearing angle
ΔA	Aileron deflection
ΔE	Elevator deflection
ΔR	Rudder deflection

ACKNOWLEDGMENTS

I would like to thank my advisor Dr. Christopher Lum for providing all the necessary resources for this project. I would also like to thank Dr. Juris Vagners for the feedback on this project as a committee member. I would like to express my gratitude to Tadej Kosel for his input. I would like to express my appreciation to Ryan Grimes for hardware and autopilot setup. I would also like to thank the other AFSL members: Helen Kuni, Nicholas Price, Ravi Patel, Hannah Rotta, Connor Kafka, Chris Hayner, Zech Latimer, and Nathan Han for their flight test support and help with software integration. This project was partially supported by the University of Washington Royalty Research Fund.

DEDICATION

For my mom and my dad

Chapter 1
INTRODUCTION

1.1 Problem Statement

Tracking circular orbits is a common task for unmanned aerial vehicles (UAV's) [17]. Most of the algorithms rely on a GPS signal. However, the GPS signal can be jammed, suppressed [18], or not available in certain mountainous and industrial areas. At the same time, for a modern UAV, the capability of flying in a GPS denied environment is crucial. This determines a need for backup navigation systems. Popular alternatives to GPS navigation are represented by transponder- and cellular-based systems [13]. But both of these alternatives may also suffer from the same problems as GPS.

A good alternative is represented by the vision-based systems. These systems do not require prior position knowledge. Because it is installed on the aircraft, vision-based navigation does not need continuous radio communication with a control tower and is capable of operating even if radio communication is interrupted. With this advantage, these systems will find a variety of usage in UAV operations such as search and rescue[8, 9, 10], fire prevention[11], cargo delivery, and agriculture[12].

A typical approach to vision-based navigation systems is based either on using two cameras (stereo vision [15]), or on using a camera with a rangefinder [6].

This paper proposes a novel vision-based, non-GPS reliant system to fly the orbit. Our approach also requires a minimal amount of equipment, which reduces UAV mass and energy consumption. We call our UAV operating the vision system, "Camera Operated Navigation Done Outside (GPS) Ranges", or simply, CONDOR. Our system does not require multiple objects on the ground or taking multiple images in a row. Simplicity of the proposed system

makes it a viable backup in the case of a stereo vision malfunction.

1.2 Status of the Problem

Work on this problem was started by Ryan Grimes[7], who proved that one camera system is feasible. Fig. 1.1 presents the UAV circling about a visual distinct target on a flat surface.

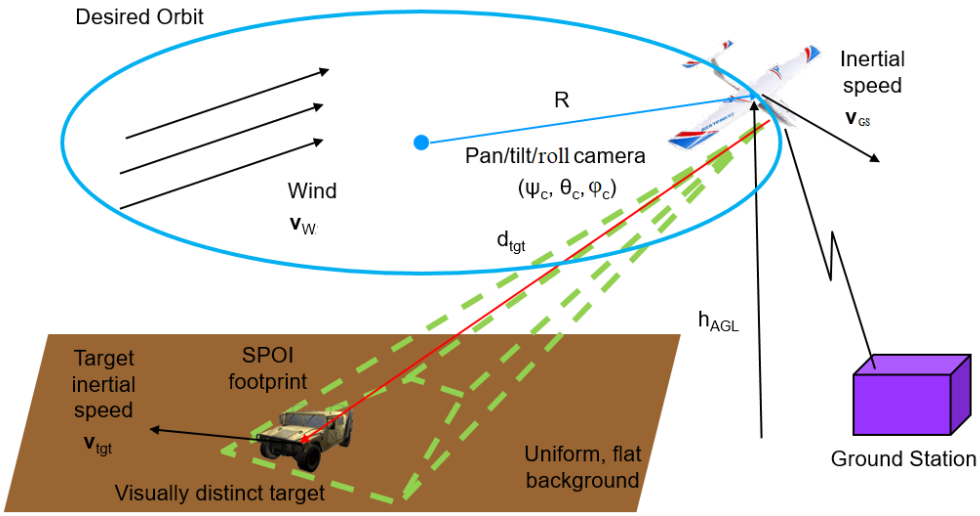


Figure 1.1: Overview of vision system operation. Adapted from [7].

This paper represents the next step towards designing a single-camera-based control system. We use an on-board camera, a gimbal, and a barometric altimeter. Their data are processed by a computer and used to control the aircraft. To facilitate debugging, we are using a ground-based computer which communicates with the aircraft via radio. In the future, the data processing will be moved onboard, making the flight control system totally autonomous.

Chapter 2

FLIGHT TEST EQUIPMENT

Let us start with an overview of the CONDOR system.

2.1 The CONDOR UAV and Ground Control System

Many changes and improvements have been applied to the system from its original iteration [7] . For the UAV airframe, we use the "UAV Skywalker Finwing" (Fig. 2.1(a)). It is equipped with a Pixhawk flight computer, which provides preliminary processing data of the data from sensors, and Pulse-Width Modulated (PWM) signals to operate servomotors. For normal flight, the onboard GPS sensor is sufficient. Flights with the vision system require a custom autopilot mode which uses vision radius data from the telemetry radio, barometric altimeter, indicated airspeed sensor (Pitot tube), and the Attitude and Heading Reference System (AHRS). The full diagram of the CONDOR UAV used for our testing is shown in Fig. 2.2.

We have 8 control channels. The first five are: aileron, elevator, thrust, rudder, and flight mode. These are linked to a first radio transmitter used by a human pilot, who performs manual takeoffs and landings of the UAV. To frame the picture, the gimbal is operated by channels 6 and 7 (gimbal pan and yaw control) which are provided by an additional transceiver. For safety reasons, the gimbal is operated by a separate person (not the pilot). Ch. 8 transmits vision system data for the navigation via the telemetry radio.

For altitude measurements barometric altimeter is used. To increase precision of the measured $\bar{h}_{AGL}$, a precalibration was performed by placing altimeter on different floors of the tower with open atmospheric access. This procedure was done at different outside air temperatures ($t_O = 15°\text{C}$ and $t_O = 25°\text{C}$, which are typical for the flight test site). This

(a) Skywalker Finwing aircraft (b) Gimbal setup

Figure 2.1: CONDOR UAV.

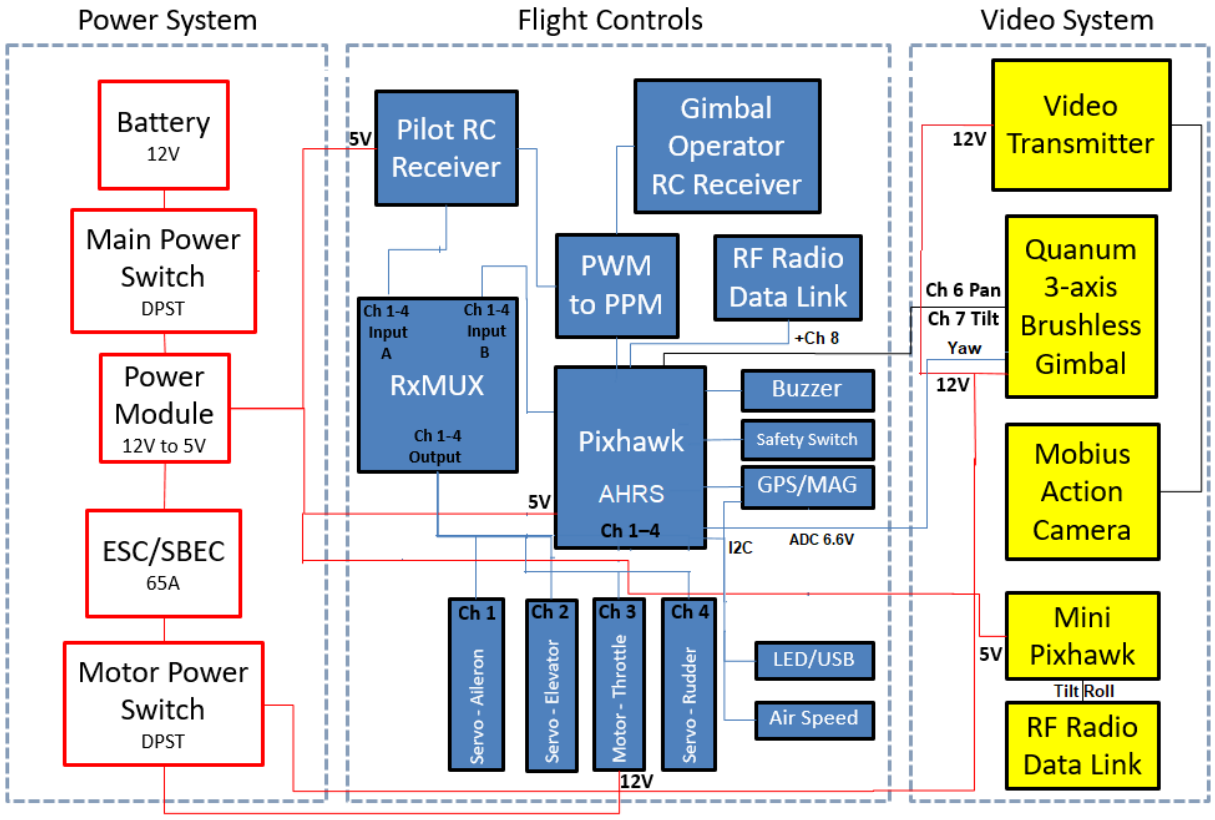


Figure 2.2: CONDOR systems. Adapted from [7].

(a) GCS infrastructure (b) GCS workstation

Figure 2.3: Flight test site.

allowed us to introduce a correction for the temperature difference.

$$h_{AGL} = \bar{h}_{AGL}(1 - (0.004 + (15 - t_O) \cdot 0.00005)\bar{h}_{AGL}) \tag{2.1}$$

The Ground Control Station (GCS) (Fig. 2.3(b)) is a mobile workspace equipped with computers with all the software necessary for our tests, as well as telemetry and video receiver antennas. The CONDOR UAV connections with the GCS are shown in Fig. 2.4. Video and gimbal attitude are downstreamed to the ground computer. The onboard Pixhawk communicates with the modified version of MissionPlanner installed on the onground computer. For testing the system, a blue tent was located in the center of the yellow-green field as shown in Fig.2.5. Due to its shape, this tent can be approximated as a flat rectangle.

2.1.1 Gimbal Setup

The gimbal setup is presented in Fig. 2.1(b). The quanum 3-axis brushless gimbal was chosen to stabilize the camera. The gimbal controller board is connected to the main Pixhawk servo Ch.6 and Ch.7. These channels power the board and give control over pan and tilt. A Mobius camera is mounted on the gimbal, and connected through a USB port to the video

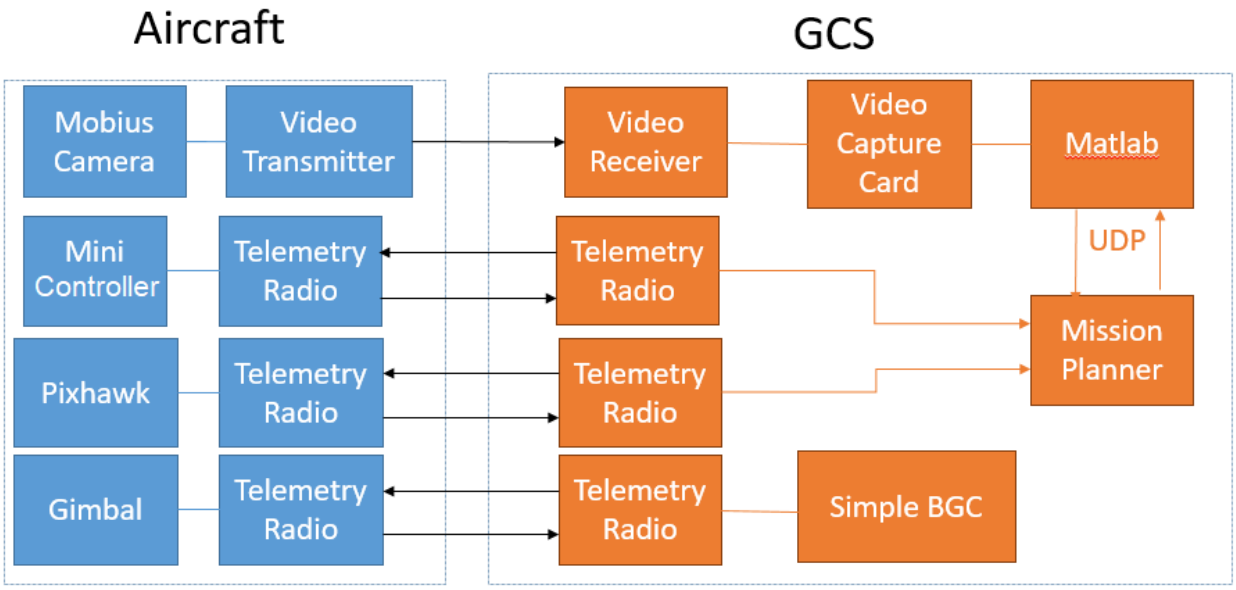


Figure 2.4: Connection diagram between UAV and GCS. Adapted from [7].

Figure 2.5: Tent as a visually distinct object.

transmitter. For tilt and roll angle measurements, a Pixhawk Mini was installed on top of the camera. The Pixhawk Mini was rigidly fixed to the gimbal. As the Pixhawk Mini is used with reduced functionality, for notation convenience, we will call it a mini-controller.

The gimbal communication is also done through the telemetry radio. The mini-controller is connected to MissionPlanner, and the gimbal is connected to the SimpleBGC user interface (both are on the ground computer) , as shown in Fig. 2.6. Since the mini-controller axes were misaligned with the gimbal axes (and thus with the camera), roll and tilt static error was found and calibrated for.

The given gimbal is not equipped with encoders. Therefore, the camera yaw angle relative to the aircraft cannot be measured directly. To fix this, a low-torque, linear, and low-hysteresis, analog potentiometer was installed on the gimbal base, and connected to the yaw axis through a tensioned wire, as shown in Fig. 2.7. The potentiometer was connected to the main Pixhawk 6.6V analog to digital converter (ADC) so that the voltage linearly corresponds to an angle between the camera and the right wing Ψ_G, which can be added to (3.24). This allows changes in pan angle to be tracked and can be used to fly noncircular orbits.

2.2 ArduPlane Description

ArduPlane 3.8.0 was pulled from the GitHub repository[1]. For the orbit controller, a custom mode, based on Chapter 5 math, was created. All the inner and outer loop gains were made accessible as parameters through MissionPlanner for tuning. This mode has access to altitude, airspeed, and AHRS data, as well as the vision radius transmitted through Ch. 8.

2.3 MissionPlanner Description

MissionPlanner 1.3.52 [4]was pulled from the GitHub repository[3]. For the vision system, two MissionPlanners are required to run simultaneously, one for the main Pixhawk, and the second for the mini-controller. The version of MissionPlanner for the main Pixhawk was modified to allow the custom orbit mode to be activated from the GCS. In this way, the

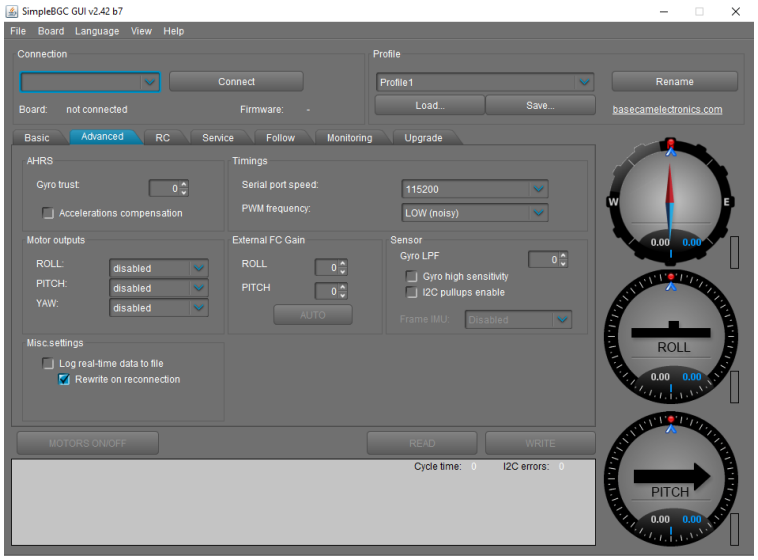


Figure 2.6: User interface for gimbal setup.

Figure 2.7: Gimbal yaw potentiometer setup.

GCS operator could ensure that the vision system was up and tracking, and then switch to fully vision based autopilot mode. A special Python script was designed for the Pixhawk MissionPlanner to communicate with Matlab using a User Datagram Protocol (UDP) Port to retrieve the vision radius and then feed it into the controller. The version of MissionPlanner for the mini-controller has a similar Python script, which sends roll and tilt data from the mini-controller using a UDP Port into Matlab. Vision radius data is sent through the telemetry radio, which simulates channel 8 of a UAV transmitter.

2.4 Matlab Vision Data Processing

The Mobius camera image is transmitted from an external video receiver into the ground computer's video capture card. Matlab reads this image using the Matlab video input package [2]. The tracking object is manually selected by the operator, and is then captured by the Integrated Learning Framework scheme [5]. The tracking algorithm was tested and proved to be reliable [7]. The tracking rectangular size is chosen to be a slightly larger than the tent inside of it. After the tent coordinates (in pixels) and roll and tilt angles, are received, they are processed as described in Ch. 3. The camera field of view and video capture card resolution are coded into the Matlab code before running the application. The vision radius and vision bearing are sent through a UDP Port to the GCS MissionPlanner.

Chapter 3
CORRECTION FACTOR DERIVATION

To get a vision radius estimate, the input data from the gimbal and the vision system are processed in a series of algorithms. First, the object coordinates from the image, in pixels, are corrected for the image distortion which occurred due to the optical lens. Second, a correction for the roll angle is applied to align the image with the horizon, followed by the tilt and the pan angle corrections respectively.

3.1 Image Distortion Correction

Due to a lens imperfection, pixels are shifted from the their positions, especially for the wide-angle lens chosen. As the image distortion depends on the lens only, this correction of the original image has to be done first. For this, we use correction mapping for both axes. To get the correction data, a flat surface was set up with precisely marked points at a known distances d_x, and d_y from the center. The camera was set up at a precisely measured distance d, pointed perpendicular to the surface and into the center point of the checkpoints. The calibration photo is shown in Figure 3.1. Fig. 3.2(b) shows how the coordinates were measured, which matched the vision tracking algorithm, described in Section 2. It was assumed, that the lens distortion is symmetrical in the vertical and the horizontal plane, so that the correction factor was symmetrically mirrored after completed one quadrant.

Using these equations to determine an angle between the center and a specific pixel,

$$\theta_X = \arctan \frac{2X_{tgt} \tan \theta_H}{S_x} \tag{3.1}$$

$$\theta_X = \theta_H \frac{X_{tgt}}{S_x} \tag{3.2}$$

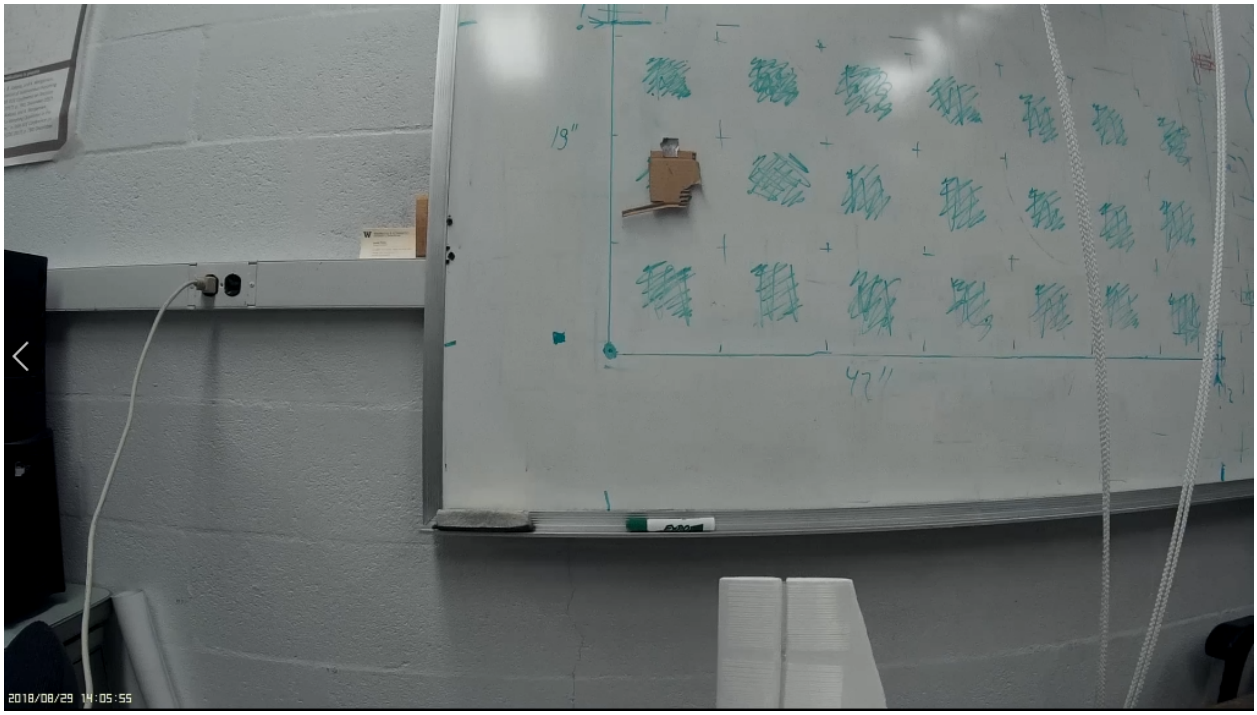


Figure 3.1: Distortion mapping photo.

the distance from the center of the image to the pixel by the horizontal d_{xm}, and the vertical d_{ym} axes can be determined. The Y-axis formula (3.1) is standard for image angle measurements. For a wide angle lens, it was found that in the X-axis it is better to use a linear dependence of angle with respect to pixel (3.2), as that describes the pixel distribution with less error, even without the distortion correction.

$$d_{xm} = d\frac{2X_{tgt}\tan\theta_H}{S_x} \tag{3.3}$$

$$d_{ym} = d\frac{2Y_{tgt}\tan\theta_V}{S_y} \tag{3.4}$$

Then the relative error can be found.

$$C_x = \frac{d_{xm} - d_x}{d_x} \tag{3.5}$$

$$C_y = \frac{d_{ym} - d_y}{d_y} \tag{3.6}$$

Now, two lookup tables $lookup_X$ and $lookup_Y$ are produced, with pixel coordinates normalized to 1, so 0 corresponds to the image center and 1 - to the image boundary pixel $\frac{S_X}{2}$, $\frac{S_Y}{2}$, with the relative error corresponding to each set of coordinates.

After the tracking algorithm measures outputs X_{tgtm} and Y_{tgtm}, these coordinates are divided by S_x and S_y respectively to get normalized coordinates, which are then plugged into a first lookup table for axis X, interpolated linearly between the defined points, and, finally, the correction coefficient is returned. The same procedure is done for the axis Y. Then the measured X_{tgtm}, and Y_{tgtm} coordinates are multiplied by each of these coefficients respectively.

$$X_{tgt} = X_{tgtm} lookup_X \left(\left| \frac{X_{tgtm}}{S_x} \right|, \left| \frac{Y_{tgtm}}{S_y} \right| \right) \tag{3.7}$$

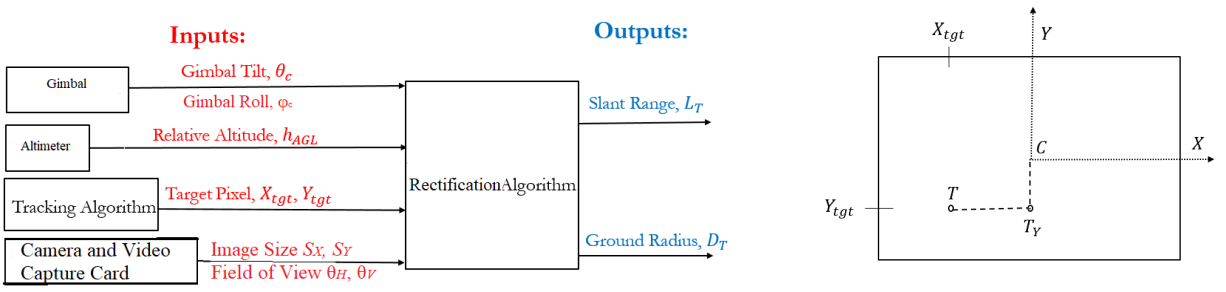
$$Y_{tgt} = Y_{tgtm} lookup_Y \left(\left| \frac{X_{tgtm}}{S_x} \right|, \left| \frac{Y_{tgtm}}{S_y} \right| \right) \tag{3.8}$$

3.2 Rectification Algorithm

Fig. 3.2(a) describes the inputs for the rectification algorithm: camera tilt θ_c and roll ϕ_c relative to the horizon, relative height between UAV and object h_{AGL}, target image coordinates, camera data (distortion mapping), resolution S_x and S_y, and the horizontal and vertical field of view θ_H and θ_V. The formulae are valid for a flat ground.

3.2.1 Roll Angle Geometry

Although the camera is stabilized by the roll angle with the gimbal, due to disturbances, zero roll angle may not always be possible. Thus the image X axis must be aligned with the horizon first. For this procedure, a 2D rotation matrix is used. X'_{tgt} and Y'_{tgt} are the object coordinates as they would be captured by the camera in a zero roll condition.



(a) Rectification Algorithm block diagram.

(b) Origin and coordinates of the image.

Figure 3.2: Tracking and rectification diagrams. Adapted from [7].

$$X'_{tgt} = (\cos(\phi_C) - \sin(\phi_C))X_{tgt} \tag{3.9}$$

$$Y'_{tgt} = (\sin(\phi_C) + \cos(\phi_C))Y_{tgt} \tag{3.10}$$

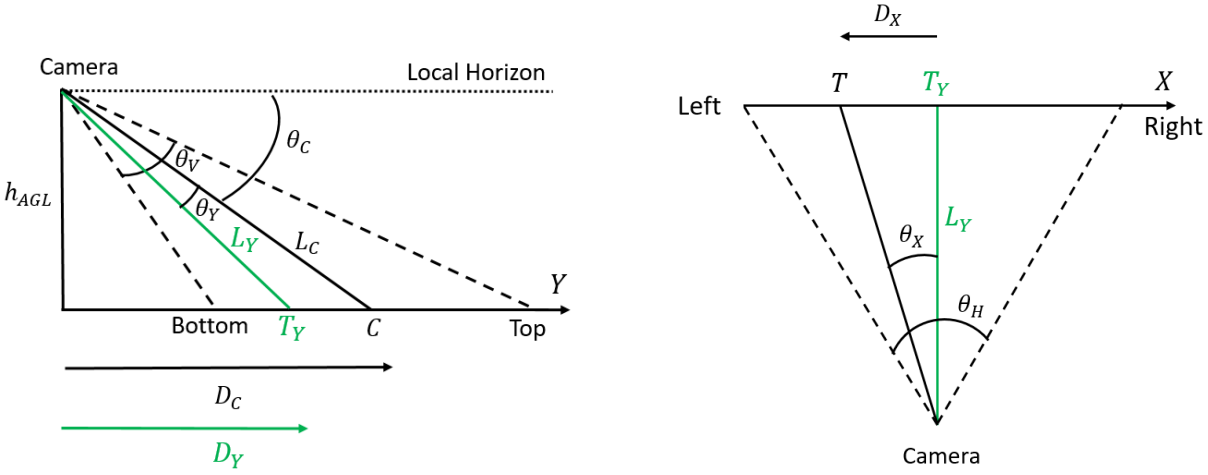
From now, assume X_{tgt} and Y_{tgt} represent roll corrected coordinates.

3.2.2 Tilt Geometry

Looking at the most basic tilt axis geometry Fig. 3.3(a), it may seem that the ground radius D_C can be easily calculated through the trigonometric equalities (3.11). But there exists a displacement due to the image being tilted, as the ground surface is projected onto an angled two-dimensional plane of the camera. Graphical description of the distortion is shown in Fig. 3.4. The more oblique the image, the larger the error grows when the object is located at a distance from the camera.

$$D_Y = \frac{h_{AGL}}{\tan(\theta_C + \theta_Y)} \tag{3.11}$$

Ref. [14], proposes a photo rectification to solve this problem. The photo rectification processes the tilted image and outputs an equivalent image as if the camera is pointed vertically down.



(a) Pan axis geometry.

(b) Tilt axis geometry.

Figure 3.3: Pan and tilt axes geometry [7].

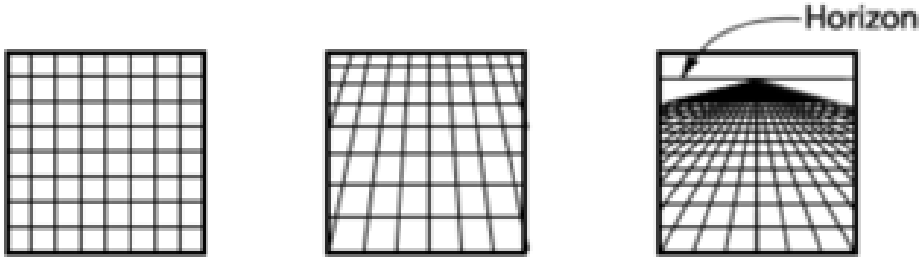


Figure 3.4: Effects of the oblique image on plane perspective [16].

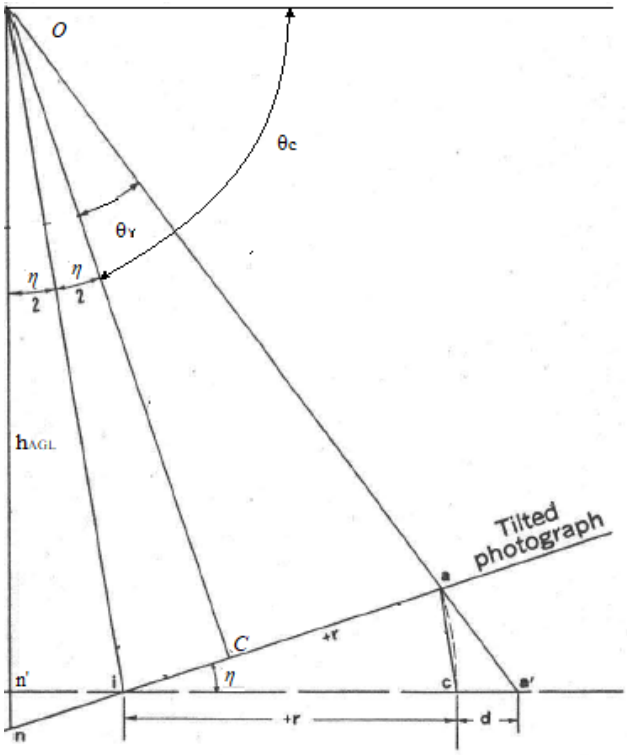


Figure 3.5: Geometry of rectification correction. Adapted from [14].

The rectification begins with a correction along the Y-axis to get the distance along this axis to the UAV. As seen in Figure 3.5, point a' represents the actual object located on the ground. OC is the image center, C is the direction where the camera points; i is the isoline, where the image and ground surface intersect; n' is the nadir point; line ia represents the plane of an image. As shown a' projects into a. If the geometry from (3.11) is considered, then the distance to point c would be calculated, but the calculation has an error. In order to calculate the right ground radius, the distance d must be found. First, find θ_Y and η.

$$\theta_Y = \arctan(2\frac{Y_{tgt}}{S_y}\tan(\theta_V)) \tag{3.12}$$

$$\eta = 90 - \theta_C \tag{3.13}$$

Now, let us find the distance from nadir to isoline, and isoline to a. The triangles $\Delta On'i$

and ΔOCi are equal.

$$n'i = h_{AGL} \tan(\frac{\eta}{2}) \tag{3.14}$$

$$+r = ia = ic = n'i + h_{AGL} \tan(\theta_Y) \tag{3.15}$$

Now, we find the correction distance d, and sum up all the components.

$$d = \frac{+r^2}{\frac{h_{AGL}}{\sin(\eta)} - +r} \tag{3.16}$$

$$D_Y = n'i + +r + d \tag{3.17}$$

$$L_Y = \sqrt{{D_Y}^2 + {h_{AGL}}^2} \tag{3.18}$$

3.2.3 Pan Geometry

As seen in Fig. 3.6, D_X can be calculated through simple trigonometry. The black net on the ground surface has the same meaning as in Fig. 3.4. As shown the pan angle is measured in the plane of the camera tilt, thus the slant range L_Y comes in handy as the tilted D_X can be calculated readily.

$$D_X = L_Y \tan(|\theta_X|) \tag{3.19}$$

$$\theta_X = \theta_H \frac{X_{tgt}}{S_x} \tag{3.20}$$

3.2.4 Combined Geometry

Now let's use the Pythagorean theorem to combine both axes to the slant range L_T and the ground distance D_T.

$$D_T = \sqrt{D_X^2 + D_Y^2} \tag{3.21}$$

$$L_T = \sqrt{D_T^2 + h_{AGL}^2} \tag{3.22}$$

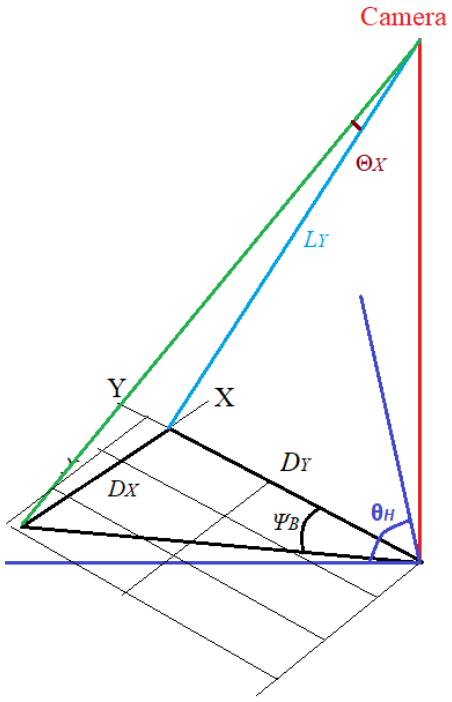


Figure 3.6: $X-$axis geometry in 3D.

3.3 Vision Bearing Estimation

For the purposes of navigation, the full relative coordinate may be useful. Full relative coordinate consists of radius to the target and the bearing, measured from the right wing. The angular deviation from the right wing can be calculated with Eq. (3.23). This deviation consists of the sum of the vision bearing Ψ_{VB}, shown in Fig. 3.6, the sideslip angle β, and the gimbal yaw angle relative to the wing Ψ_G as in Eq. (3.24). Using the magnetic compass direction, an actual bearing can be calculated. Strictly speaking, it is not needed for the circular orbits, however, it may be useful for other flying patterns.

$$\Psi_{VB} = \arctan\left(\frac{D_X}{D_Y}\right) \tag{3.23}$$

$$\Psi_B = \Psi_{VB} + \Psi_G + \beta \tag{3.24}$$

3.4 Error Analysis

The vision radius estimate depends on the precision of multiple measurements. In this section, the dependence of radius error on other variables will be shown. For that, we combine the formulae for radius estimation from section 3.2, and take derivatives by altitude, target coordinates, pitch, and roll. In this way, if every derivative is multiplied by one unit of each variable, the result will show estimation error in meters. The tilt is varied at $\Theta_C = -20°; -30°; -40°$. Altitude is set to 100 m, and the roll is zero. Plots (Figs. 3.7-3.21) show how the error is distributed along the image as its value depends on the target coordinates in the frame. Due to the flat ground assumption, figures for $-20°$ tilt are cropped at 196.3 pixels as the horizon starts above. Approaching 90°, the tilt correction factor formulae become invalid. The results show that the errors at a small tilt and at the upper part of the image are significant. This is a trade-off for using one camera. For the best performance, high precision sensors must be used, all the equipment must be calibrated, and tilt should be as small as possible. The total error is calculated as the sum of all the errors.

$$\begin{aligned}\Delta D_T =& \Delta h_{AGL}\frac{\partial D_T(h_{AGL},\Theta_C,\phi_C,X_{TGT},Y_{TGT})}{\partial h_{AGL}} + \Delta\Theta_C\frac{\partial D_T(h_{AGL},\Theta_C,\phi_C,X_{TGT},Y_{TGT})}{\partial\Theta_C} \\ &+ \Delta\phi_C\frac{\partial D_T(h_{AGL},\Theta_C,\phi_C,X_{TGT},Y_{TGT})}{\partial\phi_C} + \Delta X_{TGT}\frac{\partial D_T(h_{AGL},\Theta_C,\phi_C,X_{TGT},Y_{TGT})}{\partial X_{TGT}} \\ &+ \Delta Y_{TGT}\frac{\partial D_T(h_{AGL},\Theta_C,\phi_C,X_{TGT},Y_{TGT})}{\partial Y_{TGT}}\end{aligned} \tag{3.25}$$

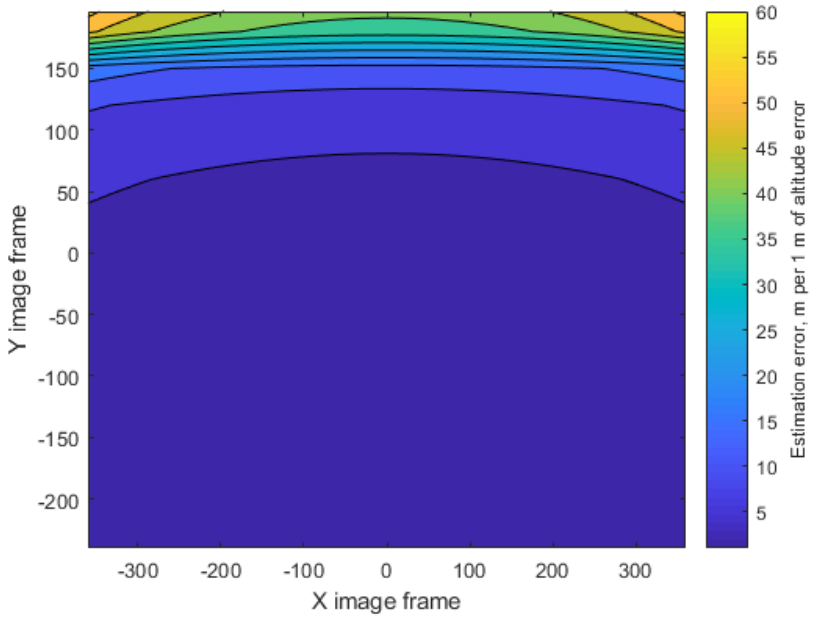


Figure 3.7: Estimation error with altitude measurement error of +1 m at −20° tilt.

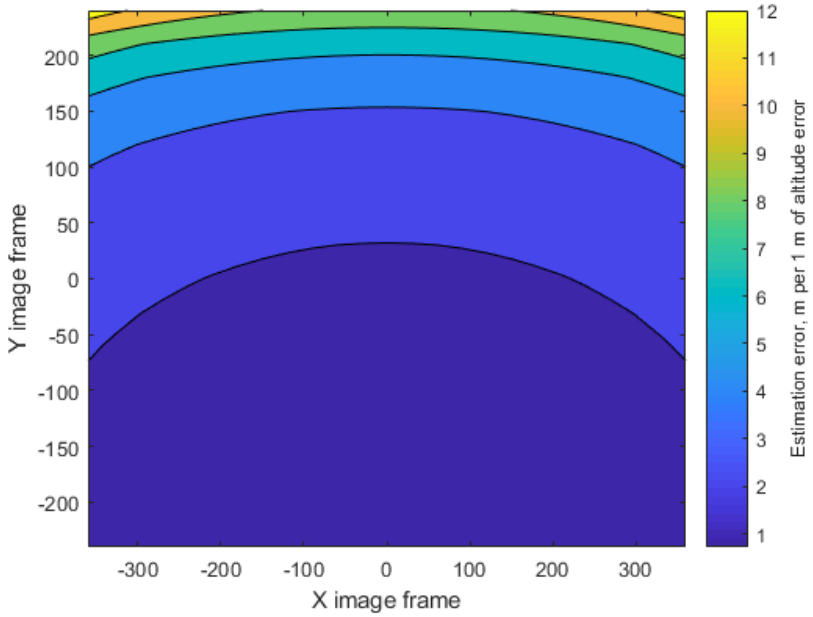


Figure 3.8: Estimation error with altitude measurement error of +1 m at −30° tilt.

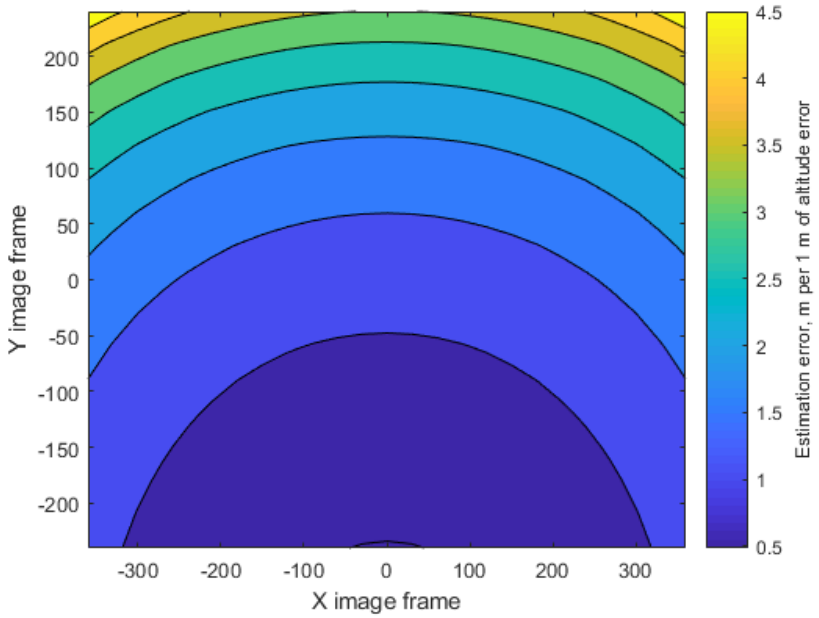


Figure 3.9: Estimation error with altitude measurement error of +1 m at −40° tilt.

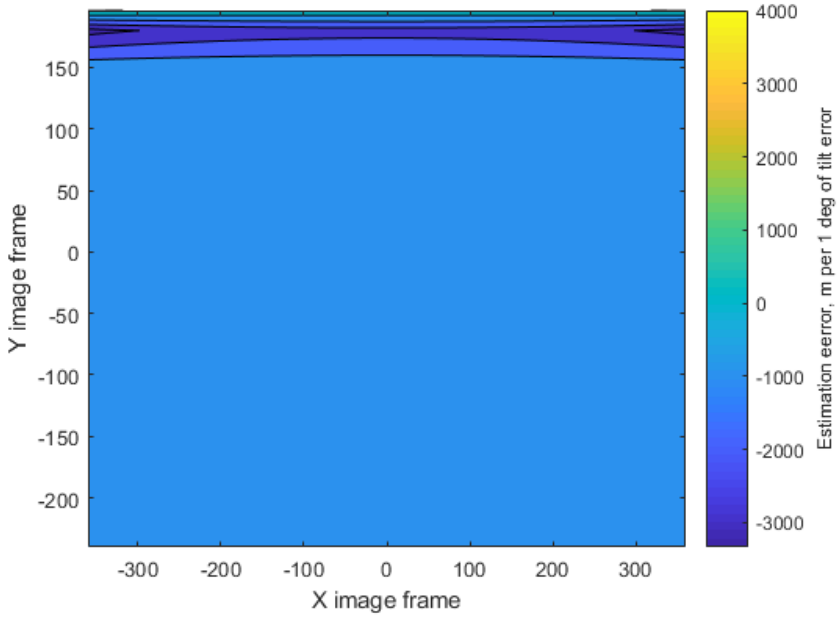


Figure 3.10: Estimation error with tilt measurement error of 1° at −20° tilt.

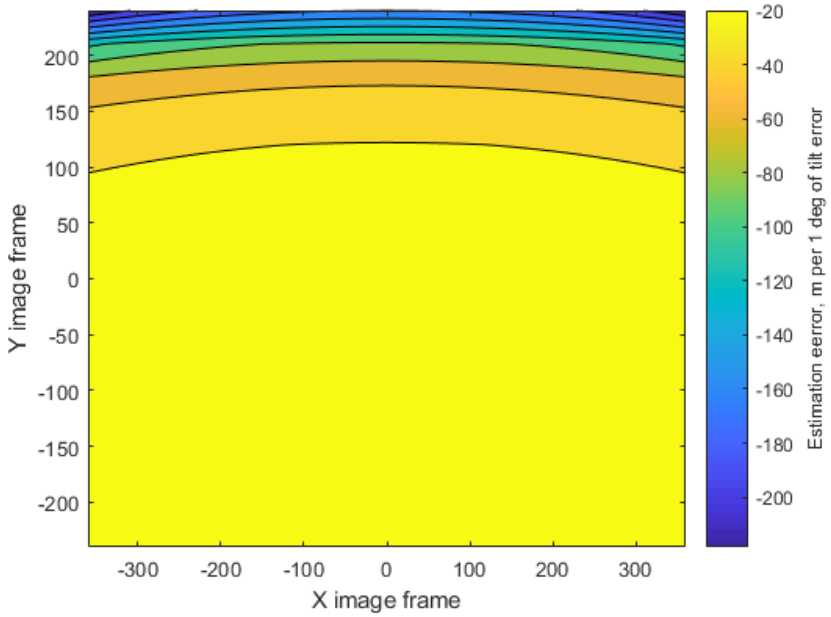


Figure 3.11: Estimation error with tilt measurement error of 1° at −30° tilt.

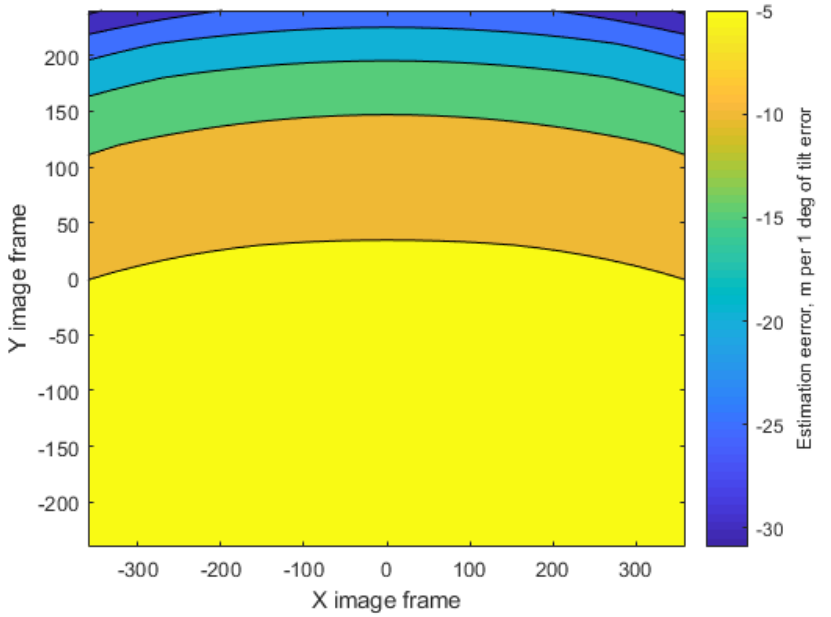


Figure 3.12: Estimation error with tilt measurement error of 1° at −40° tilt.

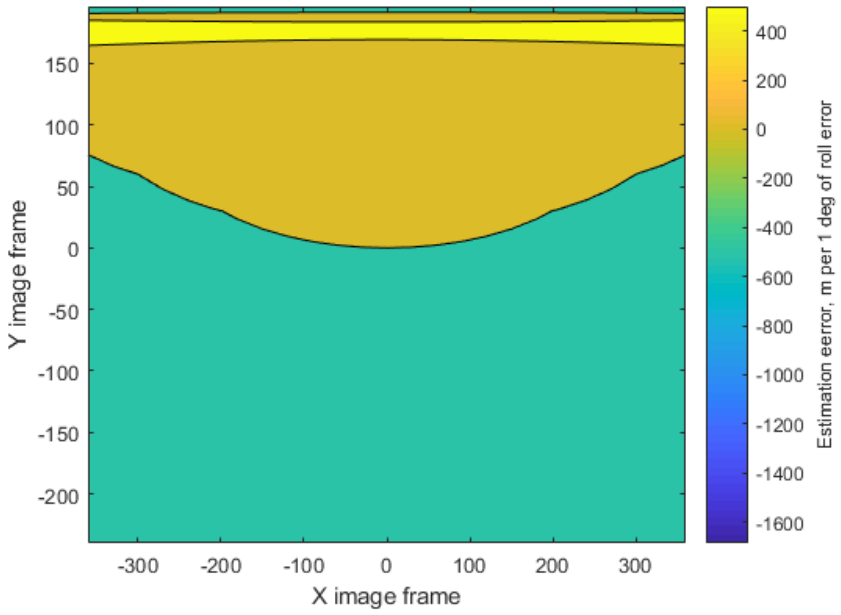


Figure 3.13: Estimation error with roll measurement error of 1° at −20° tilt.

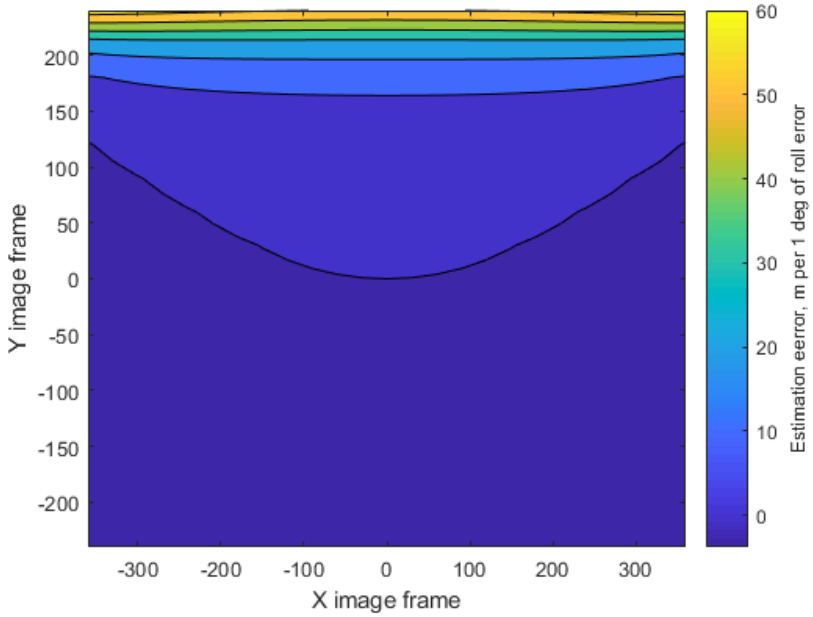


Figure 3.14: Estimation error with roll measurement error of 1° at −30° tilt.

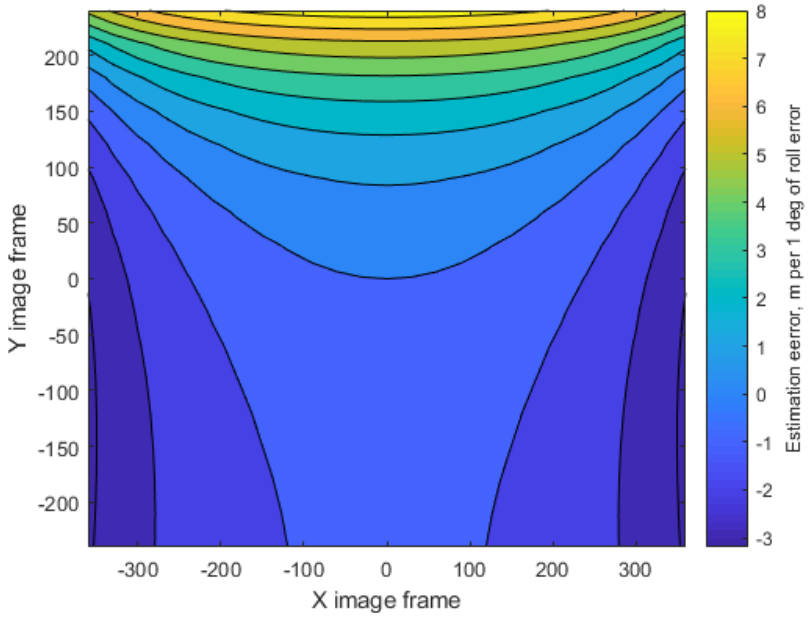


Figure 3.15: Estimation error with roll measurement error of 1° at −40° tilt.

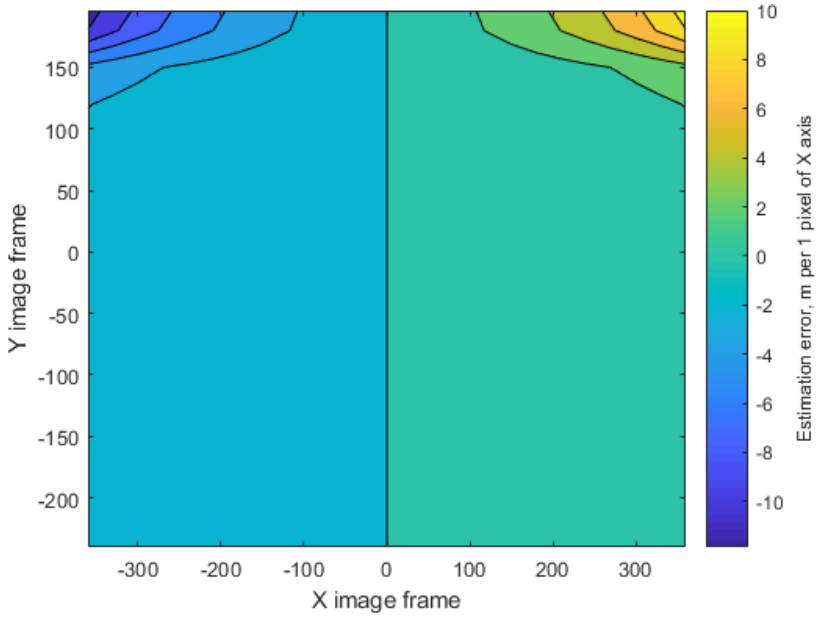


Figure 3.16: Estimation error with X coordinate measurement error of 1 pixel at −20° tilt.

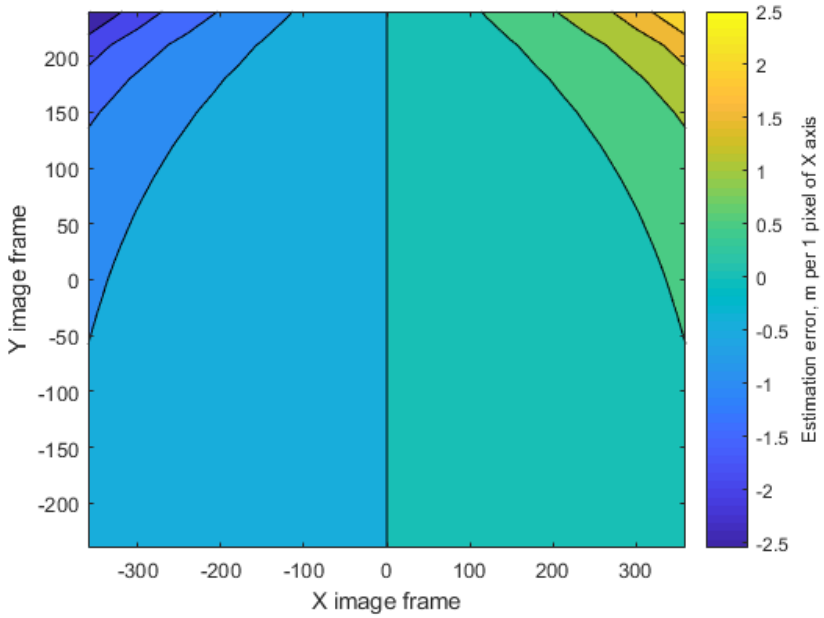


Figure 3.17: Estimation error with X coordinate measurement error of 1 pixel at $-30°$ tilt.

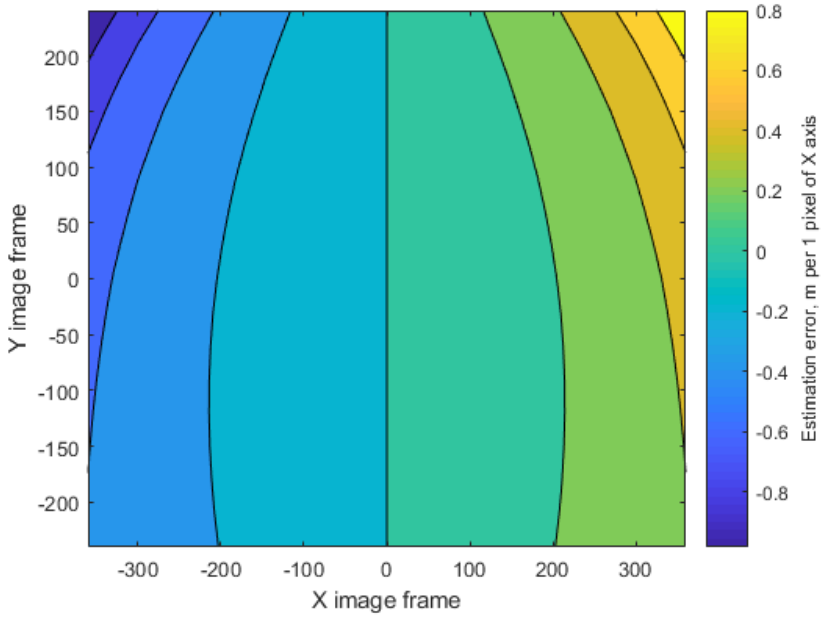


Figure 3.18: Estimation error with X coordinate measurement error of 1 pixel at $-40°$ tilt.

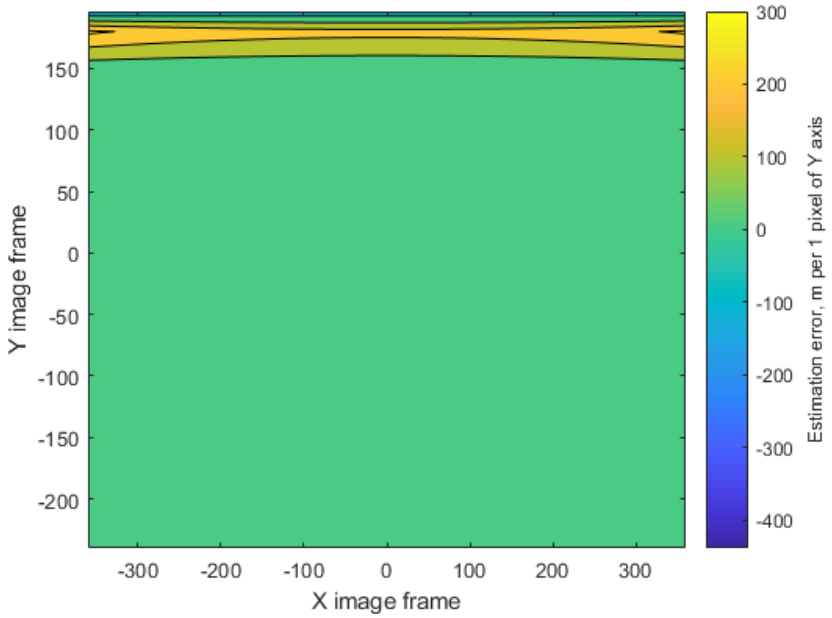


Figure 3.19: Estimation error with Y coordinate measurement error of 1 pixel at $-20°$ tilt.

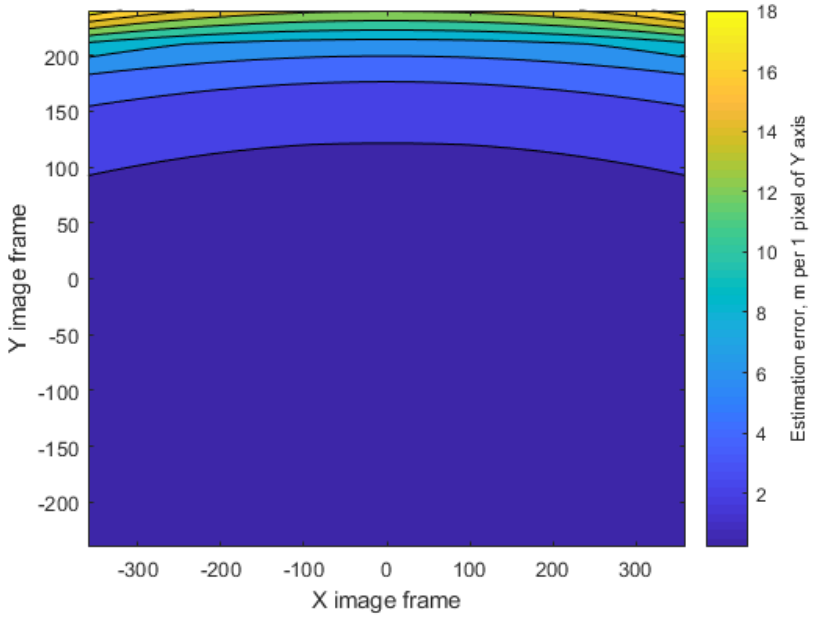


Figure 3.20: Estimation error with Y coordinate measurement error of 1 pixel at $-30°$ tilt.

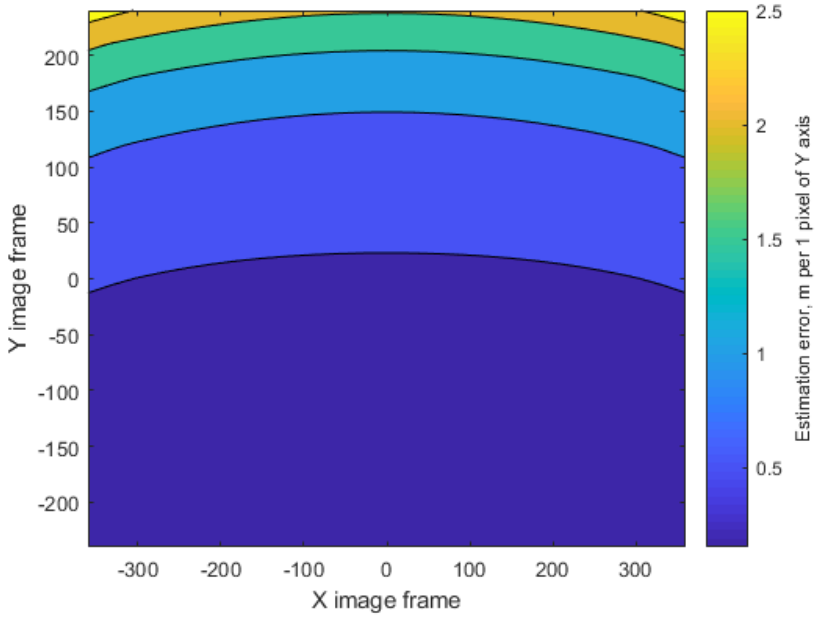


Figure 3.21: Estimation error with Y coordinate measurement error of 1 pixel at $-40°$ tilt.

Chapter 4

RECTIFICATION ALGORITHM RESULTS

For the on-ground test, a series of images with different altitude, tilt, and pan values have been obtained. The setup is described in Chapter 2.

For the first test, the camera was mounted at $h_{AGL} = 4.2'$ (camera lens can move additional $\pm 0.1'$) as shown in Fig. 4.1, and the object was installed 4.8' away. Camera tilt and pan were varied to get the tracked object in different positions. The results are presented in Table 4.1. The second test was done at $h_{AGL} = 44'$ as shown in Fig.4.3(a), with the object located 75'. Results are shown in Table 4.2.

However, it was noticed, that when the tilt angle changes too quickly, as shown in Fig. 4.2, the above estimates are not correct. To model this dynamical error effect, the gimbal was manually tilted as in Fig.4.3. The measured data are presented in Fig.4.2. This error is caused by two factors:

- Gimbal AHRS is not capable of measuring tilt and roll precisely during prolonged motions, it requires an approximately steady position for measurements to converge.

- Telemetry radio is not always capable of transmitting a message with data reliably. As can be seen in Fig.4.2(b), the number of steps between the messages vary. Image processing is done at 7.5 images per second, but the telemetry data may not be accessible for up to a second even though both antennas are located close to each other while testing. When this system is completely localized on the aircraft, this will not be an issue.

Thus, with the existing equipment, the vision system provides reliable results in static position, the only error comes when there is a prolonged change in tilt or roll.

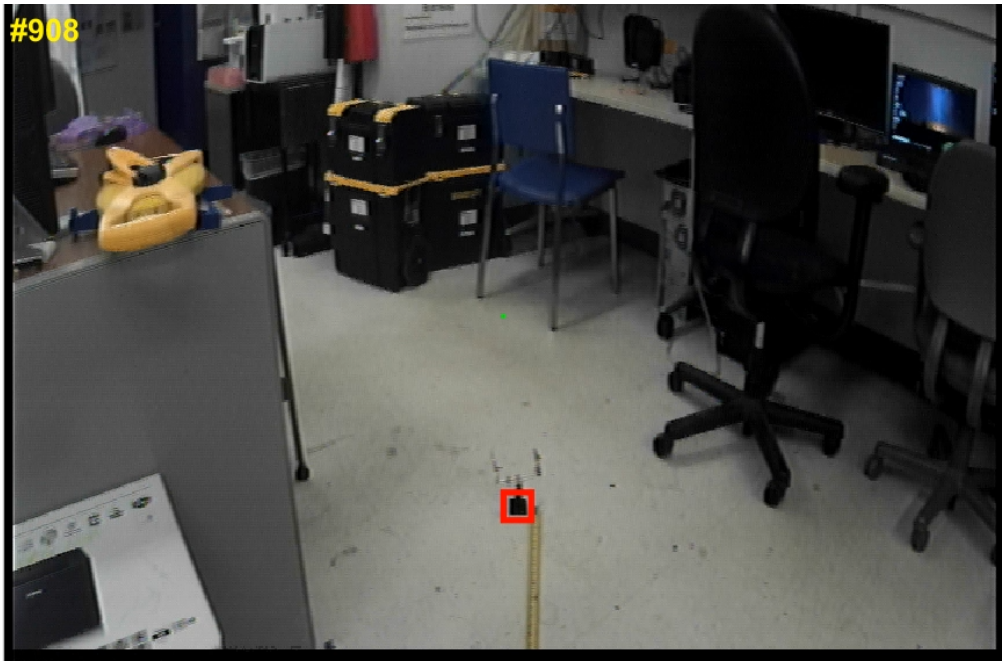
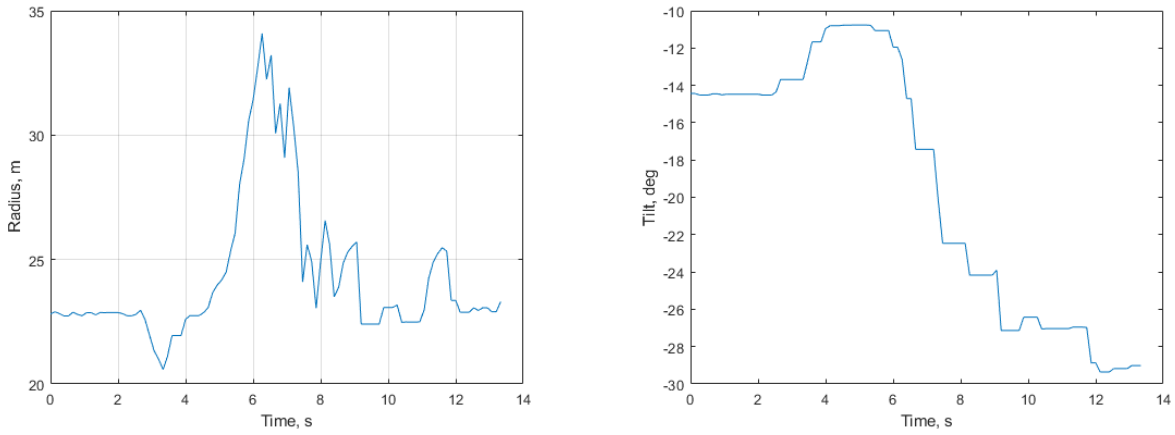


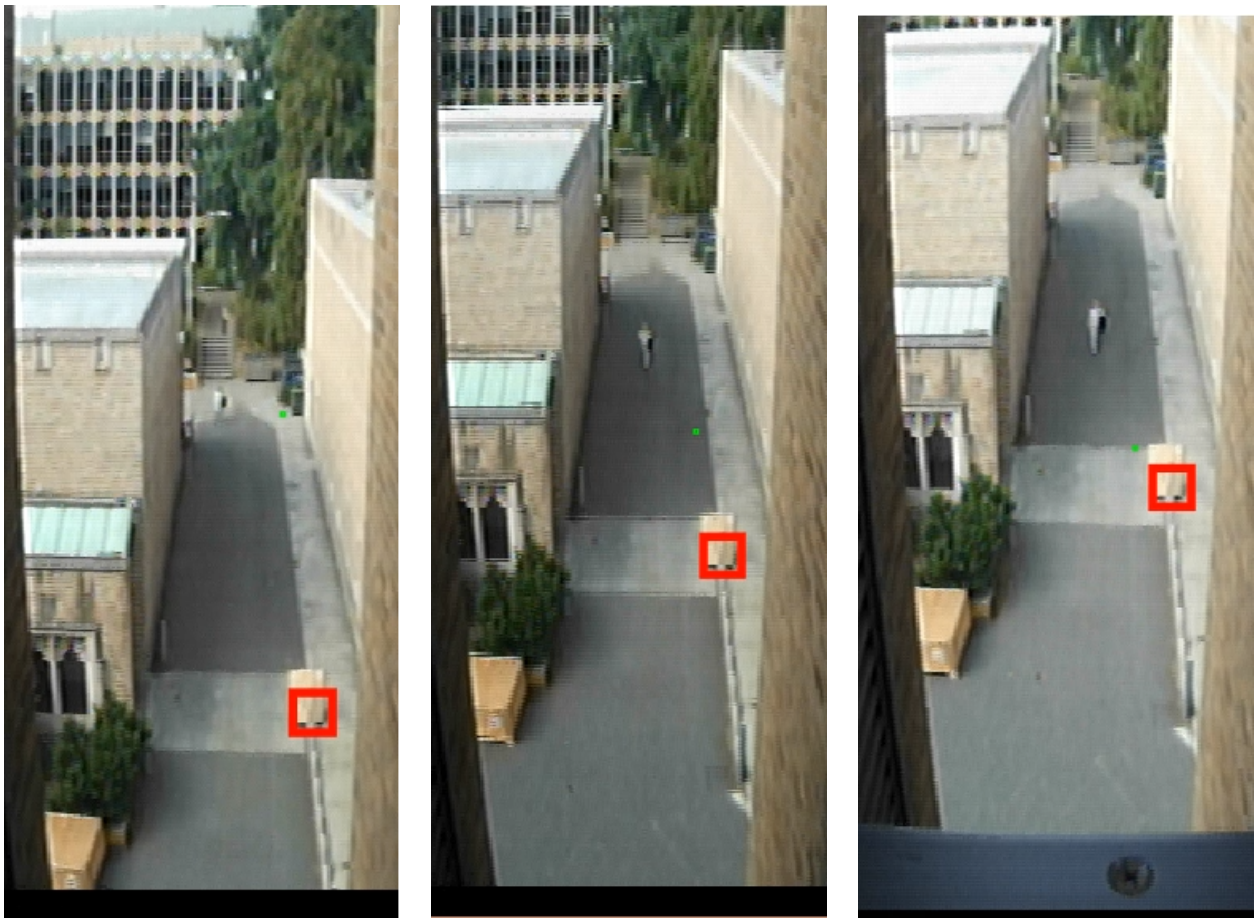
Figure 4.1: Vision system ground test location.



(a) Vision estimates under change of tilt.

(b) Tilt change.

Figure 4.2: Dynamic error description.



(a) Initial position. (b) Tilt up motion. (c) Final position.

Figure 4.3: Modelling dynamical errors situation.

Table 4.1: Vision system ground test low altitude.

Tilt, deg	X_{TGT}	Y_{TGT}	**Roll, deg**	**Radius, m**	**Relative error, %**
-22.110	192.90	-164.76	-1.09	4.550	-5.2083
-21.764	9.90	-135.76	-1.32	4.625	-3.6458
-20.726	-209.10	-167.76	-0.96	4.762	-0.7917
-35.769	195.90	-2.76	-1.44	5.033	4.8542
-38.184	-2.10	0.24	-1.06	4.9854	3.8625
-38.107	-207.10	8.24	-1.11	4.917	2.4375
-47.971	196.90	102.24	-1.46	4.930	2.7083
-47.802	-17.10	153.24	-0.71	5.207	8.4792
-47.815	-200.10	114.24	-0.44	5.229	8.9375

Table 4.2: Vision system ground test high altitude.

Tilt, deg	X_{TGT}	Y_{TGT}	**Roll, deg**	**Radius, m**	**Relative error, %**
-36.209	-0.08	80.61	-2.09	76.374	1.8320
-27.035	16.92	-14.39	-0.05	75.395	0.5267
-14.138	14.92	-149.39	-0.42	74.865	-0.1800

Chapter 5
CIRCULAR ORBIT STABILIZING CONTROLLER

To test the full vision system in autoflight, we used a circular orbit. As the aircraft is not equipped with a sideslip angle sensor, the vision bearing was not used. It was shown in [7] that the vision radius alone is sufficient to hold the desired distance from the object. Below we do a quick recap of the controller and show the new modifications.

5.1 Lateral Control

The controller's lateral channel structure is shown in Figure 5.1.

The outer loop controller calculates the heading rate required to drive the radius error down to zero.

$$\dot{\psi}_{err} = -(K_{P_{outer}}(R - R_{des}) - K_{D_{outer}}\dot{R}) \tag{5.1}$$

The outer loop is responsible for corrections to the current heading rate, which is defined in (5.2) under the assumption of a coordinated turn.

$$\dot{\psi} = \frac{V_{GS}}{R} \tag{5.2}$$

The inner loop controller is designed to set the desired heading rate using the control surfaces. It is responsible for stability and responsiveness of the aircraft.

The reference heading rate consists of the error heading rate and the current heading rate. The ground speed (V_{GS}) is required, however only indicated airspeed (V_{IAS}) is measured, so this could lead to an error. Also, if the UAV approaches too close to the target, the component may blow up, due to R in the denominator, and make the UAV to turn inside even more. That is why this component must be limited. As the desired radius and V_{GS} are known, the boundaries are set at $\pm 40\%$ of $\dot{\psi}$ value.

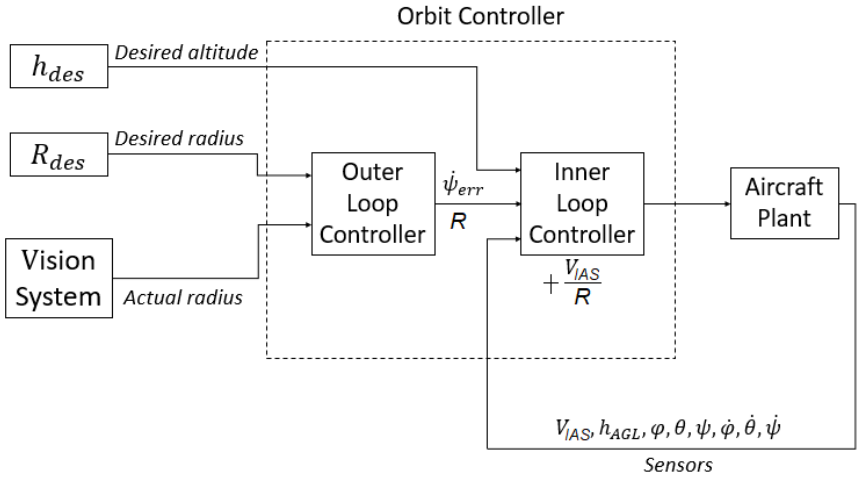


Figure 5.1: Orbit stabilizing controller block diagram. Adapted from [7].

$$\dot{\psi}_{ref} = \dot{\psi}_{err} + \dot{\psi} \tag{5.3}$$

The desired bank angle is calculated, and the aileron command is produced through the proportional-derivative controller.

$$\phi_{ref} = \arctan\left(\frac{V_{GS}\,\dot{\psi}_{ref}}{g}\right) \tag{5.4}$$

$$\begin{aligned}\Delta A &= -\left(K_{P_\phi}\left(\phi_{ref} - \phi\right) - K_{D_\phi}\,\dot{\phi}\right) \\ &= -\left(K_{P_\phi}\left(\phi_{ref} - \phi\right) - K_{D_\phi}\,p\right)\end{aligned} \tag{5.5}$$

One of the safety steps is to limit the ϕ_{ref} component to $\pm 35°$.

If the bank angle commands heading rate to be greater or less than desired, the rudder will compensate the remaining error. The rudder deflection is calculated with a proportional controller. $\dot{\Psi}$ is in the inertial frame. It is calculated from yaw and pitch gyroscopes as the rotation rate is measured in body frame.

$$\begin{aligned}\Delta R &= K_{P_{\dot{\psi}}}(\dot{\psi}_{ref} - \dot{\psi}) \\ &= K_{P_{\dot{\psi}}}(\dot{\psi}_{ref} - r\frac{\cos(\phi)}{\cos(\theta)} + q\frac{\sin(\phi)}{\cos(\theta)})\end{aligned} \tag{5.6}$$

There exists a possibility that rudder may command a rapid turn. As an additional safety, another control loop is set. In case the bank angle exceeds $\pm 45°$, the bank angle hold will be activated and held a at $45°$ bank angle, and the rudder will be set to $\Delta R = 10°$. In this way, the UAV will hold bank angle and constant altitude to ensure safety, until the calculated rudder input is reduced or the pilot takes over. The bank angle hold is a proportional-derivative controller.

$$\Delta A = -\left(K_{P_\phi}\left(45 - \phi\right) - K_{D_\phi}\dot{\phi}\right) \tag{5.7}$$

5.2 Altitude Hold

A simple proportional-integral controller is used for the altitude hold.

$$\theta_{ref} = K_{P_h}\left(h_{des} - h_{AGL}\right) + K_{I_h}\int_0^t \left(h_{des} - h_{AGL}\right) dt \tag{5.8}$$

The elevator deflection is calculated as in (5.9).

$$\Delta E = -\left(K_{P_\theta}\left(\theta_{ref} - \theta\right) - K_{D_\theta}\dot{\theta}\right) \tag{5.9}$$

Chapter 6

ORBIT CONTROLLER SIMULATOR TEST

The next step was to program the orbit controller into the ArduPlane C++ code. In order to test the code before flying the actual aircraft, the software in the loop (SITL) simulator was used. A generic small UAV model is simulated in JSBSim, connected to MissionPlanner. Though the UAV model does not represent the actual aircraft, it can still be used to debug the controller. The code loaded in the simulator is identical the one loaded on the Pixhawk. The only modification done was that the default speed of the simulated UAV was set to match the real one, so that in the case of wind, controller error is represented more accurately.

The vision system cannot be simulated in the simulator, so the GPS measurements were used to simulate the vision radius. After the aircraft takes off, it is set into the guided mode at a 120 m radius, which loiters the aircraft about a point using GPS. The desired radius of the vision orbit is set to 110 m. Now, a script takes GPS distance to the center of the orbit, introduces a delay of .33 seconds, which is the most typical telemetry data latency time. This distance is sent through PWM Ch.8. When ready, the custom mode itself is switched on. In the process of tuning, satisfactory parameters for the controller were found. The test was done with the wind of 8 kt as shown in Fig.6.1. The results demonstrated that the controller is capable of transitioning 10 m to the desired orbit, maintaining a stable orbit, though it is shifted due to the wind as the controller law depends on V_{GS} and custom mode does not have access to this value.

(a) Initial 120 m orbit with GPS controller.

(b) Transition onto 110 m orbit by custom mode.

(c) Custom mode orbit.

Figure 6.1: SITL simulator results of the custom mode.

Chapter 7
FLIGHT TEST RESULTS

This section presents the results of the vision system estimates along with the controller performance on a full non-GPS flight. Two sets of results are present: one with the original controller, and one with a modified controller for better orbit tracking.

7.1 Vision System Results

For the flight test, a tent was placed in the middle of the field as it is described in Chapter 2.1. After takeoff, the UAV was set to loiter around the tent at the desired altitude of 100 m and a reference radius of 110 m. After that, the MissionPlanner Python and Matlab scripts were started. When the vision radius estimates matched the GPS radius, the custom flight mode was switched on. Figure 7.1 shows the vision radius estimates as compared to the GPS values. As seen in the figure, the vision system estimated the radius at all times, except for the seconds 54-58, when the vision tracking was lost.

The overshoots between 15 and 25 seconds, and 46 and 56 seconds are dynamical errors and can be explained by two factors.

During the testing, it was found that the tilt and the roll angle outputs from the mini-controller may be incorrect. This error usually happens on the transition from the downwind leg to the crosswind leg, where the wind causes the aircraft to change velocity and sideslip angle. In Figure 7.4, these points are located on the north-western part of the loop. The same error may also be caused by the rapid flat turn, e.g. the wings are kept level and rudder is used to make turn. This error is similar to that in Fig. 4.2. It can be fixed by using better AHRS or improving an Extended Kalman Filter for attitude estimation.

After the test, it was noticed that the video capture card crops the image non symmetri-

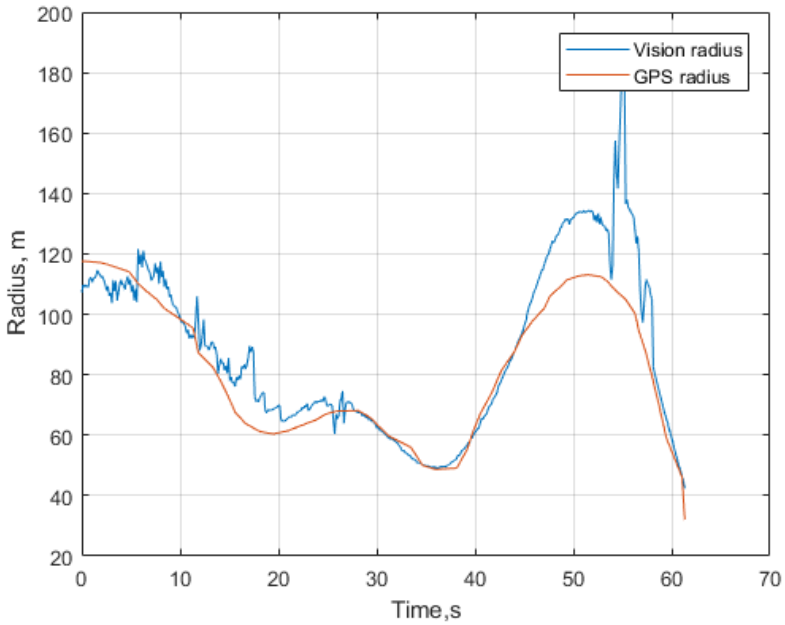


Figure 7.1: Vision system performance.

Figure 7.2: Original picture cropped by the video capture device, and asymmetrical distortions.

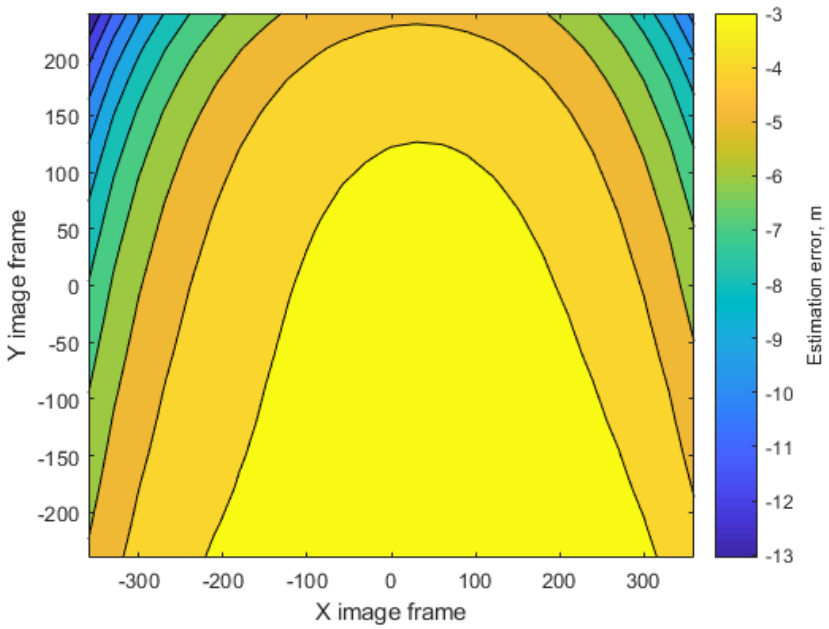


Figure 7.3: Errors distribution for vision orbit at 50 seconds.

cally, and the center of the image does not coincide with the center of the camera. In Fig.7.2, the red line shows the boundaries of the image being processed, as compared to a full image, and the green arrows point at the two dots placed symmetrically with respect to the camera position. The roll angle is zero, and the camera is pointed perpendicular to the wall. It is seen, that beyond the left dot there is more distance to the left boundary, which means bigger distortion at the left side. The distortion correction was designed for the upper-right quadrant as described in Ch. 3. Due to the image asymmetry, this gives bigger error in the left part of the picture and in the X_{TGT} and Y_{TGT} corrections.

In addition, due to imperfect roll stabilization, there was a roll angle of $-11°$, which caused higher errors on the left side of the image. During the seconds 46-56, the tent was in the upper-left corner of the frame, so the effect of error increased. Fig.7.3 depicts the total error if we assume all the contributing factors (3.25) had a measurement error $+1$ of respective measurement unit. The asymmetry due to roll is also seen.

7.2 Visual Orbiting Results

7.2.1 Original Controller Results

Fig. 7.4 presents the trajectory followed by the UAV in the custom mode. The reference radius of 110 m wasn't held properly. In Figure 7.5 it can be seen, that the radius trajectory deviates from the reference radius of 110 m, returns, and deviates again. This is due to wind, which forces the aircraft to turn inside at the start of the loop, decreasing the vision radius. This causes the component of Eq. (5.2) to blow up, resulting in a greater right turn command. The first outer loop setting didn't allow to compensate for the increase.

This can be illustrated in the following way. Equation components (5.3) are shown in Figure 7.6, and these two components are summed up. The outer loop gains for the first component in this flight were too low to compensate for the second component, even though it was limited to $\psi = 0.21$. That is the sum of these components remains positive during the entire maneuver, which corresponds to the right bank angle. On the downwind leg the UAV accelerates, which makes it move farther northwest after the first loop. In Figure 7.7, we can see the consistent right bank angle commanded by the inner loop controller. Figure 7.8 shows the rudder holding the right deflection to hold the desired rate of turn. In Figure 7.9 aileron deflections are shown, and the left deflection is commanded to hold the desired bank angle, which is typical for this aircraft.

7.2.2 Modified Controller Results

The following was offered to fix the controller performance. Instead of (5.3), we used average ground speed V_{GS} divided by the commanded radius.

$$\dot{\psi} = \frac{V_{GS}}{R_{DES}} \tag{7.1}$$

This will keep the second component constant. Then the outer loop controller gains can be increased to allow for a left turn, the inner loop controller can be tuned for UAV stability.

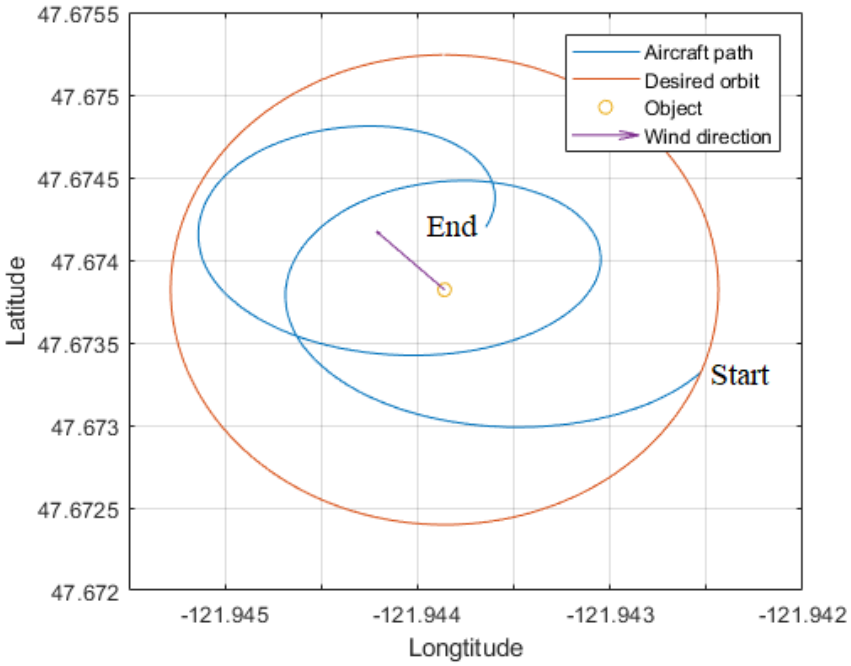


Figure 7.4: Orbits done using Vision System guidance.

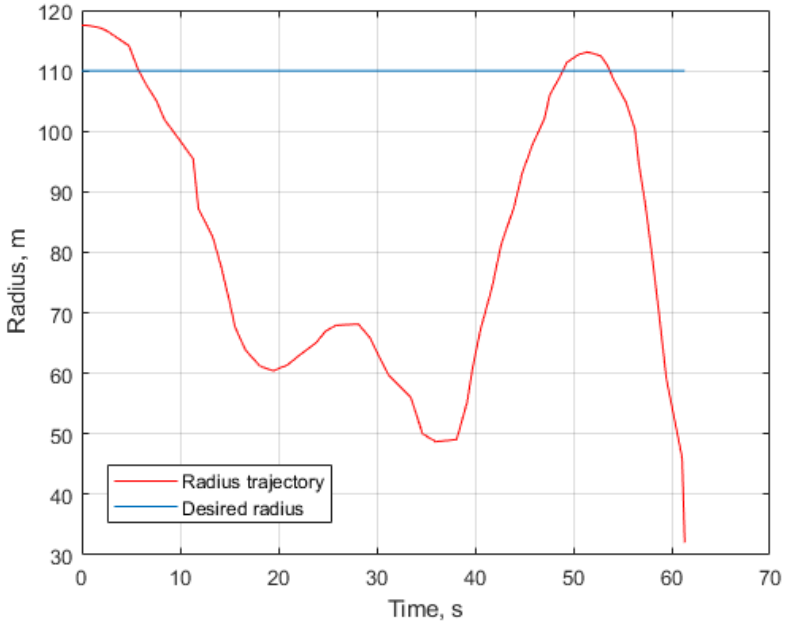


Figure 7.5: Radius trajectory compared to the desired radius.

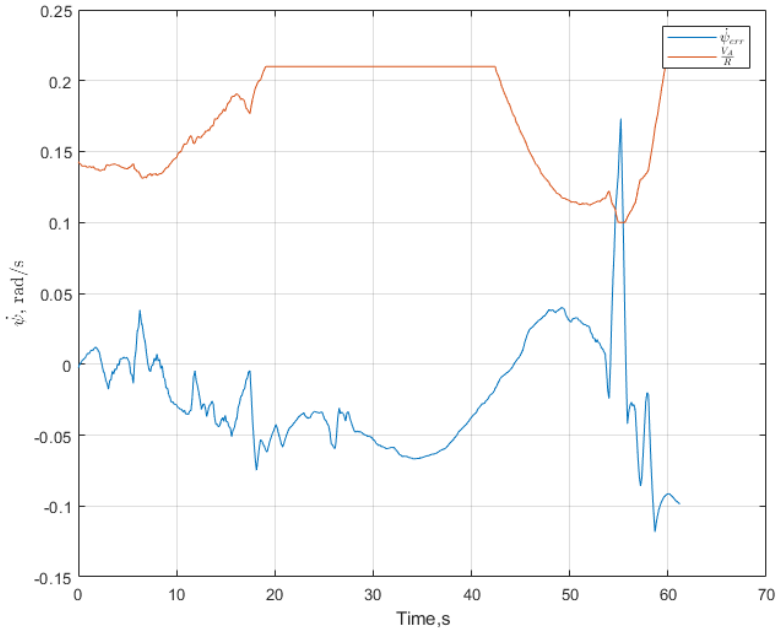


Figure 7.6: Outer loop controller components.

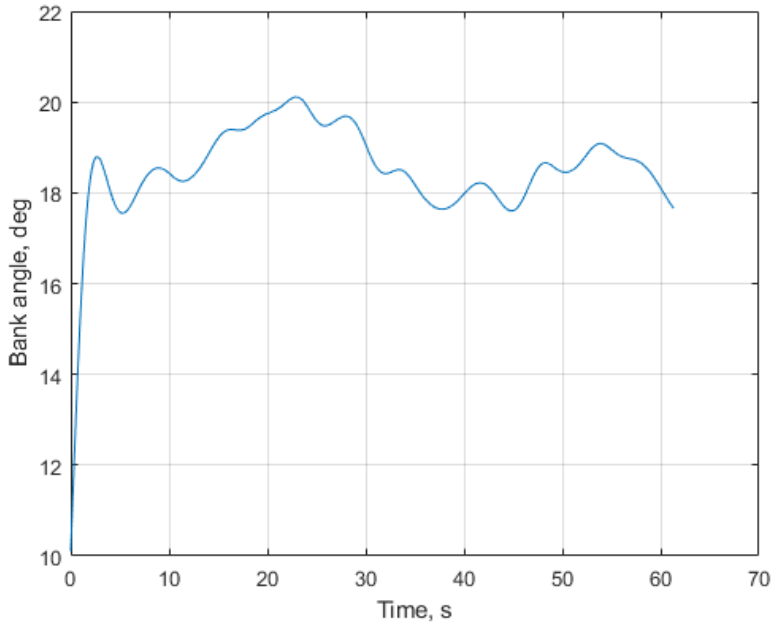


Figure 7.7: Bank angle during vision loop.

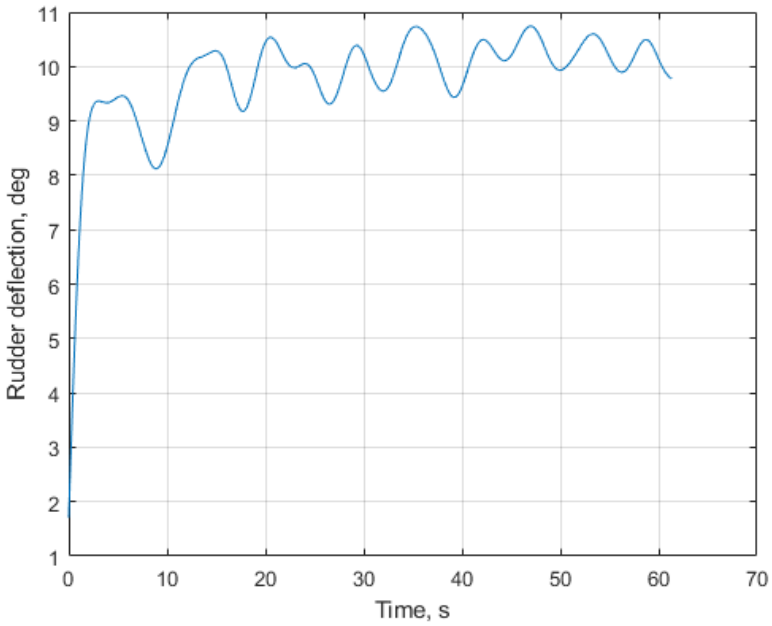


Figure 7.8: Rudder deflection during vision loop.

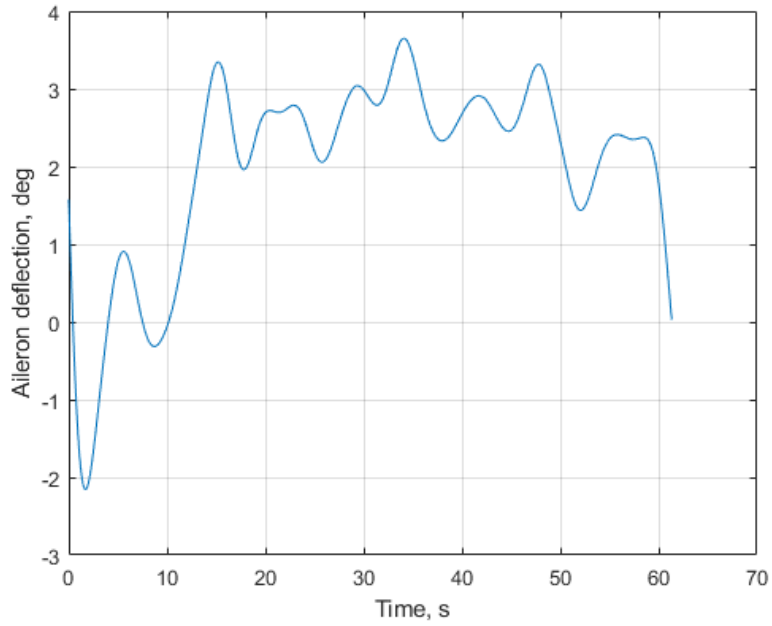


Figure 7.9: Aileron deflection during vision loop.

This logic was applied to the aileron, while the rudder, acting as a compensator, kept initial logic. Table 7.1 describes the difference.

The vision results are presented in Fig.7.11. The test site temperature was measured with error, and there was an underestimation of altitude, thus the results are lower than if altimeter was set up properly. Jumps at seconds 22 and 32 are caused by a jump in the roll and pitch angles of the gimbal, respectively. The resultant loop is presented in Fig.7.10. The new outer loop controller for the ailerons is presented in Fig.7.12. The current iteration of the outer loop controller had enough authority for ailerons. The first half of the loop was tracked with a small error. At the wind side, the UAV was blown inside the loop, and the rudder, which had the old logic, continued to turn UAV to the right as shown in Fig.7.13. This made UAV turn further inside the loop. The ailerons attempted to counteract this by increasing the left deflection, as shown in Fig.7.14.

A next step for improving the controller performance will be to implement the same logic for constant heading rate in the rudder controller, and implement a turn coordinator, which will improve the aileron controller logic as it is currently based on the assumption of no lateral force acting on the UAV.

7.2.3 Controller Logic Comparison

Table 7.1 shows how the desired heading rate is computed. The desired heading rate is used to calculate the desired bank angle under the coordinated turn assumption. An aileron command is issued to hold the desired heading rate. The desired heading rate consists of two components. Fig. 7.15 depicts original logic where the blue current orbit gives the base heading rate of the current circle, Fig. 7.16 shows modified logic, where the base heading rate is calculated by the red desired orbit (ideal heading rate at desired radius).

7.3 Vision System Estimation Error Compilation

Over the course of the flight and ground testing the error sources were identified and presented in Table 7.2. The errors found were either static or dynamical. The static errors

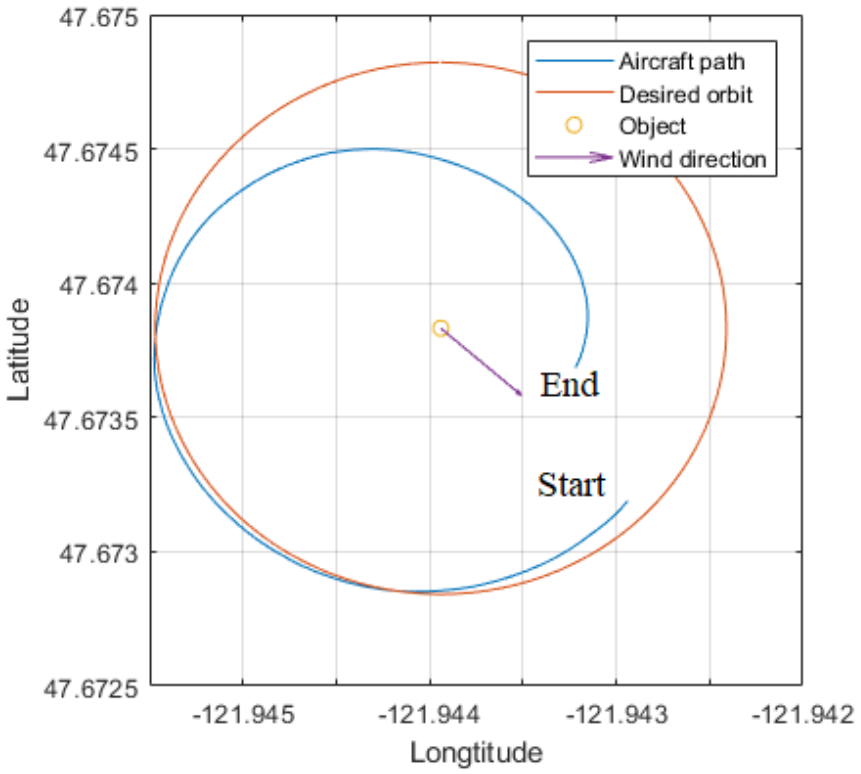


Figure 7.10: Second orbit done using Vision System guidance.

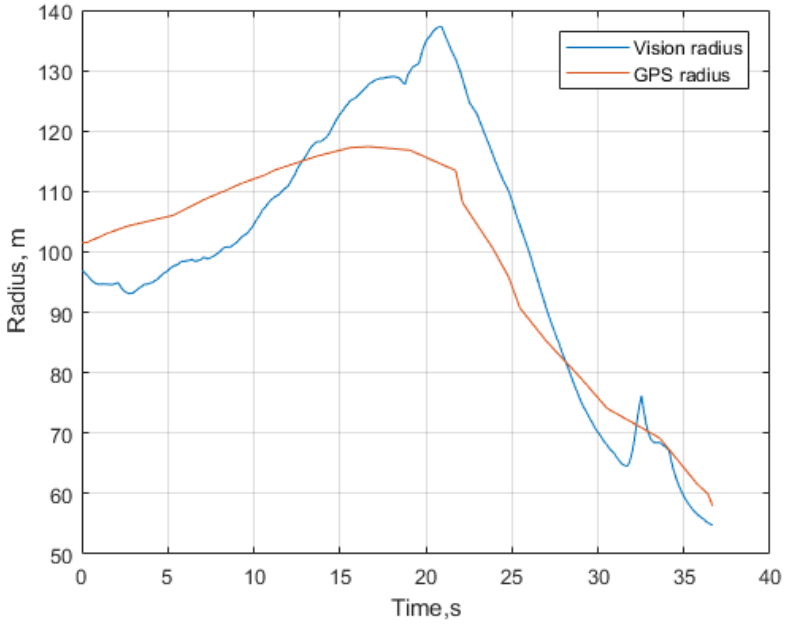


Figure 7.11: Second orbit vision system performance.

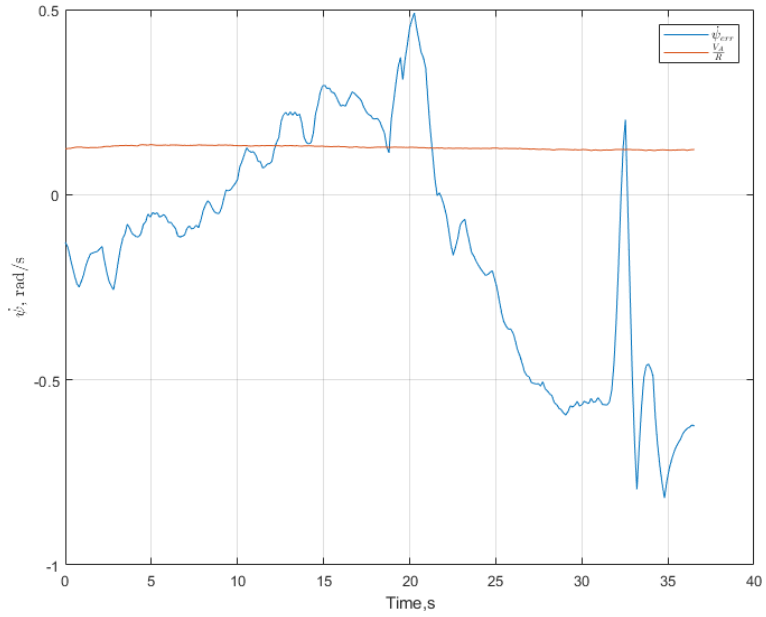


Figure 7.12: Second orbit outer loop controller components.

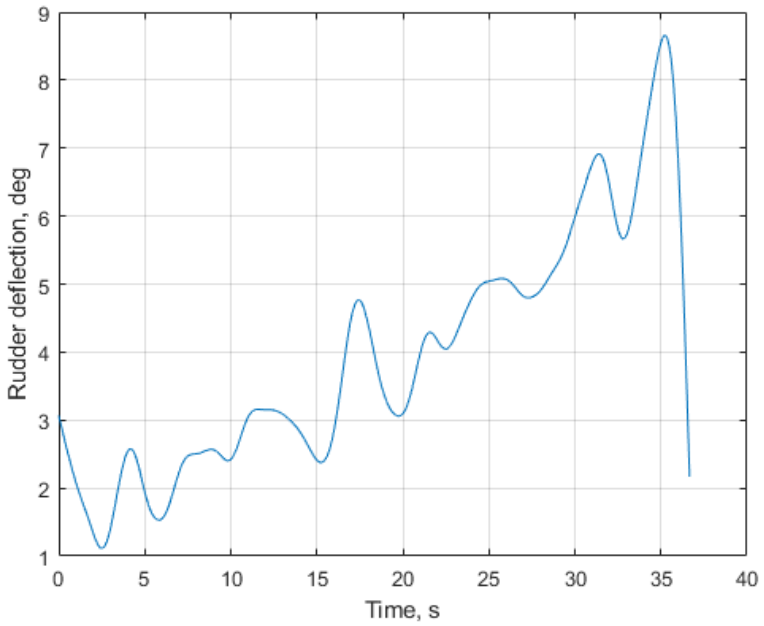


Figure 7.13: Rudder deflection during second vision loop.

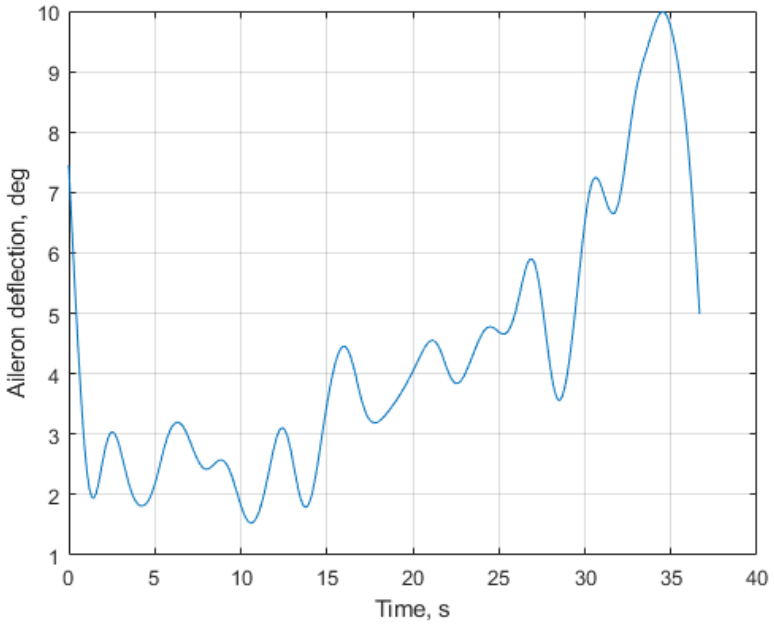
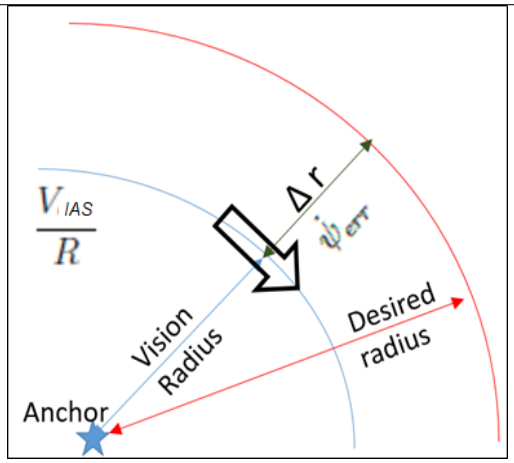
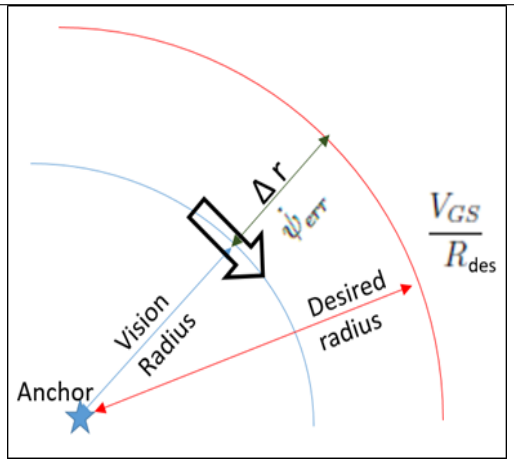


Figure 7.14: Aileron deflection during second vision loop.

Table 7.1: Desired heading rate calculation for aileron

Component	Original Controller	Modified Controller
	 Figure 7.15: Original controller logic.	 Figure 7.16: Modified controller logic.
Base heading rate	$\frac{V_{IAS}}{R}$ −required to hold current radius	$\frac{V_{GS}}{R_{des}}$ −required to hold desired radius
Heading rate error	Defined as an additional heading rate required to get to desired radius	Defined as a difference from a constant average heading rate required to hold desired radius at a constant average V_{GS}

originate from the incorrect sensor calibrations, and noise in video transmissions. The dynamical errors can be caused by the wrong attitude measurement due to the acceleration, or loss of target tracking due to the camera vibrations. The errors increase when the tilt is closer to the horizon and the target is located in the upper and side regions of the image as shown in section 3.4.

Table 7.2: Error Analysis.

Error type	Error source
Radius Estimates has a constant relative error	Height measurements are incorrect
Radius Estimates incorrect closer to image corners	Lens distortion causes wrong coordinates measured
Radius Estimates accuracy fluctuates	Target may get into image regions with higher error multiplier, or tilt is too high
Radius Estimates accuracy degrades on sudden motions	AHRS attitude measurements are incorrect due to rapid changes in orientation

Chapter 8

CONCLUSIONS

In this project, a single camera vision-based navigation system for UAV's was built and successfully tested. This system allows for the operation of an aircraft along circular orbits without using GPS.

This system includes a single camera installed on a gimbal, a barometric altimeter, a computer for data processing, and an aircraft controller. The image from the camera is processed using Matlab. The aircraft is controlled with a custom ArduPlane mode. All necessary algorithms have been developed and codes have been written: the image rectification, the orbit controller, and connections between Matlab and MissionPlaner.

The test results demonstrated satisfactory performance of the proposed vision system.

The on-ground tests showed that the average error does not exceed 3%. The peak error could be as high as 9%. Most of this error comes from the asymmetrical image distortion in the upper corners.

During the flight tests, measurements were mostly accurate. However, strong wind caused the vision radius error to occasionally increase up to 20-40%. After these large errors, the recovery time may could be up to 15 s. Using higher precision components can help to reduce the error and the recovery time. For maximum precision, tilt should be closer to straight down, and the target should be in the center of the image towards the bottom.

The circular orbit controller was designed to follow the vision radius guidance. The controller is prioritizing safety. The controller is able to track a circular trajectory, but still has room for improvement in windy conditions.

Future work is to move image processing onboard to remove delay, change the rudder loop to have a turn coordinator, and add the optical flow ground speed sensor.

For debugging purposes, the image processing during the flight tests was done by the on-ground computer. Communication between the aircraft and the on-ground computer was done using radio channels. However, this system can be installed completely on the plane, thus eliminating the need for the radio connection between the plane and the ground. It is as also possible to modify this system to handle more complicated trajectories.

This system can find use in a number of applications that require operation in GPS-denied conditions. It is also useful as a backup navigation system for other types of aircraft.

BIBLIOGRAPHY

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

[2] Image acquisition toolbox support package for os generic video interface. Matlab File Exchange. https://www.mathworks.com/matlabcentral/fileexchange/45183-image-acquisition-toolbox-support-package-for-os-generic-video-interface.

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

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

[5] Integrated learning framework for pedestrian tracking. GitHub ILFPT. https://github.com/Prinsphield/ILFPT.

[6] A. G. Edmundo Guerra, Rodrigo Mungua. Uav visual and laser sensors fusion for detection and positioning in industrial applications. *MDPI Sensors*, 18, 2018.

[7] R. J. Grimes. Visual anchoring: Fixed-wing uas orbit stabilization about a visual anchor point without gps dependence. Master's thesis, University of Washington, Seattle, WA, June 2018.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

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

[13] C. W. Lum, H. Rotta, R. Patel, H. Kuni, T. Patana-anake, J. Longhurst, and K. Chen. Uas operation and navigation in gps-denied environments using multilateration of aviation transponders. In *Proceedings of the AIAA SciTech 2019 Forum*, San Diego, CA, January 2019.

[14] M. G. Misulia. A derivation for the image displacement due to tilt. *U. S. Coast and Geodetic Survey*, 34:461–463, December 1946.

[15] R. J. D. Moore, S. Thurrowgood, D. Bland, D. Soccol, and M. V. Srinivasan. Vulnerability assessment of the transportation infrastructure relying on the global positioning system. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, October 2009. IEEE.

[16] B. E. W. Paul R. Wolf, Bon A. Dewitt. *Elements of Photogrammetry with Applications in GIS*. McGraw-Hill Education, 4st edition, 1998.

[17] 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.

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