
Theses and Dissertations

Spring 2010

Design and test of a multi-camera based orthorectified airborne imaging system

Nicole Lynn Becklinger
University of Iowa

Follow this and additional works at: <https://ir.uiowa.edu/etd>

 Part of the [Industrial Engineering Commons](#)


Copyright 2010 Nicole Lynn Becklinger

This thesis is available at Iowa Research Online: <https://ir.uiowa.edu/etd/461>

Recommended Citation

Becklinger, Nicole Lynn. "Design and test of a multi-camera based orthorectified airborne imaging system." MS (Master of Science) thesis, University of Iowa, 2010.
<https://doi.org/10.17077/etd.he3h1a5v>

Follow this and additional works at: <https://ir.uiowa.edu/etd>

 Part of the [Industrial Engineering Commons](#)

DESIGN AND TEST OF A MULTI-CAMERA BASED ORTHORECTIFIED AIRBORNE IMAGING SYSTEM

by

Nicole Lynn Becklinger

A thesis submitted in partial fulfillment of the requirements for the Master of Science degree in Industrial Engineering in the Graduate College of The University of Iowa

May 2010

Thesis Supervisor: Professor Thomas Schnell

Graduate College
The University of Iowa
Iowa City, Iowa

CERTIFICATE OF APPROVAL

MASTER'S THESIS

This is to certify that the Master's thesis of

Nicole Lynn Becklinger

has been approved by the Examining Committee for the thesis requirement for the Master of Science degree in Industrial Engineering at the May 2010 graduation.

Thesis Committee:

Thomas Schnell, Thesis Supervisor

Geb Thomas

Andrew Kusiak

ACKNOWLEDGEMENTS

First, I would like to thank my advisor Professor Tom Schnell, the University of Iowa, and Yang Zhu and Tim Ethrington of Rockwell Collins for making this project possible. Yang in particular has put in a lot of time, effort, and road trips from Minnesota to get the imaging software up and running. I would like to thank my coworkers at the Operator Performance Laboratory for their help with flight testing, construction, and coding. I thank my friends and family for all their support, and for putting up with my thesis writing obsession for the past few months. A special thanks goes to my dad for teaching me how to design and build things.

TABLE OF CONTENTS

LIST OF TABLES	vi
LIST OF FIGURES.....	vii
LIST OF EQUATIONS	x
LIST OF ACRONYMS.....	xii
CHAPTER 1. INTRODUCTION	1
1.1 Problem Statement	1
1.2 Solution Approach.....	2
CHAPTER 2. BACKGROUND AND LITERATURE	5
2.1 Properties of Remotely Sensed Images	5
2.2 Remote Imaging Platforms.....	7
2.3 Limitations of Current Airborne Imaging Systems.....	10
2.4 KML Programming and Google Earth.....	12
CHAPTER 3. OAI SYSTEM OVERVIEW	14
3.1 Airborne components	14
3.1.1 Imaging cameras	15
3.1.2 Phidget and Camera Triggering Circuit Board	18
3.1.3 Elma Rack Computer	19
3.1.5 Pan Tilt Units	20
3.1.6 Airborne Control Computer.....	21
3.1.4 GPS and IMU Systems	22
3.1.5 Antennas	23
3.1.7 Scene camera	25
3.2 Ground Station Components	25
3.2.1 Ground Station Pan Tilt	26
3.2.2 Antennas	27
3.2.3 Control Computer	27
3.2.4 Image Server	28
3.2.5 Image Processing Computer	28
CHAPTER 4. HARDWARE DEVELOPMENT	29
4.1 Design of Camera Head	31

4.1.1 Concept Designs	31
4.1.2 Prototype Phase.....	35
4.1.3 Final Design	36
4.2 Design of Other Components.....	38
4.2.1 Floor Raiser Design	38
4.2.2 Front Equipment Rack Design.....	41
4.2.3 Aft Equipment Rack Design	42
CHAPTER 5. SOFTWARE	45
5.1 Image Processing Software	45
5.2 KML Super Overlay.....	47
5.2.1 Image Pyramid.....	47
5.2.2 Procedure for Creating a KML Super Overlay	49
5.3 KML Super Overlay Generator Program and User Interface	54
CHAPTER 6. CALIBRATION AND DATA COLLECTION.....	64
6.1 Camera Calibration	64
6.2 Data Collection.....	69
6.2.1 Airborne Components Procedures	69
6.2.2 Ground Station Procedures	75
6.3 Overview of Data Flow	77
CHAPTER 7. IMAGING RESULTS	79
7.1 Image Resolution as a Function of Altitude.....	79
7.2 Placement Accuracy.....	87
CHAPTER 8. CONCLUSIONS AND FUTURE WORK	93
8.1 Conclusions	93
8.2 Future Work	94
8.2.1 The Next Phase of OAI Development	94
8.2.2 Future Applications of the OAI System.....	95
8.3 Closing Remarks	96
CHAPTER 9. WORKS CITED	97
APPENDIX A:KML SUPREOVERLAY CODE EXAMPLES.....	101
A.1 Example Top Level File for Super Overlay	101
A.2 Example of Mid-Level KML Super Overlay File.....	102
A.3 Example of Level 1 KML File	105

LIST OF TABLES

Table 1: Accuracies of IMU/SPAN system (Novatel)	23
Table 2: Pixel Size of Image Based on Altitude and Camera Position Angle	83
Table 3: Distance of Near and Far Edges of Image Area from Aircraft at Different Altitudes and Camera Head Position Angles	86
Table 4: Image Placement Accuracy	91

LIST OF FIGURES

Figure 1: ProE Model of Hardware.....	3
Figure 2: Diagram of Airborne components	15
Figure 3: Camera Overlap Horizontal.....	17
Figure 4: Camera Overlap Vertical	17
Figure 5: Camera Triggering Process.....	19
Figure 6: Ground Station Components.....	26
Figure 7: Complete Hardware Concept Design	30
Figure 8: Picture of OAI System Installed in Aircraft	30
Figure 9: Evolution of Camera Enclosure Concept Design: A) Wedge B) Joinable Wedge C) Long Box D) Slotted Box E) Short Box F) Short Slotted Box (Final Design, Shown with Camera).....	33
Figure 10: Camera Head with Plate Design	34
Figure 11: Camera head with Wedge Design.....	34
Figure 12: Camera Head Prototypes	35
Figure 13: IMU Mounting Plate Prototypes.....	36
Figure 14: Finished Camera Head Top View.....	37
Figure 15: Finished Camera Head Side View	37
Figure 16: Finished Camera Head Front View	38
Figure 17: Floor Raiser Concept Designs	40
Figure 18: Finished Floor Raiser.....	40
Figure 19: Concept Design for Front Equipment Rack.....	41
Figure 20: Front Equipment Rack.....	42

Figure 21: Concept Design of Aft Equipment Rack	44
Figure 22: Completed Aft Rack with Camera Pan Tilt Attached.....	44
Figure 23: Example of a finished OAI image	46
Figure 24: The First Three levels of an Image Pyramid. Squares containing no part of the original image (marked by a red X) are not created by the image pyramid software ..	48
Figure 25: KML Super Overlay Generator GUI	55
Figure 26: Image Pyramid Options Section of KML Super Overlay Generator GUI.....	56
Figure 27: Help Window for the "Why Use .png Format?" Link	57
Figure 28: Image Pyramid Options Section with "Use .png File Format" Option Active	59
Figure 29: Help Window for the "How Do I Use Transparencies?" Link	59
Figure 30: Coordinate Input Options Section.....	60
Figure 31: Help Window for the "What is a .geo File? Link	61
Figure 32: Help Window for the "How Do I Enter Latitude and Longitude?" Link.....	62
Figure 33: KML File Options Section of the KML Super Overlay Generator Interface	63
Figure 34: Camera Calibration Setup.....	65
Figure 35: Camera Calibration Board	65
Figure 36: Example of Calibration Image Set with Six Imaging Cameras and a 7th Calibration Camera	66
Figure 37: Calibration Scan Grid: Each Dot Represents a Point in the Scan Grid. Units are in Steps with One Step Equal to 5 Degrees of PTU Rotation	68
Figure 38: Pre-Flight Setup with Ground Power Cart Visible	72
Figure 39: Steps in the image collection process	78
Figure 40: Geometry of Image Capture	80
Figure 41: Image Resolution Based on Altitude and Camera Head Position Angle.....	81
Figure 42: Illustration of a Situation Where Camera View Angle (a) Added to Camera Head Position Angle (b) is Equal to 90 Degrees. The Far Edge of the Imaging Field Never Intersects the Ground.....	82

Figure 43: Distance of Near (Bottom Lines) and Far (Top Lines) Edges of Image Area Based on Altitude and Camera Head Position Angle85

Figure 44: Screen Shot from Google Earth Showing Google Earth Overlays on OAI Image87

Figure 45: GCPs Used in the Calculation of Image Placement Accuracy. Each GCP is marked with a Red X.....89

LIST OF EQUATIONS

Equation 1: Pixel size (P) as a function of field of view (a) and altitude (H)	6
Equation 2: Number of tiles (T_N) in a Given Level (l) of an Image Pyramid	48
Equation 3: Degrees Latitude Per Pixel (Lat_p) in Terms of the Northern Boundary of the Original Image (N_o), Southern boundary of the Original Image (S_o), and Height in Pixels of the Original Image (h_o).....	49
Equation 4: Degrees Longitude Per Pixel (Lon_p) in Terms of the Eastern Boundary of the Original Image (E_o), Western Boundary of the Original Image (W_o), and Width in Pixels of the Original Image (w_o).....	50
Equation 5: Calculation of Height (h_n) and Width (w_n) in Pixels of the Image Composed of the Highest Resolution Image Pyramid Tiles Based on the Number of Levels in the Image Pyramid (L_{max})	50
Equation 6: Southern Coordinate of the New Image (S_n) in Terms of the Northern Coordinate of the New Image (N_n), Latitude per Pixel (Lat_p), and Height of the New Image in Pixels (h_n)	51
Equation 7: Eastern Coordinate of the New Image (E_n) in Terms of the Western Coordinate of the New Image (W_n), Longitude Per Pixel (Lon_p), and Width of the New Image (w_n)	51
Equation 8: Calculation of the Northern Coordinate of Tile “im名称_l_m_n” (N_{lmn}) Based on the Northern Coordinate of the Total Image (N_n), the Southern Coordinate of the Total Image (S_n), Level in the Image Pyramid (l), and the Tile's North-South Position (m)	52
Equation 9: Calculation of the Southern Coordinate of Tile “im名称_l_m_n” (S_{lmn}) Based on the Northern Coordinate of the Total Image (N_n), the Southern Coordinate of the Total Image (S_n), Level in the Image Pyramid (l), and the Tile's North-South Position (m)	52
Equation 10: Calculation of the Eastern Coordinate of Tile “im名称_l_m_n” (E_{lmn}) Based on the Eastern Coordinate of the Total Image (E_n), the Western Coordinate of the Total Image (W_n), Level in the Image Pyramid (l), and the Tile's East-West Position (n)	52
Equation 11: Calculation of the Western Coordinate of Tile “im名称_l_m_n” (W_{lmn}) Based on the Eastern Coordinate of the Total Image (E_n), the Western Coordinate of the Total Image (W_n), Level in the Image Pyramid (l), and the Tile's East-West Position (n)	52
Equation 12: The Four Child Files (C_1, C_2, C_3, C_4) for the Parent File Corresponding to “im名称_l_m_n”	54

- Equation 13: Calculation of Pixel Size (s_p) Based on Altitude (H) Camera Head View Angle
(a) Position Angle of the Camera Head (b) and Number of Pixels (N_p) or
Alternatively in Terms of the Ground Distance Imaged by the Camera Head (D)..80
- Equation 14: Calculation of the Location of the Near Edge of the Image Relative to the Aircraft
(l) Based on Altitude (H) and Camera Head Position Angle (b).....84
- Equation 15: Calculation of the Location of the Far Edge of the Image Relative to the Aircraft (L)
Based on Altitude (H), Camera Head Field of View (a) and Camera Head Position
Angle (b)84

LIST OF ACRONYMS

Acronym	Meaning
AES	Advanced Encryption Standard
ATR	Air Transport Radio
CMOS	Complementary Metal-Oxide-Semiconductor
CNC	Computer Numerically Controlled
GCP	Ground Control Point
GEO	Geographic
GPS	Global Positioning System
GUI	Graphical User Interface
IMU	Inertial Measurement Unit
JPEG	Joint Photographic Experts Group
KML	Keyhole Markup Language
LCD	Liquid Crystal Display
LOD	Level of Detail
NASA	National Aeronautics and Space Administration
NASA-JPL	National Aeronautics and Space Administration Jet Propulsion Laboratory
OAI	Orthorectified Airborne Imaging
OPL	Operator Performance Laboratory
PNG	Portable Network Graphic
ProE	Pro Engineer Software
PTP	Point to Point
PTU	Pan Tilt Unit
SFB	Synthetic Flight Bag

SPAN	Synchronized Position Attitude Navigation
UAS	Unmanned Aircraft System
UAV	Unmanned Aerial Vehicle
UCL	University College London
URL	Uniform Resource Locator
USB	Universal Serial Bus
WAAS	Wide Area Augmentation System
XML	Extensive Markup Language

CHAPTER 1. INTRODUCTION

1.1 Problem Statement

This project was performed in order to develop an airborne imaging toolset for use in a range of 2 dimensional orthorectified imaging applications. The imaging equipment used is a 6 camera airborne imaging system and ground station provided by Rockwell Collins. Hardware designed and constructed by the University of Iowa's Operator Performance Laboratory (OPL) was used to improve functionality of the imaging system and allow it to be installed in the OPL's Beechcraft Bonanza. Software provided by Rockwell Collins and NASA Jet Propulsion Laboratory (NASA JPL) allowed for the six images produced in a single frame to be stitched into a single image, cropped, and rotated. This software results in wide area images that are orientated such that north is always up on the user's screen regardless of the orientation at which the pictures were taken. Flight tests were performed to test the Orthorectified Airborne Imaging (OAI) system. Additional software created by the author allows stitched representations of an area of interest to be exported to Google Earth for viewing.

The OAI system presented in this report is intended to enhance the situational awareness of the user in a variety of situations. The combination of recent orthorectified images and the geographic tools found in Google Earth allow the user to more easily understand the content of each image. The six camera system allows a greater area to be covered than in a single image, allowing the user to get more information about a situation from a single frame. While this system does not

produce results that are truly real time, the finished images can be viewed and uploaded in a matter of seconds while the aircraft is still in the air. Even though the OAI system does not provide images that are truly real time, users are able to examine images while the aircraft is still in the air. Areas of interest can be identified and examined more closely by having the aircraft perform additional imaging passes. The OAI system can be controlled remotely, allowing for future implementation in unmanned vehicles.

1.2 Solution Approach

The first stage of the project was to alter the existing imaging system to fit the aircraft and to improve the design of the camera head. This required the design and construction of a new camera head, two equipment racks to contain airborne hardware, and floor raisers for the aircraft. A screen shot of a ProEngineer model of the hardware created for this project can be seen in Figure 1. When the design and construction of equipment was completed, the imaging equipment was installed in the OPL's 1973 Beechcraft Bonanza. The imaging equipment in the aircraft was able to communicate with ground station equipment and transmit images via two antennas. The first, a Motorola canopy antenna, transmitted flight path information and allowed the rotating head of the ground station to track the aircraft in order to maintain the data link. A second Motorola point to point connection antenna was used to transmit images from the aircraft to the ground and to control and monitor the airborne equipment from the ground station.

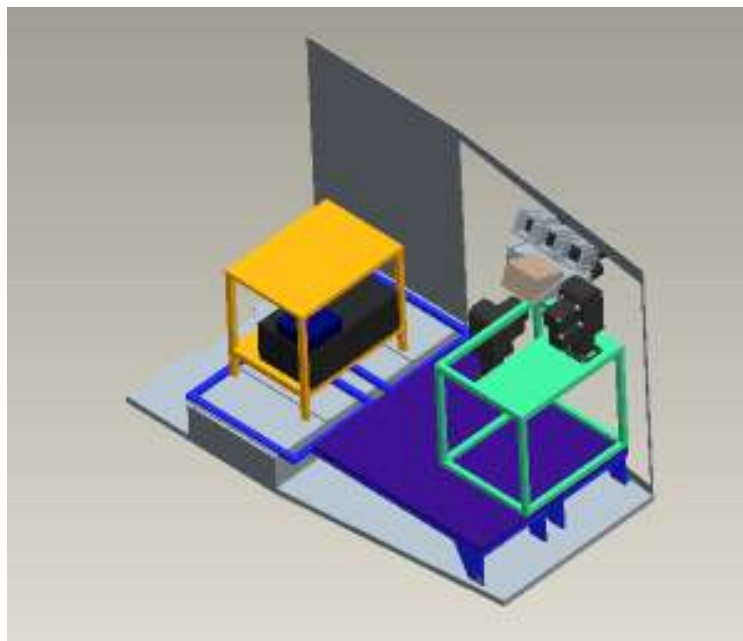


Figure 1: ProE Model of Hardware

Several programs were used to process the images. Software produced by Rockwell Collins and the National Aeronautics and Space Administration Jet Propulsion Laboratory (NASA-JPL) was used to create a data file with Global Positioning System (GPS) location and frame number for each image, stitch each 6 image set, rotate the composite image to north-up orientation, and add a border around the composite image. A Keyhole Markup Language (KML) super overlay generator program created by the author used the position data from the stitched images to position them Google Earth. A Graphical User Interface (GUI) was created to automate the creation of the KML super overlay and provide the user with options for creating, displaying, and storing the KML super overlay.

Data collection was performed during a series of flight tests taking place between March 2009 and August 2009. Early flight tests were intended to check equipment functionality. As flight testing progressed, a steady rate of processed images was

achieved. Flight tests were made over the Iowa City Airport and surrounding areas. This allowed for a variety of natural and man-made features to be recorded.

Success of the OAI system can be evaluated by its ability to acquire and process images in near real time. Analysis of the field of view of images allows the expected image resolution at a given imaging altitude to be calculated. Measurement of the accuracy of image placement relative to the Google earth coordinate system and real world ground reference points can be used to assess the accuracy of image placement.

CHAPTER 2. BACKGROUND AND LITERATURE

The first airborne images were taken by a French photographer using a hot air balloon in 1868 (Professional Aerial Photographers Association, 2009). The first systematic use of remote imaging was the use of aerial photography for performing battle damage assessment during World War I (Finnegan, 2006). Photo surveying for military, agricultural, and urban planning purposes gained popularity in the 1930's and 1940's (Committee on Science and Technology, 2008). The late 1960's and early 1970's marked the beginning of satellite photography with the NASA Landsat satellite observatory launched in 1972 (National Aeronautics and Space Administration, 2009). The Land Remote Sensing Policy Act of 1992 ensured public access to Landsat data and brought about licensing procedures for commercial remote sensing operations using satellites (United States Congress, 1992). More recently, Unmanned Aerial Vehicles (UAVs) have come into use for many surveillance, monitoring, and mapping tasks (LaFranchi, 2006), (AT Electronic and Communication International, 2009), (AT Electronic and Communication International, 2009).

2.1 Properties of Remotely Sensed Images

In most modern airborne imaging systems, images are digital in nature though a few analog systems are still in use (NASA, 2008), (Gao, 2009). Digital images are composed of a grid of discrete points called pixels. Each pixel has a number or numbers associated with it that give the value for the color or brightness of light associated with that point in the picture grid. Pixels represent a limitation on the resolution of an image. While pixels may be interpolated, or duplicated to make an image take up more space, it is not possible to obtain additional information by doing so. An important consideration

when determining the suitability of an imaging method is the pixel size, or the amount of area on the actual surface represented by a single pixel. Resolution of an airborne image can be calculated as a function of altitude and the instantaneous field of view according to Equation 1 below. Spatial resolution, which refers to the dimensions of the smallest ground features that can be identified in an image, can be considered along with or instead of pixel size. Since finer resolution and smaller pixel size tend to be more costly in terms of data storage, equipment, or purchasing image data, it is often advantageous to use the largest pixel size and coarsest resolution that can be used for a particular application (Gao, 2009).

$$P=a*H$$

Equation 1: Pixel size (P) as a function of field of view (a) and altitude (H)

The number and range of wavelengths recorded is another important factor when considering an airborne imaging platform. While many sensors record only in the visible spectrum, it is also possible to record data in other wavelengths. For example, the oceanographic satellite SeaWiFS uses near infrared wavelengths to for measuring aerosol radiance (National Aeronautics and Space Administration, 2008). Choosing the number and range of channels to measure can be difficult. Having channels with too similar of a spectral profile or dividing the spectrum into bands that are too narrow can result in data redundancy. Absorption of some wavelengths by ground surface features or the atmosphere can further complicate data collection (Gao, 2009).

In applications where position data accompanies the image, several methods of referencing can be used. One method is to choose Ground Control Points (GCPs) in the image and match them with coordinates of the actual objects using either a topographical map or by taking a portable GPS device to each location and measuring the coordinates. This method can be effective for areas with well-defined GCPs that are readily accessible, but less effective in difficult to access areas or areas with few identifiable features. Another option is to measure the location of the sensor using a GPS device, then use the mathematical relationship between the sensor, GPS unit, and the surface to calculate the coordinates of the image. The accuracy of this method is highly dependent on the accuracy of the GPS unit used (Gao, 2009) .

2.2 Remote Imaging Platforms

Remote imaging systems can be classified in many ways, such as sensor type, primary purpose, resolution, and type of vehicle carrying the imaging system. In this thesis, remote imaging systems will be grouped according to the type of vehicle used. Vehicle type is a useful way to group remote imaging systems because almost all airborne remote imaging systems can be described as being part of a manned aircraft, a UAV or a satellite.

Manned aircraft such as airplanes, blimps, and helicopters have been used for a variety of airborne imaging applications since 1858 (Professional Aerial Photographers Association, 2009). Today, many aerial photography companies operate in the United States, taking pictures of property for mapping, construction site monitoring, and aesthetic purposes (Aerial Photography Services, 2008), (Mazzone, 2009), (Fly By Photos, 2006). Remote sensing devices including radar, optic sensors, and infrared

sensors have also been used by the military for intelligence operations, command and control, surveillance, reconnaissance, and target tracking (Northrop Grumman, 2010). A particularly noteworthy system is the Northrop Grumman Night Hunter II system, which uses 5-6 sensors in conjunction with target identification and tracking software for passive imaging, tracking, and weapons targeting (Northrop Grumman, 2010). Airborne surveillance is also available outside the military through private companies (AME Info, 2005), (Airborne Surveillance, 2010). These companies provide surveillance services to groups that might not have continuous need for an airborne surveillance platform such as law enforcement agencies, private companies, and surveyors.

Unmanned Air Vehicles (UAVs) have also been used extensively for airborne imaging. In agriculture, UAVs with remote sensors have been used to monitor crops, property and livestock (AT Electronic and Communication International, 2009), (LaFranchi, 2006). Similarly, UAV based remote imaging systems have been used for monitoring oil pipelines, property, and in law enforcement (AT Electronic and Communication International, 2009). UAVs have also been implemented as a replacement for manned air vehicles in search and rescue operations (Rasmussen, Morse, & Taylor, 2007). Infrared and visible spectrum UAV sensors have also been used for monitoring fires (Esposito, Rugino, & Moccia, 2009), (Esposito, Rufino, & Moccia, 2007), (Sausser, 2007). A well publicized example occurred in August 2007 when a 12 channel spectral sensor developed by NASA was used to help pinpoint hotspots during a wildfire in Santa Barbra County California. Thermal images taken by the sensor were overlaid on three-dimensional terrain inside Google Earth. The infrared sensor made it

possible to “see” through the clouds of smoke to pinpoint current hotspots and damaged areas based on temperature (Sauser, 2007).

The 2009 United States Air Force Unmanned Systems Flight Plan groups Air Force UAVs into three classes based on size. The small Unmanned Aircraft System (UAS) category as of 2009 includes the Wasp III, RQ-11 Raven, and the Scan Eagle. These hand or catapult launched systems contain electro-optical and or infrared cameras and can be flown manually or autonomously using GPS. Their maximum range is from 3 miles in the case of Wasp III to 68 miles in the case of Scan Eagle (United States Air Force, 2009). The primary mission for all of these small UAS systems is to provide real time situational awareness and surveillance in daytime or nighttime conditions. The next category is medium UAS, which contains the MQ-1 Predator and the MQ-9 Reaper. In addition to optical and infrared cameras, both of these UAVs are equipped with laser target markers and laser illuminators. The MQ-9 Reaper is also equipped with hard points allowing for the attachment of various weapons. The Predator is used primarily for intelligence, surveillance, reconnaissance, combat search and rescue, and overwatch tasks. The Reaper is used primarily for persistent strike, but can also be used for any of the tasks more closely associated with the Predator. Both aircraft can be flown with a range of approximately 100 miles within line of sight, and can be flown beyond line of sight using satellites. The only Air Force aircraft in the large UAS category is the RQ-4 Global hawk. Several variations of the Global Hawk are available, and the sensors used vary depending on the model. Some are very similar in sensor layout to the Predator, while others have an additional Battlefield Airborne Communications Node, Enhanced

Integrated Sensor Suite, or Multi-Platform Radar Technology Insertion (United States Air Force, 2009).

Satellite based remote imaging platforms have provided imaging data since the late 1960's. Some satellites are operated by government agencies others are operated commercially. The quality and type of data available varies depending on the individual satellite. Common spectra recorded are visible, infrared, and near infrared. Satellite based platforms can be grouped roughly by the resolution of the data they produce. Low-resolution imagery has resolution ranging from a few kilometers to 5 or 10 meters (Gao, 2009). High-resolution satellite imagery has resolution ranging from a few meters (Gao, 2009) to 0.41 meters in the case of the GeoEye-1 satellite (GeoEye, 2010). Some satellite data is available free to the public, either through free downloads or earth viewers such as Google Earth and NASA WorldWind. High-resolution data produced by commercial satellites are sold by region. In some cases, processed data is also sold. Some companies offer data that have been corrected for distortions and noise. Others offer consumer-specified data processing (Gao, 2009).

2.3 Limitations of Current Airborne Imaging Systems

A summary of the limitations of current airborne imaging systems can be found in the record for the One Hundred Tenth Congress's field hearing on the applications and benefits of remote sensing data (Comittee on Science and Technology, 2008). In this document the viewpoints of many different groups that use remotely sensed imagery were presented. Even though these groups used imaging for such varied applications as geological surveying, infrastructure planning and firefighting among others, they had similar thoughts on the shortcomings of current remote imaging technology.

One of the major challenges facing remote imaging is the ability to provide local governments and organizations access to images that are high resolution, recent, and cost effective. While global satellite projects such as ResMap provide downloads of satellite image data at no cost, these images are often either too old or have resolution that is too low to be of practical use. Hiring a professional aerial photographer or commercial satellite service is such a financial burden for some communities that they rely on maps sketched by hand in small aircraft or by hiring a satellite once every five years (Committee on Science and Technology, 2008). Even a \$7,000 UAV crop monitoring system (LaFranchi, 2006) can be prohibitively expensive for a civilian user. Programs sponsored by national and state governments help by sponsoring satellite programs and dispatching military resources for wild fire observation (Sausser, 2007), (Kumar & Cohen, 2009). However, remote imaging resources for local government use seem to be spread too thinly to provide adequate data. In the case of wildfires, old images might be useful for prevention strategies, but do not provide the information needed to manage a fire in an emergency situation (Committee on Science and Technology, 2008). In August 2007, near real time imagery from satellites and UAVs were used to help combat a wildfire in southern California (Sausser, 2007). At the 2008 field hearing, one group's representative stated that while technology developed for military use showed great potential in fighting large wildfires such as those occurring in August of 2007, the technology might prove more useful if it was made available to local agencies (Committee on Science and Technology, 2008).

Another challenge faced by many of the groups represented at the hearing was storing and interpreting the data. Many communities who wanted data for a small region

were only able to acquire datasets covering a huge area. Several groups representing collectives of small communities expressed concern over the sheer amount of data being stored, and the difficulty of locating data subsets. Trying to isolate data from a specific area over several years of data sets for the purpose of tracking urban expansion over time seemed to be a particularly difficult task, as was the cost of hiring and training personnel to process the data (Committee on Science and Technology, 2008). Proper interpretation of images can also be difficult due to inaccuracies in the location or orientation of an image within the global coordinate system. Distortions can be caused by atmospheric conditions, position of the sensor relative to the earth, and errors within sensors. Software used to interpret images does not always present a clear indication of the accuracy of results, and can easily produce results that are completely inaccurate if the software is used incorrectly (Gao, 2009).

2.4 KML Programming and Google Earth

The Keyhole Markup Language (KML) was created by the Keyhole Company in 2001 as a data format for an earth browser called Earth Viewer. It has since become a standard language used for the visual presentation of geographic information. KML is an Extensible Markup Language (XML) data format. It is used to display geographic information for a variety of earth viewers as well as in other programs such as AutoCAD and Adobe Photoshop (Werneck, 2009).

One well known earth viewers that uses KML is Google Earth. The precursor to Google Earth was developed by Keyhole. The concept was further developed by Google after Google purchased Keyhole in October 2004. The first complete version of Google Earth was released in June 2005 (Google, 2010). As of January 2010 three versions of

Google Earth are available. Google Earth 5.0 is a free downloadable version featuring satellite imagery, historical images, three dimensional buildings, and other features.

Google Earth Pro and Google Earth Enterprise are purchasable versions of Google Earth that contain additional software for corporate users (Google, 2010). All versions of Google Earth contain satellite imagery of the Earth's surface, with varying resolution and age of images. In all versions, overlays showing roads, place names, weather, points of interest, and other features can be displayed by selecting them from a menu (Google, 2009). Additional user-created features, tours, and three dimensional structures are available online (Werneck, 2009).

Using KML and Google Earth together, it is possible to create a range of objects and overlays. Using either Google Earth menus or direct KML coding, it is possible to create points of interest with webpage-like descriptions attached to a certain latitude and longitude. Points of interest can be linked to create virtual tours within Google Earth. Picture overlays can be placed over land surfaces. Overlays can serve many purposes depending on the picture used. A series of old aerial photos can be digitized and overlaid on an area one at a time with a time delay between frames in order to show the evolution of an area over time. Likewise, real time or near real time images or overlays can be used to show weather patterns or cloud cover. Three-dimensional features can also be imported from Google SketchUp or other three-dimensional modeling programs. When KML code is created either directly or through Google Earth's menus, it is stored locally, but can be published to a website or server allowing others to view added features (Werneck, 2009).

CHAPTER 3. OAI SYSTEM OVERVIEW

This chapter is intended to give an overview of the electronic components of the OAI system. Components are categorized as belonging to either the airborne portion of the OAI system or to the ground station portion of the system. Specifications and a brief description of the functions of each component are given.

3.1 Airborne components

The components listed in this section were included in the airborne portion of the OAI system. During flight testing, the components in this section were installed in the OPL's Beechcraft Bonanza research aircraft using the hardware described in Chapter 4. A system diagram of the airborne portion of the OAI system can be seen in Figure 2.

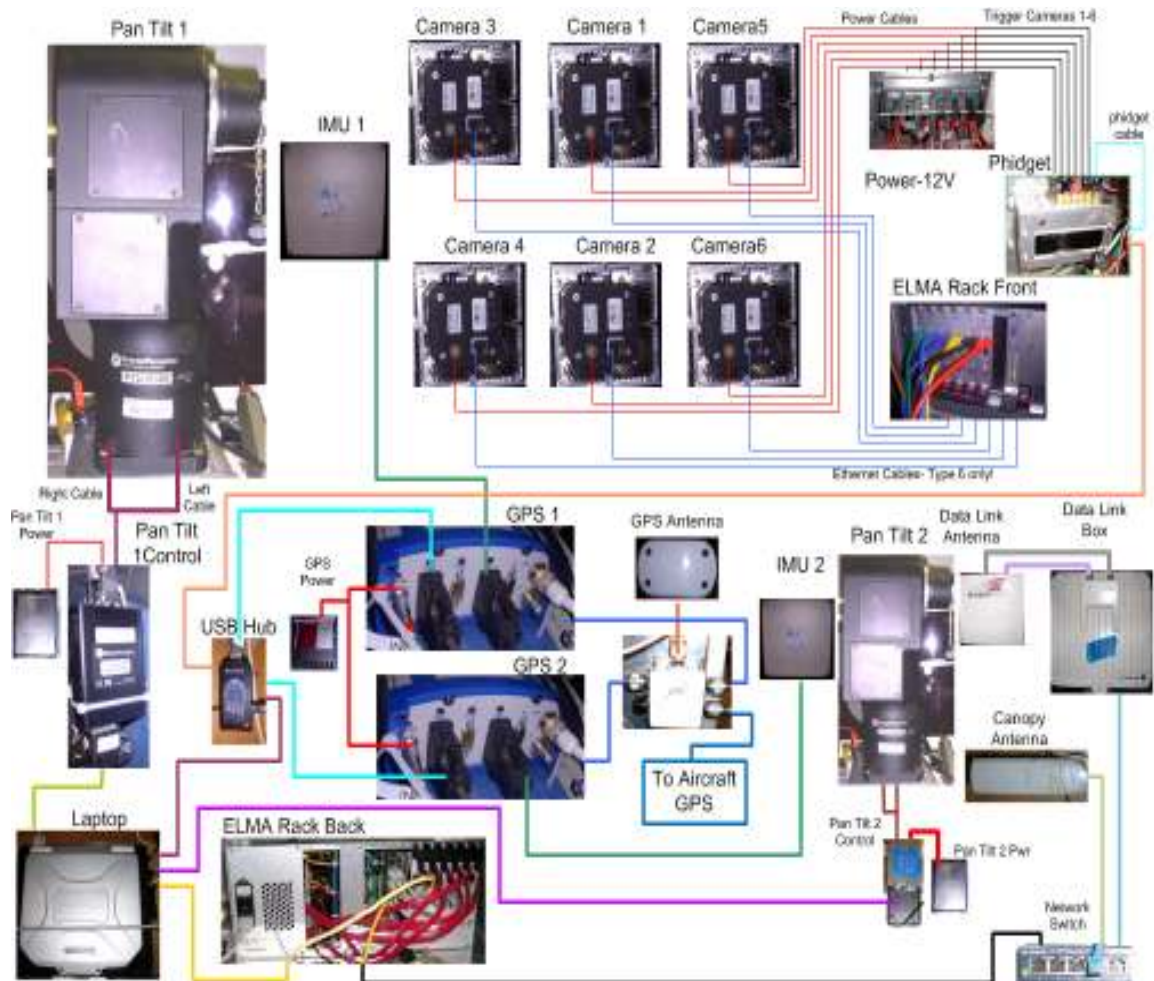


Figure 2: Diagram of Airborne components

3.1.1 Imaging cameras

Six Illunis XMV-16000 16M pixel imaging cameras were used in the imaging head. Each camera records in the visual spectrum using a Kodak KAI-16000 sensor. Cameras were triggered by their power/trigger cable which was attached to the Phidget device as seen in Figure 2. The cameras are not capable of storing images. Instead images were exported to the ELMA computer via Cat-6 Ethernet cables. The sensing

area of each camera is 37.25 mm by 25.7 mm, resulting in images that are 4,872(H) by 3,248(V) pixels. The pixel size inside the camera is 7.4um by 7.4um (Illunis LLC) .

The purpose of the imaging cameras was to record visual spectrum images of the area below the aircraft. Each of the six cameras was installed into a camera mount which was attached to the camera head. The camera mounts positioned the cameras such that there was an angle of seven degrees between cameras in the horizontal direction and a twenty two degree angle between the top and bottom rows. The overlap created by angling the camera positions prevents blank spots from occurring at the seams of the six small images when they are stitched together to form a large image. Development of the camera head is described in detail in Chapter 4. The camera head was attached to a Pan Tilt Unit (PTU), which is described in detail later in this section. Recorded images were saved and processed at the Elma brand computer, then transmitted to the ground station via the data link antenna. A backup copy of each image remained on the EMLA computer's hard drives and could be retrieved post-flight if the image was not transmitted.

While the camera head and cameras are designed for portrait and landscape camera orientations, portrait orientation was used for the duration of this project. The pattern of camera viewing area overlap can be seen in Figure 3 and Figure 4. Figure 3 shows the horizontal overlap for the top row cameras 1, 3 and 5. The same configuration is used for the bottom row cameras 2, 4, and 6. Figure 4 shows the vertical overlap between the top and bottom rows.

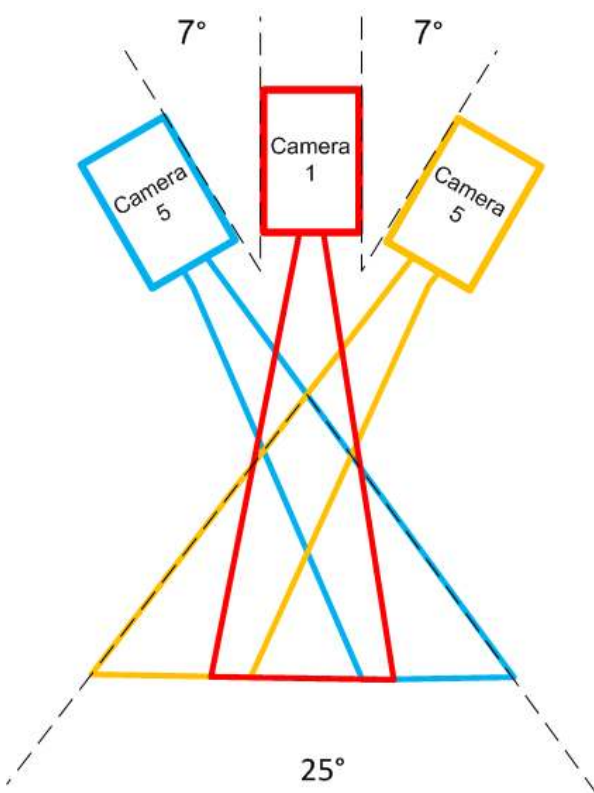


Figure 3: Camera Overlap Horizontal

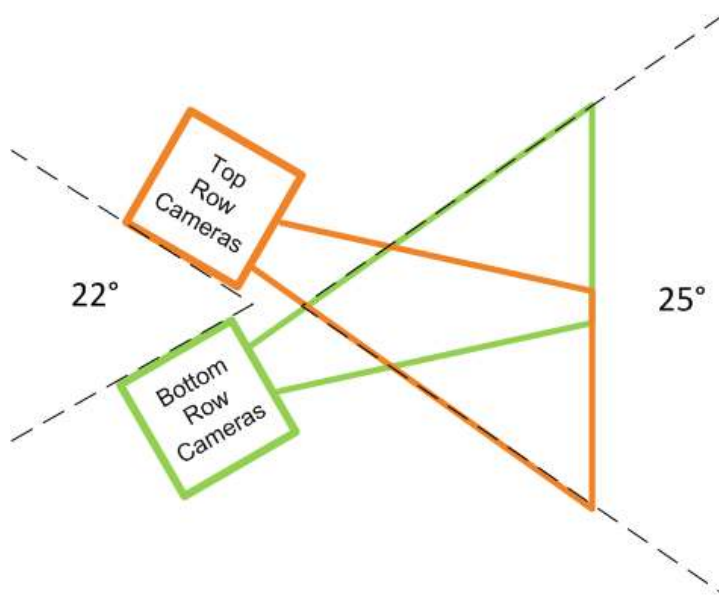


Figure 4: Camera Overlap Vertical

3.1.2 Phidget and Camera Triggering Circuit Board

A Phidget is a programmable input-output device used to interface Universal Serial Bus (USB) sensors and other components with a PC. The Phidget used with the OAI system is the 1203 - PhidgetTextLCD 20X2. This model features a 20 character by 2 line Liquid Crystal Display (LCD) display, 8 analog inputs, 8 digital inputs, and 8 digital outputs. It is compatible with Windows 2000, Windows XP, Windows Vista, Windows CE, Linux, and Mac OSX and can be programmed using Visual Basic 6, Visual Basic.Net, C#.Net, C++, Flash9, Java, Flex, LabVIEW, Python, and Cocoa (Phidget Inc, 2009). This model of Phidget is designed to control any device that accepts a Complementary Metal-Oxide Semiconductor (CMOS) signal or a +5V control signal. The Phidget's circuit board can also be modified to control other devices (Phidget Inc, 2009). The Phidget used for the OAI project had been modified by Rockwell Collins to be capable of controlling the six Illunis XMV 16000 imaging cameras.

The purpose of the Phidget was to trigger the first master camera which then triggered the other cameras. The camera triggering process was initiated by the airborne control computer, which also controlled the pan tilts as part of the image capture software. A signal was passed from the control computer to the Phidget via USB cable. The signal was then passed through the Phidget and triggered the master camera. The other cameras were then triggered by the master camera via a circuit housed in the same casing as the Phidget. A diagram of the process can be seen in Figure 5. Passing the signal through a master camera before triggering the others was supposed to allow changes in exposure time made to the master camera to be passed along to the other cameras. However, it was found that the exposure changes made to the master camera do not transfer to the other

cameras in a satisfactory manner. Simple triggering of cameras using the supplementary circuit board was found to be satisfactory. The supplementary circuit board remained as an artifact of the first version of the OAI system even though passing the signal through the master camera first does not add any benefits to the current version of the system.

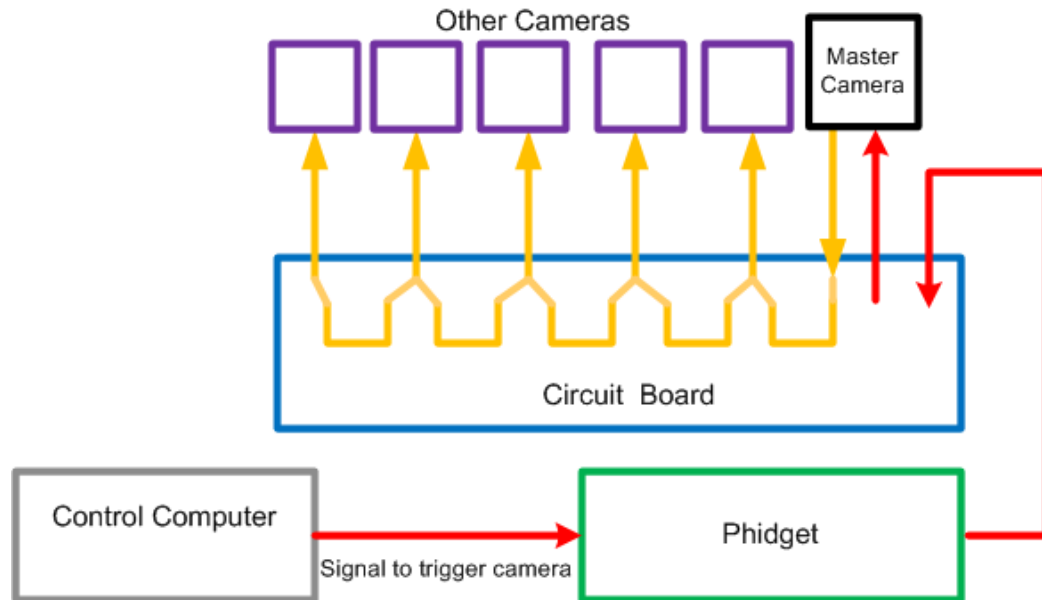


Figure 5: Camera Triggering Process

3.1.3 Elma Rack Computer

The computers used for airborne image processing were housed in an ELMA Type 11C, 4U 19" Rack Mount chassis containing six hard disk drives. The ELMA computers had several functions in the OAI system. Each of the six hard drives operated independently and processed the images of a single camera. Six CAT 6 Ethernet cables connected the imaging cameras to the front of the ELMA rack. Each Ethernet cable corresponded to a separate hard drive. When images were taken, six instances of the same processes occurred in parallel. Each hard drive saved backup copies of the images from a single camera and compressed images from that camera for transmission to the

ground station. The ELMA computers were networked with the control computer, allowing it to be controlled remotely by either the airborne control computer or through the ground station via data link. The ELMA computer was also networked with the data link antenna allowing processed images to be transferred to the ground station.

3.1.5 Pan Tilt Units

Two Directed Perception PTU-300 model Pan Tilt Units (PTUs) were used as part of the airborne equipment. One was used to point the camera head, while the other was used to point the airborne data link antenna. The maximum speed of the PTU is between 50° per second and 100° per second depending on loading. Its position resolution is 0.006°, allowing for highly accurate positioning. Software provided with the PTUs allows for manual and program-based control of the system. There are two factors limiting the PTU's motion. Software stops prevent the PTU from rotating any further than allowed by the control software, and hard stops form a physical barrier to motion. It is possible to not have hard stops and allow the pan tilt to be freely rotating (Directed Perception). Both PTUs in the airborne setup used both hard stops and software stops.

The purpose of the camera head PTU, labeled Pan Tilt 1 in Figure 2, was to ensure that the cameras were pointing to the ground location specified by the control software. Slight variations in the position of the aircraft were remedied by PTU adjustments. Likewise, the data link antenna's PTU tracked the position of the ground station antenna in order to maintain the data link. Control of the PTUs during flight testing was performed by Rockwell Collins control and tracking software located on the control computer, which was controlled from the aircraft or by the ground control station.

3.1.6 Airborne Control Computer

The computer used to control the airborne components was a Panasonic CF-30 Toughbook. The Toughbook is designed to survive shock, vibration, moisture, and dust. The model used for the OAI system uses 2 gigabytes of ram and has a 160 gigabyte hard drive. The operating system is windows XP (Panasonic). The control computer is connected to both PTUs, both GPS units and the ELMA rack. It is also indirectly connected to the cameras and data link antenna via the ELMA rack.

The control software for the PTUs and camera triggering was created by Rockwell Collins. A ground target for imaging was selected using the control software. The antenna PTU was programmed to point at the ground station based on data from the canopy antennas. Manual control of both PTUs is possible at any time by clicking on directional arrows. Imaging was controlled from the same software by pressing a “take pictures” button.

The control computer was operated either in the air or by the ground station. Airborne operation of the control computer was accomplished either by directly interacting with the control computer or by using a remote desktop program to connect a small tablet computer to the control computer. Due to space constraints in the aircraft, a small tablet PC was found to be easier for the aircrew to operate than the larger laptop. Also, the screen of the tablet computer was less prone to glare. During flight tests, the control computer was also operated by the ground station via the data link antenna. The control computer remained connected to the tablet PC via remote desktop in case connection with the ground station was lost.

3.1.4 GPS and IMU Systems

Two Novatel DL-4 Plus GPS units were used as part of the OAI system. One unit provided GPS data for the camera PTU control software and the camera head IMU. The other unit provided GPS data for the antenna PTU and antenna head IMU. A third GPS unit integrated into the aircraft was used by the pilot for flight tasks. All three GPS units used a single GPS antenna mounted to the top aft portion of the aircraft. The GPS signal was split and routed to the three GPS systems as seen in Figure 2.

The DL-4 Plus GPS unit has a number of features that made it an ideal GPS unit for this project. The DL-4 Plus is advertised as being shock, moisture, and dust resistant. It is capable of tracking up to 12 satellites simultaneously. Data logging can be performed via computer or by means of an integrated compact flash card. The DL-4 Plus can be computer controlled or operated as a stand-alone device (Novatel, 2005). Accuracy of the device depends on the type of signal being used. For the L1 signal in combination with the Wide Area Augmentation System (WAAS), position accuracy of the DL-4 Plus is 1.2 meters and its measurement accuracy is 0.75mm. The data rate for both position and measurement is 20 hertz. Time accuracy is 20ns and velocity accuracy is 0.03 meters per second (Novatel, 2006).

A Novatel IMU-G2 inertial measurement unit (IMU) with synchronized position attitude navigation (SPAN) technology was mounted to each of the two airborne PTUs. This type of IMU contains a ring laser gyro with an input range of ± 1000 degrees per second. The SPAN system samples position, attitude and velocity data at a rate of 100 hertz from the gyro and translates that information into position information (Novatel).

An overview of the dimensional accuracy of the combined IMU-SPAN system can be seen below in Table 1.

Table 1: Accuracies of IMU/SPAN system (Novatel)

Attribute	Root Mean Square Accuracy
Pitch	0.015°
Roll	0.015°
Azimuth	0.05°
Velocity	0.02m/s
Acceleration	0.03 m/s ²

Each IMU was rigidly attached to the moving portion of its corresponding PTU. In the case of the camera head PTU, it was attached behind the imaging head and served as a counter weight. For the antenna PTU, it was attached to the mounting bracket for the data link antenna on the opposite side of the PTU. Due to its high level of accuracy, even slight motions such as unevenness in the flight path were detected by the IMU. When combined with the GPS units, it was possible to determine the position of the camera head at a given time with a high degree of accuracy. When the OAI system was tracking the ground station and an imaging target, the IMUs' information was fed into the PTU control software. Based on this information, slight changes in the position of the PTUs were made to compensate for aircraft motion.

3.1.5 Antennas

Three antennas were included in the airborne portion of the OAI system. One was the aircraft's GPS antenna, another was the data link antenna, and the third was the canopy antenna.

As mentioned in the GPS equipment section, an Aero Antenna Technology model TA2775-81 GPS antenna mounted to the bottom of the aircraft was used to provide a GPS signal to both GPS units used in the OAI system and to the aircraft's GPS. The TA2775-81 model can receive both L-L1 and L2 signals. The L-L1 band operates in the 1525-1585 Mhz band and the L2 band operates in the 1217-1237 Mhz band (Aero Antenna Technology Inc, 2008).

A small Motorola 2451AP model canopy antenna was mounted to the aft equipment rack underneath the camera PTU. This antenna broadcasted aircraft location data to a corresponding ground station antenna. The location data was used to point the data link antenna towards the ground station. The canopy antenna operates in the 2.4 GHz band and has Advanced Encryption Standard (AES) encryption. Data rate is 7 Mbps aggregate. One advantage of the 2451AP model is that its synchronization and signal modulation prevent unauthorized users from accessing the system and allows more than one device to operate on the same frequency. Even if other transmitters are present the signal can retain integrity (Motorola). The canopy antenna was connected to a network switch that was connected to the GPS systems and control computer indirectly via the ELMA rack as seen in Figure 2.

Images and corresponding GPS data were transmitted through a Motorola 58600 series Point to Point (PTP) Ethernet bridge system. The PTP 58600 operates in the 5.8 GHz band with a channel size of 30 Mhz. Range of the data link connection can be up to 124 miles line of sight. The connection bandwidth can vary depending on the application, with a maximum of 300Mbps (Motorola). The data transmission requirements of the OAI system were less than half of the maximum, and no bandwidth limitations were

discovered during testing. The latency of the signal is listed as being typically less than 1 ms each direction, but once again is dependent upon the configuration of the system. The data transmitted is encoded with AES 128 bit encryption to be sure that only the corresponding receiver is able to read the transmitted data. The antenna unit used with the PTP 58600 system was a Radiowaves Xcelerator Series Flat Panel antenna. The Radiowaves antenna is capable of operating in the 5.15-5.85 GHz range with a beam with of 18 degrees (Motorola). The data link antenna was attached to the ELMA rack via a network switch as seen in Figure 2, and received its data from the ELMA rack.

3.1.7 Scene camera

During the planning phase of the OAI project, the flight path features of the OPL's Synthetic Flight Bag system (Advanced Infoneering, Inc) were going to be used to maintain the aircraft's trajectory. Preliminary flight tests revealed that the aircraft's instruments and the view from the aircraft's windows were sufficient to maintain a satisfactory trajectory. To further enhance the pilot's view, a small digital video camera was mounted to the aft rack underneath the imaging head and connected to a small LCD display visible to the pilot. The scene camera allowed the pilot to see roughly where the camera head was pointed and improved the pilot's ability to maintain a satisfactory trajectory. Note that the scene camera is not pictured in Figure 2.

3.2 Ground Station Components

This section provides an overview of the ground station components of the system along with their basic functions. A systems diagram of ground station components can be seen in Figure 6. Since many of the ground station components are identical to

airborne components, descriptions of some components will be limited to their role in the ground station as equipment specifications have been given in the previous section.

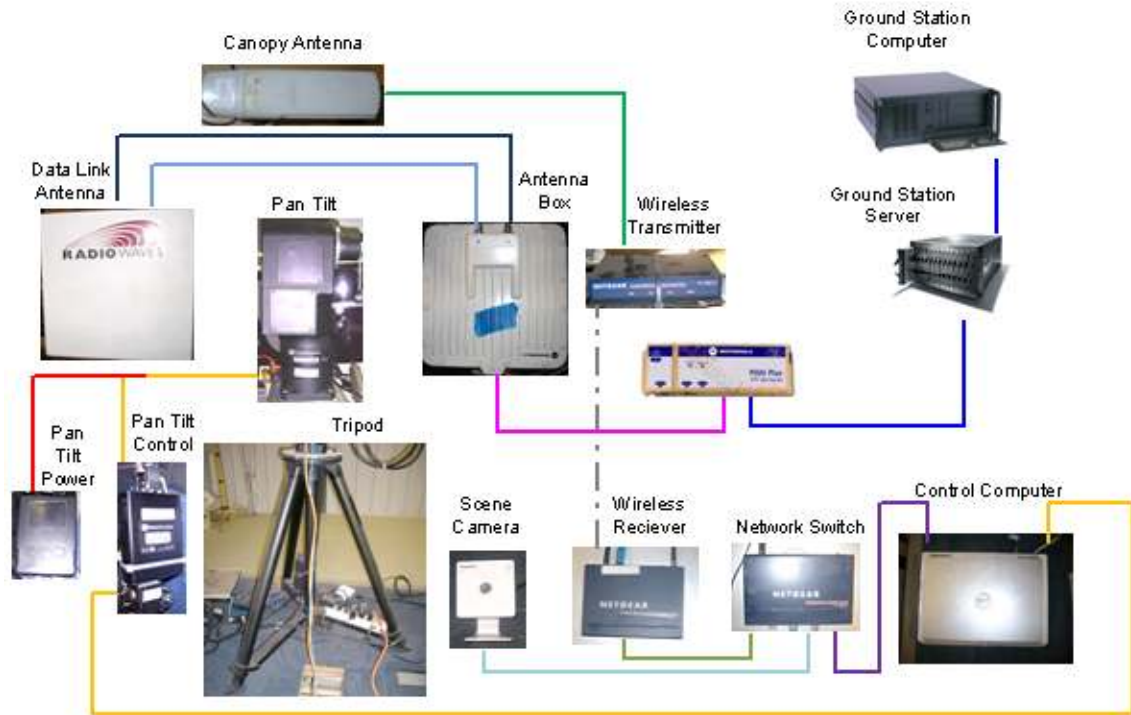


Figure 6: Ground Station Components

3.2.1 Ground Station Pan Tilt

In order to keep the antennas on the ground station aligned with those in the aircraft, the ground station antennas were mounted to a PTU-d300 PTU. The ground station PTU is the same model those used for the camera head and the airborne data link antenna. In order to accommodate circular flight paths, it was necessary for the ground station antenna to be able to rotate 360 degrees without twisting the cords attached to the antennas. To accomplish this, the ground station PTU was modified to include a slip ring connection. An Ethernet cord was connected through part of the slip ring system to allow

data link signal to pass through the rotating joint, and electrical power was attached to another portion of the slip ring. The electric connection through the slip ring was connected to a power strip mounted to the camera head that then provided electricity for all equipment mounted to the ground station PTU. The entire ground station antenna assembly was mounted on a tripod to make the ground station antenna portable, provide a stable base for the PTU, and to keep the antenna equipment off of the ground.

3.2.2 Antennas

Two ground station antennas were mounted to the ground station PTU. The first was a canopy antenna identical to the airborne canopy antenna. This antenna received the location information transmitted by the airborne canopy antenna and transmitted it to the ground station control computer via a short range wireless internet connection. A wireless connection was used for the canopy antenna because space on the slip ring system was not available for the canopy antenna's network connection. The second antenna was the data link antenna, which is also identical to its airborne counterpart. The ground station data link antenna received image and GPS data from the airborne data link antenna then passed the data through the slip ring of the PTU to the ground station server, then on to the image processing computer.

3.2.3 Control Computer

The ground station control computer runs control software identical to that of the airborne control computer on a Dell Inspiron 6000 laptop. Unlike the airborne computer, the ground based computer controls one PTU instead of two. The ground station PTU was either programmed to track the airborne data link components using the canopy antennas or was controlled manually using direction arrows.

3.2.4 Image Server

After arriving at the ground station through the data link, image and location data were passed to the ground station image server. The image server was used to store backup copies of OAI images and data. The original image sets were stored on the server, and then copies of the images were passed to the processing computer. Processed images were also saved to the server.

3.2.5 Image Processing Computer

A desktop computer was used for processing OAI images. Six images taken at the same time were copied from the image server and processed using Rockwell Collins image processing software. The software took as an input six images and corresponding GPS data from a single point in time, stitched them into a single image, rotated them in a North-up orientation, and then output a single stitched image and file with location information. Image processing was repeated for every set of images taken by the camera head while the plane was still in the air. The Rockwell Collins imaging software is discussed in more detail in Chapter 5.

The image processing computer also served as a point for remote control of airborne components and monitoring of the entire OAI system. The image processing computer was able to remotely link to the airborne control computer through the data link antenna. Six instances of a simple program monitoring the incoming data from each camera provided the ground station operator with an indication that each camera was taking pictures and the images were being transmitted successfully through the data link. A view of the output folder for the image processing software allowed the ground station operator to know that image processing was being completed successfully.

CHAPTER 4. HARDWARE DEVELOPMENT

This chapter describes the process used to design the hardware for the airborne portion of the OAI system to fit the imaging needs of the project presented in this report and to integrate the flight hardened system into the OPL's Beechcraft Bonanza. The camera head was designed and fabricated along with two equipment racks, floor raisers, and miscellaneous small components.

Several general design constraints were relevant for all components described in this chapter. Because the equipment created for this project was intended for use in an aircraft, safety and reliability were top concerns for all components. Each component had to be connected directly or indirectly to hard points in the aircraft. Heavy or bulky items such as the pan tilt devices required additional safety tethers connected directly to aircraft hard points. It was important to keep components light, yet rigid enough to not vibrate excessively.

The general design process for creating each component began with concept designs made using Pro Engineer (ProE) software. The completed digital model of the concept design can be seen in Figure 7 and a picture of the installed system can be seen in Figure 8. Rough models of each component were constructed out of inexpensive plywood to check for fit and design integrity. Many of the rough models and finished components were produced by creating two dimensional part files from the three dimensional concept designs, then using a Shopbot Computer Numerically Controlled (CNC) milling machine to cut the parts. The final components were made out of aluminum.

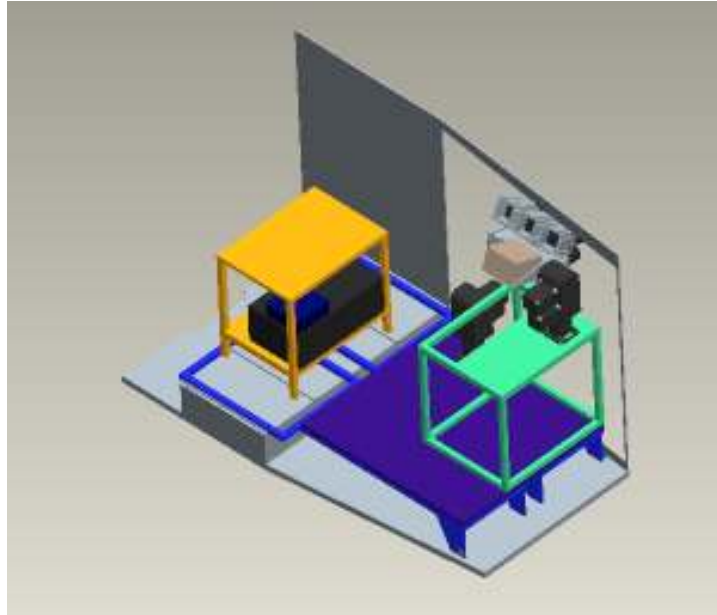


Figure 7: Complete Hardware Concept Design



Figure 8: Picture of OAI System Installed in Aircraft

4.1 Design of Camera Head

Of the components designed for this project, the camera head had the most extensive list of design constraints. Specifications of the PTU-300 PTU dictated that the weight of the entire camera head could not exceed 35 pounds, however for optimal speed and performance it was desirable to cut the weight in half, to 17.5 pounds (Directed Perception). Given that the cameras weighed 1.2 pounds each and the IMU weighed 7.5 pounds, the goal was to design a camera head using less than 2.8 pounds of aluminum. It was also extremely important that the IMU be rigidly attached to the camera reference framework in order to achieve accurate location tagging of each image. In order to avoid gaps in the stitched images, the position of the 6 cameras had to be calibrated so that there was overlap between the viewpoints. An adjustable framework was needed for the camera mounts to allow for calibration. However, the camera mounts must also be held rigidly once the proper angle is achieved. Another required feature was the ability to remove cameras from the camera mounts without affecting calibration. A final design requirement was to provide support structures to run two cables from each camera to the computer rack to prevent the PTU from becoming entangled in cables.

4.1.1 Concept Designs

The design of the camera head was an evolutionary process that yielded multiple concept designs. A summary of these concept designs can be seen in Figure 9 . The first idea explored was to use identical modules for each camera with the angles between cameras built in to each unit. The individual units could be fitted together like puzzle pieces so that additional cameras could easily be added to the system at a later date. This concept can be seen in designs A and B of Figure 9. Design A shows a basic wedge and

design B shows the same design with puzzle piece like additions so that the units interlock like blocks. The wedge unit concept was abandoned because the position of the camera mounts was not adjustable and the complex geometry of the units would be difficult to machine. Some features from the wedge design were carried on to subsequent designs, such as rectangular cavities to protect the camera and a center plate that would allow cameras to be removed from each unit without disturbing calibration settings. The next generation of concept designs can be seen in boxes C and D of Figure 9. The designs seen in boxes C and D simplified the camera mounts into boxes that could be tied together with metal sheets or wedges. While the box designs would be easier to machine than wedge design, the centrally located mounting plate meant that the camera mounts would have to be either welded together or machined out of a solid block of aluminum. To simplify machining and allow for easier adjustment of lens settings, the protective housing around the lens end of the camera was removed as seen in concept designs E and F of Figure 9. The final camera mount design was similar to concept design F. However, instead of making the camera mount out of a solid piece of aluminum, a removable aluminum plate for mounting the camera was attached to the front of a length of extruded aluminum. The front mounting plate could then be removed without disturbing the orientation of the extruded portion, allowing cameras can be removed from the camera mount without disturbing the calibration.

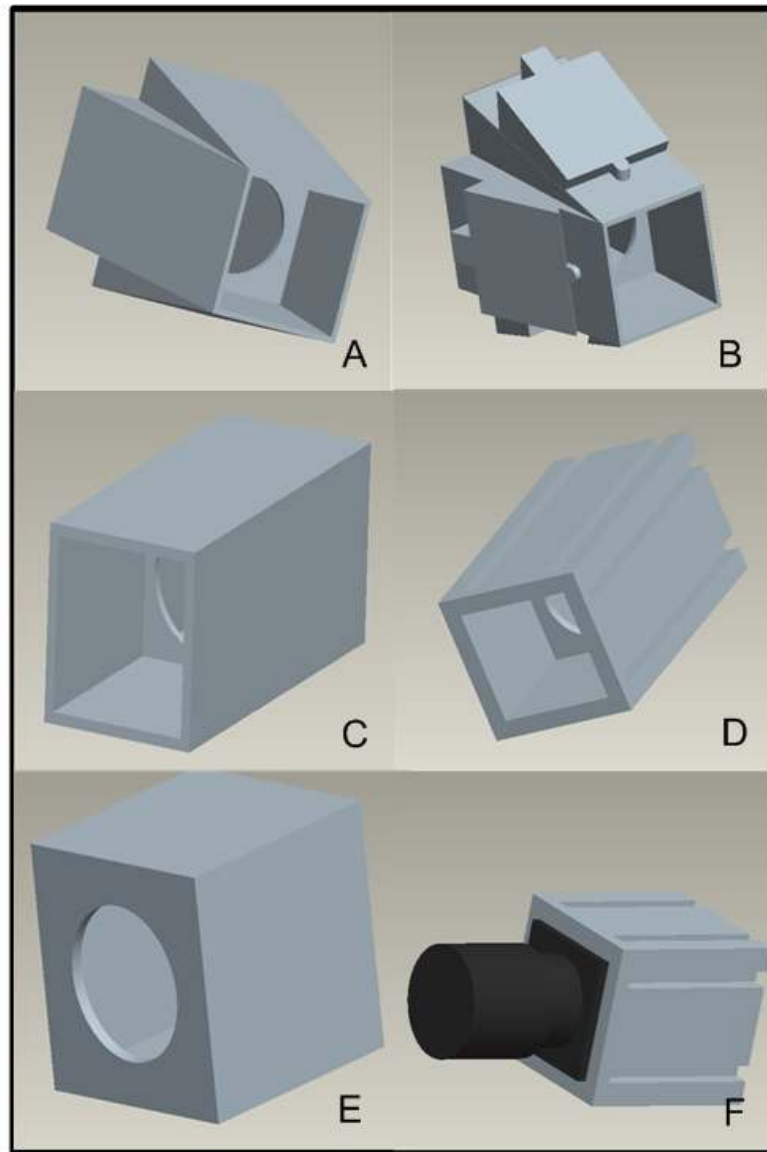


Figure 9: Evolution of Camera Enclosure Concept Design: A) Wedge B) Joinable Wedge C) Long Box D) Slotted Box E) Short Box F) Short Slotted Box (Final Design, Shown with Camera)

Since the final camera mount design did not include a way to create angles between cameras, it was necessary to design a series of plates to hold the camera mounts in the correct position. Figure 10 and Figure 11 show the two concept designs that were considered. The design shown in Figure 10 held the camera mounts together by a series

of metal plates containing the proper angles. The design shown in Figure 11 uses adjustable wedges hold the camera mounts together. The wedge design was thought to be more versatile than the plate design since a large number of wedges could be rapidly machined to achieve different overlap angles between cameras. However, a plate design was chosen for the camera head because it allowed the positions of the camera mounts to be adjustable and the overall structure was found to be more rigid during the prototype phase.

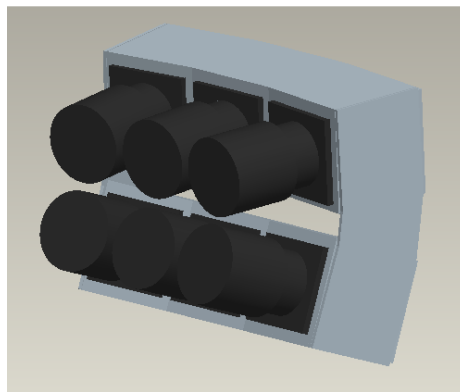


Figure 10: Camera Head with Plate Design

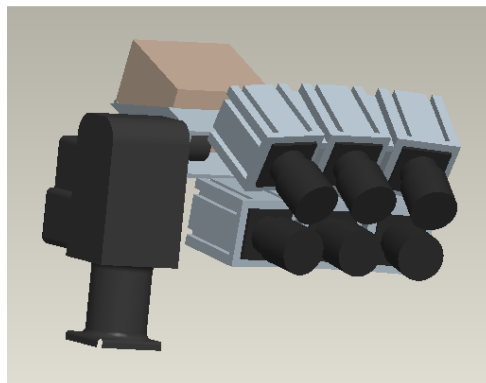


Figure 11: Camera head with Wedge Design

4.1.2 Prototype Phase

The second phase of the camera head design was to create several plywood models of the camera head. This was done to verify fit of components, and to make any final changes to the design. Photographs of two plywood prototypes can be seen in Figure 12. Several changes were made to the camera head design during the prototype phase. The design with support structures located on the outside edge of the camera head seen on the left side of Figure 12 was found to bend easily. To solve the problem, the primary support structure was relocated to the center of the camera head. The updated design can be seen in the right side of Figure 12. The change in the support structure configuration also resulted in a slight change to the IMU mounting plate as seen in Figure 13. The left side of Figure 13 shows the first concept design and the right side shows the modified design. Additional support structures were added to the rear of the camera head during the prototype phase. The support structures provided the camera head with additional stability and a place to tie down the power and network cables from each camera.



Figure 12: Camera Head Prototypes



Figure 13: IMU Mounting Plate Prototypes

4.1.3 Final Design

The finished camera head can be seen in Figure 14, Figure 15, and Figure 16. The slots on the back end of each camera used for adjusting camera mount position during calibration and the rear braces added to the design in the prototyping phase can be seen in Figure 14. Figure 15 illustrates how the weight of the cameras and the weight of the IMU balance over the arm of the PTU. The weight balance allows the PTU arm to rotate more easily. Figure 16 shows how each camera is attached to its camera mount by means of an aluminum plate. The finished camera head had a total weight of approximately 19 pounds, which allowed the PTU to function well even though it was slightly higher than the goal weight of 17.5 lbs.

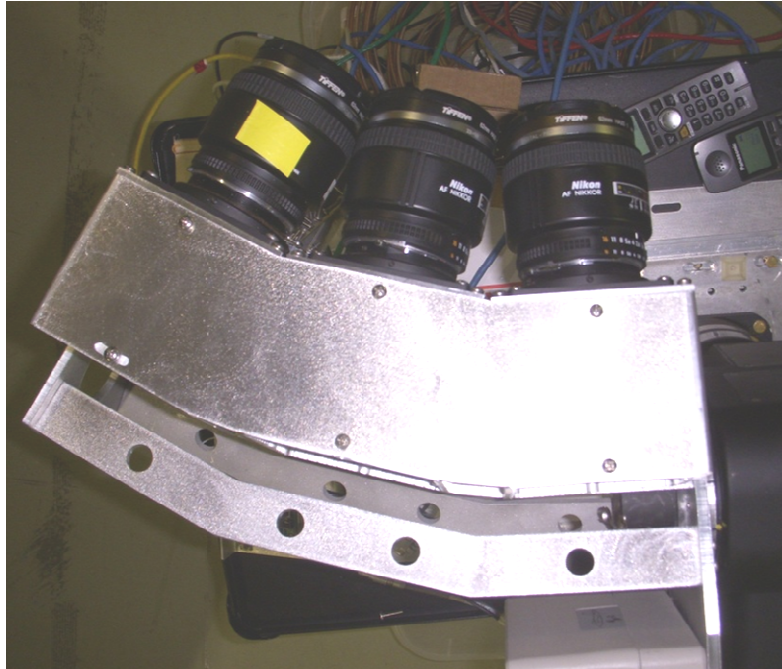


Figure 14: Finished Camera Head Top View

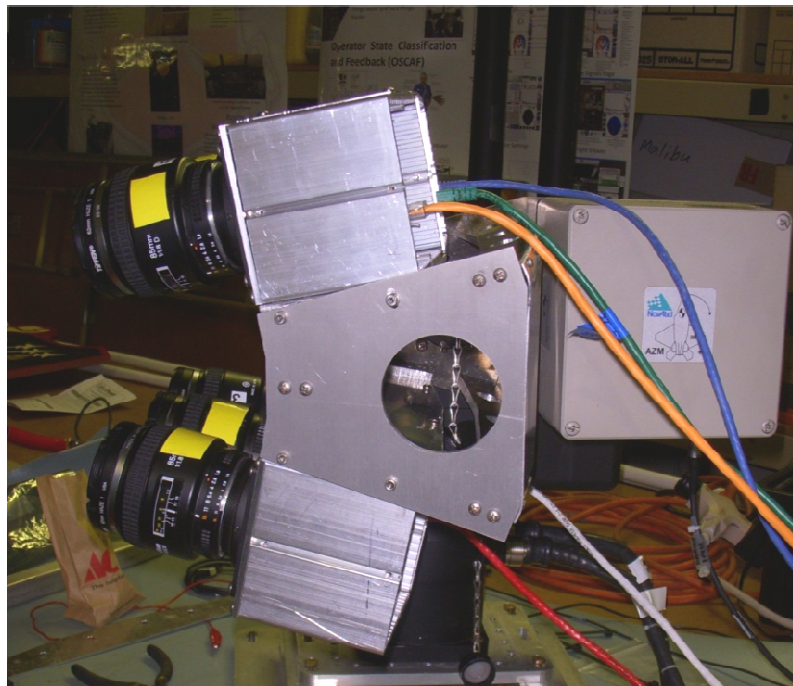


Figure 15: Finished Camera Head Side View



Figure 16: Finished Camera Head Front View

4.2 Design of Other Components

Design of other components for this project followed the same procedure as the design of the camera head. Because the other components were less complex than the camera head, many were completed after only a single concept design and prototype. In some cases only the outer dimensions of a component were reflected in the prototype to ensure proper fit.

4.2.1 Floor Raiser Design

Floor raisers were added to the OPL's Beechcraft Bonanza to create a flat surface tied into the hard points of the aircraft on which to mount the airborne portion of the OAI

system. The addition of floor raisers simplified the design of other components because they could be mounted anywhere on the floor raiser instead of being mounted directly to hard points. The floor raiser also made installation of equipment easier because the raised floor is level with the aircraft doorway and the middle seat rails. The floor raisers made it possible to slide equipment racks into position instead of lifting them up and over the door frame, then lifting them again on to the middle seat rails or hard points. The space between the raiser and the aircraft's floor also made it easier to insert tools for tightening bolts.

Concept Designs for the floor raisers can be seen in Figure 17. The top rectangular portion of the floor raisers connect to the middle seat rails and the tabs connect to the hard points used for the rear seats. The difference between the two designs shown in Figure 17 is the angles of the rails coming from the relative to the rectangular portion that sits on the middle seat rails. The design on the right side of Figure 17 was predicted to be easier to weld, while the design on the left better filled the space within the aircraft. Experimenting with scrap pieces of aluminum tubing revealed that the angle in the design on the left would not be any more difficult to weld than the ninety degree angle seen on the right. Therefore, the design on the left was chosen as the final design for the floor raisers.

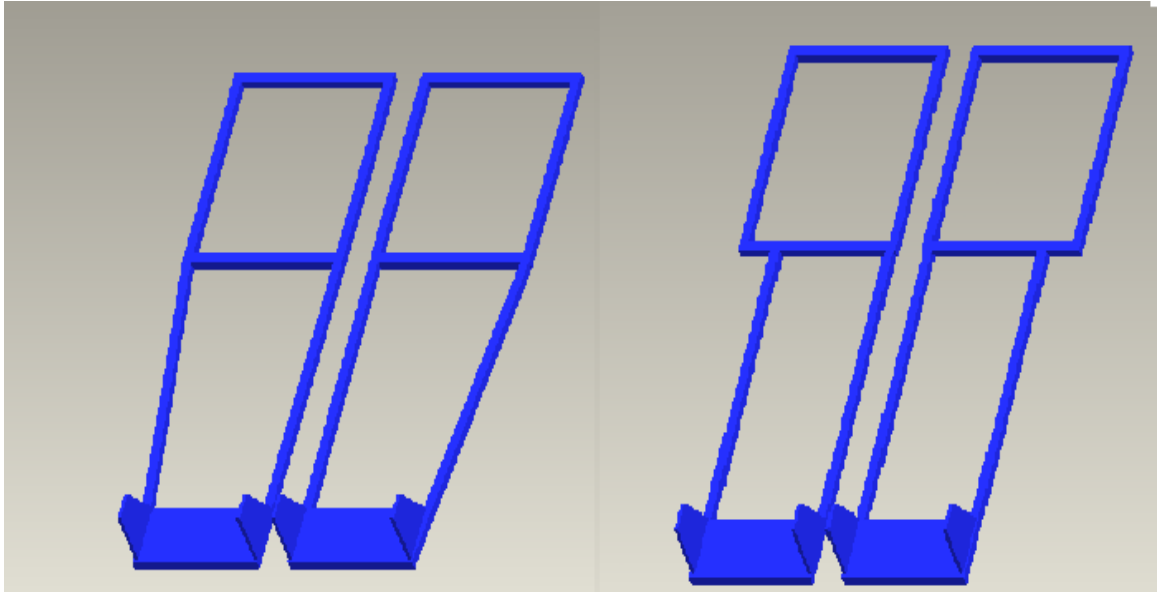


Figure 17: Floor Raiser Concept Designs

During the prototyping phase, additional crossbeams were added to the concept design. The crossbeams allowed the floor raisers to be mounted to all six back seat hard points and provided more area for equipment to be attached. One of the finished floor raisers can be seen in Figure 18.



Figure 18: Finished Floor Raiser

4.2.2 Front Equipment Rack Design

When designing the front equipment rack, it was important to provide a solid structure for attaching the Elma computer, the control computer, two GPS units, power supplies and miscellaneous cables. The dimensions of the base of the front equipment rack had to be the correct size to tie into the middle seat rails. During the design phase, a complete list of equipment that had to be attached to the front equipment rack was not available. Therefore, the concept design was created with excess space and a variety of surfaces for mounting equipment. The concept design seen in Figure 19 conforms to the 19 inch standard for rack mount computers. The 19 inch standard allowed for the installation of the ELMA computer and aligned the legs of the rack with the middle seat rails. The front equipment rack was designed to be taller than was needed for the ELMA computer so that additional equipment could be accommodated at a later date. The solid top plate provided additional stability and served as a second equipment shelf.

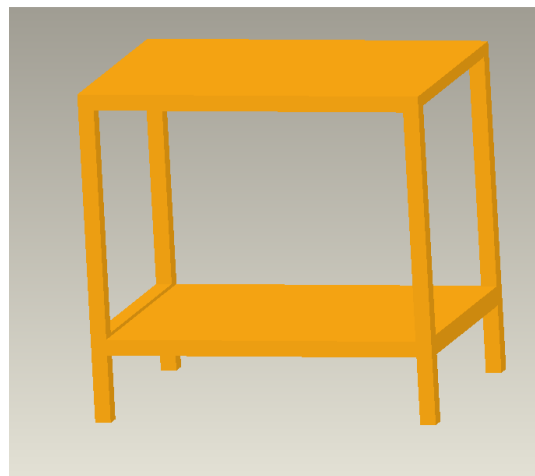


Figure 19: Concept Design for Front Equipment Rack

The finished front equipment rack can be seen in Figure 20. The dimensions of the finished front equipment rack are the same as the dimensions of the concept design model. However, side plates, a center shelf and three Air Transport Radio (ATR) style computer mounts were added to accommodate the OPL's SFB system. Not visible in Figure 20 is the power supply equipment mounted underneath of the rack. The additional equipment and support structures made the 103lb front equipment rack by far the heaviest assembly of the airborne portion of the OAI system.

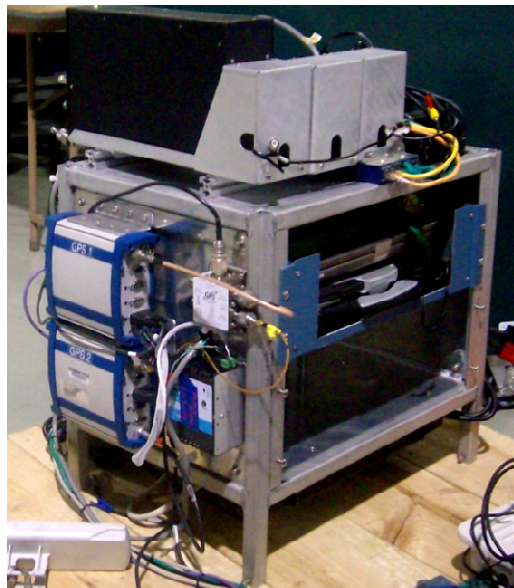


Figure 20: Front Equipment Rack

4.2.3 Aft Equipment Rack Design

The final component designed for integrating the system into the aircraft was the aft equipment rack. The aft rack provided mounting locations for the camera PTU, the Phidget, the canopy antenna and the scene camera. The most important design consideration for the aft rack was that its dimensions determined the position of the

camera PTU and therefore the position camera head. The camera head had to be close to the top aft corner of the door as possible so that the camera head could be pointed almost straight down without the view being obscured by the floor or wings of the aircraft. At the same time the aft rack had to be designed so that the camera head could not make contact with the aircraft walls, ceiling, or the outside airstream at the full limits of the PTU's rotation. The design for the aft rack also had to be rigid to prevent vibrations from the aircraft or the motion of the pan tilt from affecting the images. Additionally, the aft rack design had to allow enough space in the aircraft doorway for the data link antenna to not be obscured by the wing.

The concept design for the aft equipment rack can be seen in Figure 21. The dimensions of the aft rack were designed to allow the camera head to sit as far up, aft, and close to the outside airstream as possible while staying at least two inches away from the walls, ceiling, and airstream at maximum rotation of the PTU. An aluminum plate covering half of the top of the aft rack and an additional cross beam were added to provide additional support and stability for the camera PTU. During the prototyping and construction phase, it was found that the single plate was not sufficient to prevent vibration. Additional aluminum plates were added to the sides and back of the aft rack. These plates prevented vibration, made it easier to mount the Phidget device, allowed the PTU power supplies to be attached to the aft rack, and provided a place to attach cables. The final aft rack with camera pan tilt installed can be seen in Figure 22.

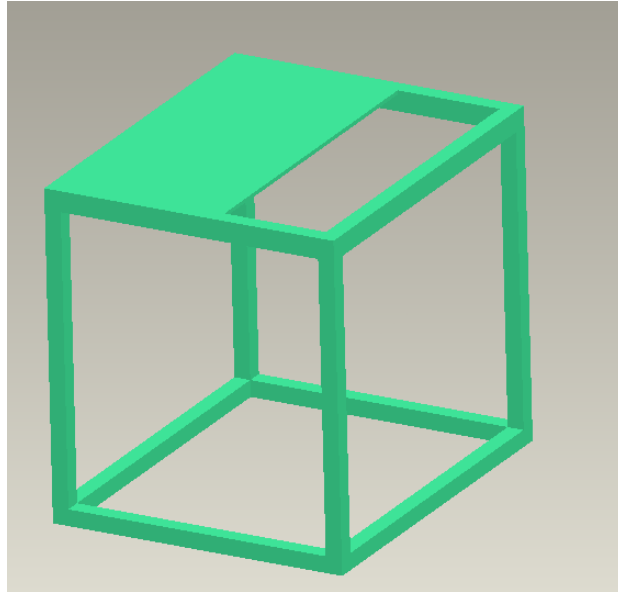


Figure 21: Concept Design of Aft Equipment Rack



Figure 22: Completed Aft Rack with Camera Pan Tilt Attached

CHAPTER 5. SOFTWARE

This chapter is intended to provide the reader with an overview of software for the OAI system. The image processing software is covered briefly with emphasis on describing its function. The KML super overlay software and user interface are described in detail.

5.1 Image Processing Software

The Rockwell Collins image processing software used to stitch and rotate OAI images is proprietary to Rockwell Collins and NASA JPL. Therefore, only the general function of the software will be discussed here.

The image processing software takes six images from a single point in time and creates a single stitched image rotated so that north is pointed upwards. Multiple factors contribute to successfully stitching and geolocating the final image. Data from the airborne GPS system is needed to determine the location of the aircraft, and IMU data gives the position and rotation of the camera head. Camera calibration data is used to quantify the overlap between images. The boundary latitudes and longitudes for each stitched image can then be calculated mathematically. Several types of image transformation are used to create the final stitched image. Because the camera head is pointed out the aircraft's door at an angle rather than being mounted directly below the plane, the amount of ground covered by each individual image in a set varies as seen in Figure 23. The imaging software contains algorithms to account for the distortion caused by the camera head position angle and transforms the images to fit a view point directly above the scene. Camera calibration data is used to determine the seam locations between images and the images are cropped accordingly. Finally, the cropped and

transformed images are attached at the seam lines and the stitched image is rotated so North is facing up. After the transformations and rotations, the image is no longer rectangular. To create a rectangular image, white space is added around the ground image as seen in Figure 23. The output of the image processing software is the rectangular image and a file in the Geographic (GEO) format that contains the latitude and longitude coordinates of the boundaries of final rectangular image.



Figure 23: Example of a finished OAI image

5.2 KML Super Overlay

The size of the final OAI images is 9,677 pixels by 8,658 pixels. This size is too large to load in Google Earth as single image. In order to view the image in Google Earth without reducing the resolution of the image, it is necessary to create a super overlay. A super overlay is a KML programming method that allows Google Earth to load smaller sections of the image at progressively higher detail levels as the user zooms in. To accomplish this, the large image is broken up into an image pyramid then loaded into Google Earth using a series of interlinked KML files.

5.2.1 Image Pyramid

The first step in creating a super overlay in Google Earth is to create an image pyramid. An image pyramid takes the large image and breaks it apart into layers of smaller images. The first layer of images contains a single image which is the original image with an added black border compressed to a small size, in this case a 256 pixel x 256 pixel square. The second level contains four images, each representing a quarter of the initial image. These images are also 256 pixel squares. Each image in level 2 seeds four images to create 16 images in level 3. An overview of the tiling process can be seen in Figure 24 below. The tiling process continues until the picture no longer needs to be compressed to make the 256 pixel square tiles. For the OAI images, six levels of image tiling are needed before the 256 pixel tiles contain uncompressed sections of the original image. The number of images in a level can be calculated according to Equation 2. Using this equation, it can be shown that the sixth level of the image pyramid contains 1,024 tiles and the image pyramid contains a total of 1,365 tiles.

$$T_N = 2^{2(l-1)}$$

Equation 2: Number of tiles (T_N) in a Given Level (l) of an Image Pyramid

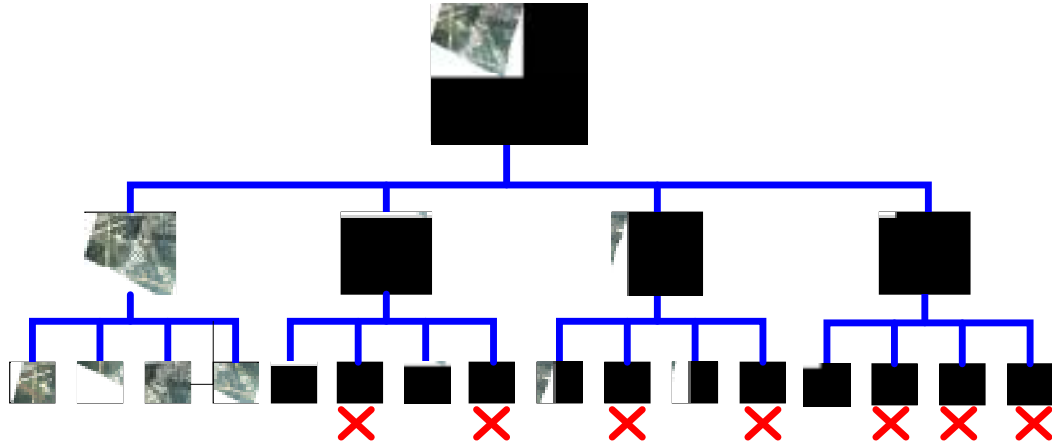


Figure 24: The First Three levels of an Image Pyramid. Squares containing no part of the original image (marked by a red X) are not created by the image pyramid software

It can be seen in Figure 24 that there is a black border added around the original image. The black border is added by the image pyramid creation software so the image can be broken up evenly into 256 pixel by 256 pixel squares for each level image pyramid. Squares in the image pyramid containing no part of the original image are not saved in the output folder. The tiles marked with a red X in Figure 24 therefore are not actually created. This saves space in the final output folder and processing time because blank tiles that have not been created do not spawn further blank tiles or take up space.

Several image pyramid creation programs are available for free online. The one used for the OAI project is called Photo Overlay Creator and was produced by the Centre

for Advanced Spatial Analysis at University College London (UCL). A download of the program is available for free on the Center for Advanced Spatial Analysis website (University College London, 2006). To use the Photo Overlay Creator, an image file is loaded into the program and a preview is displayed showing the image and the number of tiles to be created. A start button is then pressed to create a folder with the name “(name of original image)-tiles” in which all levels of the image pyramid are saved. Nomenclature for each individual tile follows the pattern: (name of original image)_(level)_(row)_(column). For example “OAIImage6_2_2_1” would be the name of the tile in the image pyramid for OAIImage6 at level 2 of the image pyramid, row 2, column 1. The nomenclature of the images is important for the super overlay creation procedure described in the next section.

5.2.2 Procedure for Creating a KML Super Overlay

In order to correctly place image tiles, it is necessary to calculate the latitude and longitude of the boundaries of each tile. While the latitude and longitude of the original image is known, black space is added around the image by the Photo Overlay Creator program to make the image pyramid divide evenly as seen in Figure 24. The first step in determining the boundaries of individual tiles is to calculate the degrees latitude per pixel and degrees longitude per pixel as shown in Equation 2 and Equation 3.

$$Lat_p = \frac{N_o - S_o}{h_o}$$

Equation 3: Degrees Latitude Per Pixel (Lat_p) in Terms of the Northern Boundary of the Original Image (N_o), Southern boundary of the Original Image (S_o), and Height in Pixels of the Original Image (h_o)

$$Lon_p = \frac{E_o - W_o}{w_o}$$

Equation 4: Degrees Longitude Per Pixel (Lon_p) in Terms of the Eastern Boundary of the Original Image (E_o), Western Boundary of the Original Image (W_o), and Width in Pixels of the Original Image (w_o)

Because the final layer of the image pyramid has the same resolution as the original image, the latitude and longitude per pixel in the altered image are the same as that the original. Given a tile size of 256 pixels x 256 pixels, dimensions of the altered image at full resolution (the number of pixels if all the full resolution tiles were combined into one image) can be calculated according to Equation 5.

$$h_n = w_n = 256 * 2^{L_{max}}$$

Equation 5: Calculation of Height (h_n) and Width (w_n) in Pixels of the Image Composed of the Highest Resolution Image Pyramid Tiles Based on the Number of Levels in the Image Pyramid (L_{max})

Coordinates for the Northern and Western boundaries of the new image are the same as those of the original image ($N_n = N_o$ and $W_n = W_o$) because the original image is located in the top left corner of the image pyramid layout. Given Equation 3, Equation 4, and Equation 5, the Southern and Eastern boundary coordinates for the new image can be calculated according to Equation 6 and Equation 7.

$$S_n = N_n - (Lat_p * h_n)$$

Equation 6: Southern Coordinate of the New Image (S_n) in Terms of the Northern Coordinate of the New Image (N_n), Latitude per Pixel (Lat_p), and Height of the New Image in Pixels (h_n)

$$E_n = W_n - (Lon_p * w_n)$$

Equation 7: Eastern Coordinate of the New Image (E_n) in Terms of the Western Coordinate of the New Image (W_n), Longitude Per Pixel (Lon_p), and Width of the New Image (w_n)

Knowing the outer coordinates of the image allows the coordinates of any tile in the image pyramid to be calculated. The calculation is based on the catalogue number of the image tile. Each tile is labeled “imagename_l_m_n” where imagename is the name of the original image, l is the level the tile is in the image pyramid, m is the position of the tile in the North-South axis, and n is the position of the tile in the East-West axis. The numbering of positions starts with (m,n)=(0,0) in the Northwest corner of the image. Calculation of the four boundary coordinates of image tile “imagename_l_m_n” can be calculated according to Equation 8, Equation 9, Equation 10, and Equation 11.

$$N_{lmn} = N_n - \left[\left(m * \frac{1}{2^l} \right) * (N_n - S_n) \right]$$

Equation 8: Calculation of the Northern Coordinate of Tile “imagenam_e_l_m_n” (N_{lmn}) Based on the Northern Coordinate of the Total Image (N_n), the Southern Coordinate of the Total Image (S_n), Level in the Image Pyramid (l), and the Tile's North-South Position (m)

$$S_{lmn} = N_n - \left[\left((m+1) * \frac{1}{2^l} \right) * (N_n - S_n) \right]$$

Equation 9: Calculation of the Southern Coordinate of Tile “imagenam_e_l_m_n” (S_{lmn}) Based on the Northern Coordinate of the Total Image (N_n), the Southern Coordinate of the Total Image (S_n), Level in the Image Pyramid (l), and the Tile's North-South Position (m)

$$E_{lmn} = W_n - \left[\left((n+1) * \frac{1}{2^l} \right) * (W_n - E_n) \right]$$

Equation 10: Calculation of the Eastern Coordinate of Tile “imagenam_e_l_m_n” (E_{lmn}) Based on the Eastern Coordinate of the Total Image (E_n), the Western Coordinate of the Total Image (W_n), Level in the Image Pyramid (l), and the Tile's East-West Position (n)

$$W_{lmn} = W_n - \left[\left(n * \frac{1}{2^l} \right) * (W_n - E_n) \right]$$

Equation 11: Calculation of the Western Coordinate of Tile “imagenam_e_l_m_n” (W_{lmn}) Based on the Eastern Coordinate of the Total Image (E_n), the Western Coordinate of the Total Image (W_n), Level in the Image Pyramid (l), and the Tile's East-West Position (n)

A KML file is created for each tile in the image pyramid. An initial KML file is created as a starting point of the super overlay. An example of the initial file can be seen in the first section of Appendix A. The user selects this initial file to open the super

overlay in Google Earth. The file consists of some start up code, followed by a command called network link which has sub commands of name, region, and link. The network link command is used to load the file specified by the link sub command, in this case the image tile that makes up the first level of the image pyramid. The region sub command provides the boundaries of the region where the file is to be loaded, and uses the Level of Detail (LOD) command to specify the conditions under which the specified region becomes active. In other words, if the user is looking at the specified area within a certain zoom range, the file containing the first level tile of the image pyramid becomes active and that tile is displayed.

All KML files associated with image pyramid tiles between level 0 and level (l-1) have the same format. An example of an intermediate-level file can be seen in the second section of Appendix A. The first command of each of these files specifies the active region of that file. Each of the files then includes four network links to its four child files. The network links specify the region, name, level of detail, and link are for each child file. Equation 12 can be used to determine the four child files for parent file (l,m,n). As a reminder, l refers to level in the image pyramid, m refers to the North-South coordinate of the corresponding image tile, and n refers to the East-West coordinate of the corresponding image tile. The intermediate level KML files conclude with a ground overlay command which displays the current image tile as a ground overlay.

$$C_1=(l+1,2m,2n)$$

$$C_2=(l+1,2m,2n+1)$$

$$C_3=(l+1,2m+1,2n)$$

$$C_4=(l+1,2m+1,2n+1)$$

Equation 12: The Four Child Files (C_1, C_2, C_3, C_4) for the Parent File Corresponding to “ $im名称_l_m_n$ ”

Because the files for tiles of the final level in the image pyramid have no child files, the KML code for the final layer of tiles does not contain any network links. The KML files associated with the final level of the image pyramid contain a region command to specify the active region and a ground overlay to display the current tile. An example of a final level KML file can be seen in the third section of Appendix A.

5.3 KML Super Overlay Generator Program and User Interface

Individually typing all of the KML files for an image pyramid would be a tedious, slow, and repetitive task for the user. In order to automate the process a KML super overlay generator program with a graphical user interface (GUI) was created. The GUI includes all steps necessary to put an airborne image into Google Earth, including image pyramid selection, image placement, and KML file generation. The GUI was designed to be used by both the author and by future users of the OIA system. Future users might not be familiar with the role of the PNG file format, transparencies, latitude and longitude

coordinates, or the GEO file format. In order to explain these features and help the user make informed decisions about KML super overlay options, help links are provided for these features. The GUI can be seen in Figure 25. Features of the GUI are documented in this section, and the C# code for the program is included in Appendix B.

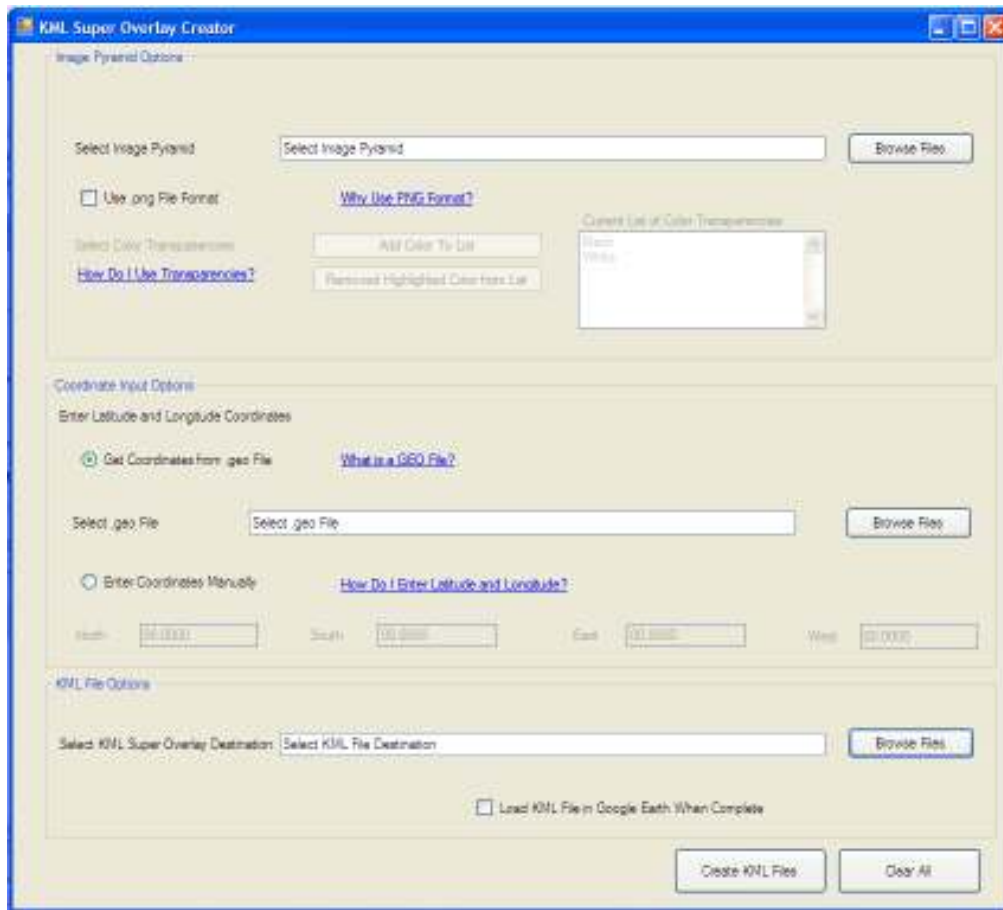


Figure 25: KML Super Overlay Generator GUI

The first section of the KML generator can be seen in Figure 26. This section of the GUI provides options for selecting an image pyramid and for choosing which if any colors will appear as transparent in the Super Overlay. The first control in the image pyramid options section of the GUI is “Select Image Pyramid”. The “Select Image” control allows the user to select the image pyramid that is to be placed in Google Earth

by entering a file path or selecting a folder using the “Browse Files” button. The location of the file is then displayed in the “Select Image Pyramid” text box. If a file path typed into the control does not represent a valid folder, an error message indicating an invalid file path appears.

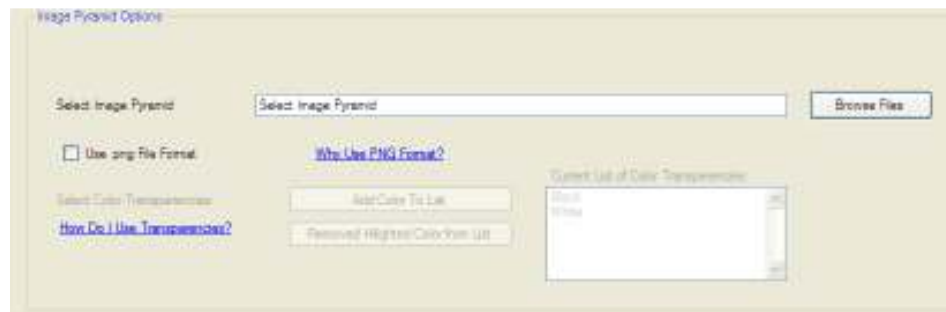


Figure 26: Image Pyramid Options Section of KML Super Overlay Generator GUI

The next control of the “Image Pyramid Options” section is a checkbox titled “Use .png File Format”. The Portable Network Graphic (PNG) file format is used to create images with a bitmap of indexed colors under lossless compression. The PNG format contains three color channels and an alpha channel that specifies the opacity of pixels (Gao, 2009). Using the PNG file format is desirable in situations where colored borders need to be removed around the image because it is possible to specify pixels to appear as transparent in the Google Earth photo overlay. The Photo Overlay Creator used to create the image pyramid can only output images in the Joint Photographic Experts Group (JPEG) file format. The transparency feature cannot be used with a JPEG file format because the JPEG format has no alpha channel. In order to convert the image pyramid to PNG format it is necessary to change the file type of every tile in the image

pyramid. Changing the file types is not necessary in cases where no transparencies are to be used, and the process of converting files to PNG and applying transparencies takes several minutes. Therefore, the conversion of the image pyramid to PNG format is an optional feature of the KML generator program. Since future users of the airborne imaging system may not be familiar with the purpose of the conversion to PNG file format, a help link titled “Why use .png Format?” is placed next to the “Use .png Format” check box.

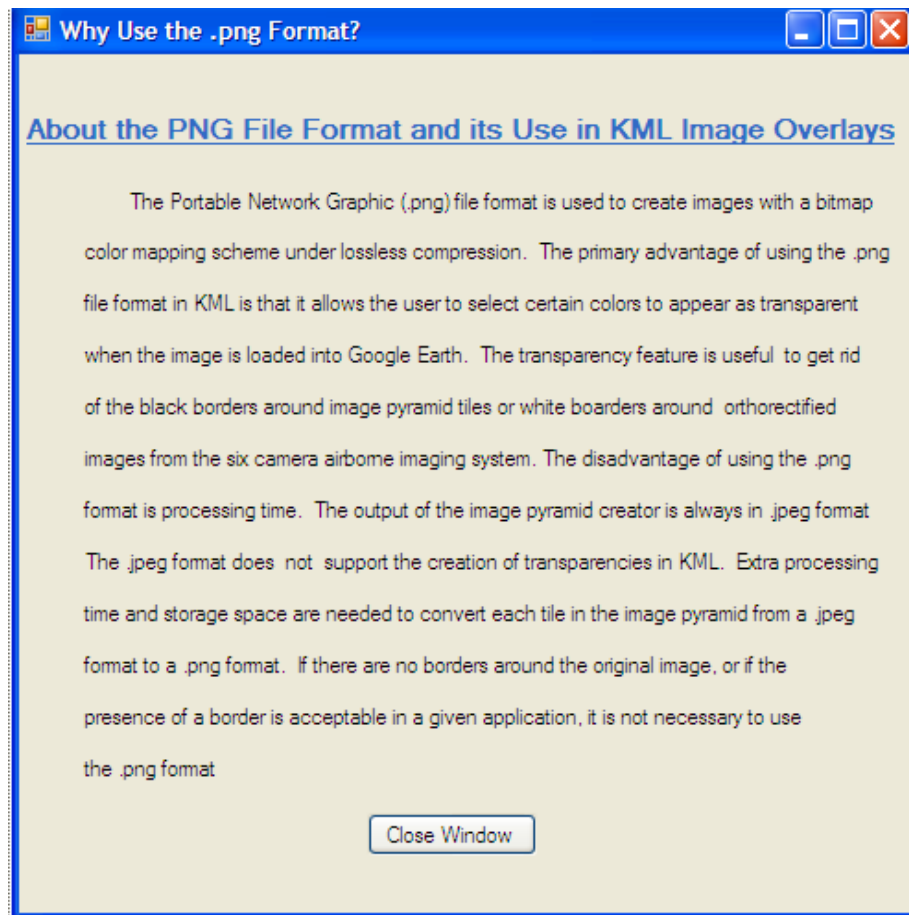


Figure 27: Help Window for the "Why Use .png Format?" Link

If the “Use .png File Format” checkbox is selected, additional features of the Image Pyramid Options box become active as seen in Figure 28. These features allow the user to add and remove colors from the list of colors to be made transparent in the Google Earth super overlay. The default colors listed are black and white. Black is included because it is the color that the Photo Overlay Creator adds around the image to make the image pyramid divide evenly. White is included because it is the color added to the border of the stitched images to make them rectangular. The stitched image is not visibly affected by the use of transparencies because very few pixels in the image are pure white or pure black. The “Add Color to List” button is used to add colors to the list of transparency colors. The “Add Color to List” button opens a Windows color selection dialog from which the user can select a predefined or custom color. The result of the user color selection is then displayed in the “Current List of Color Transparencies” list box. To remove a color from the list, the user highlights the color to be removed and clicks the “Remove Highlighted Color from List” button. Both user-added colors and default colors may be removed using the “Remove Highlighted Color from List” button. The “How Do I Use Transparencies?” link opens a window that explains the use of transparencies. The “How Do I Use Transparencies?” help screen can be seen in Figure 29.



Figure 28: Image Pyramid Options Section with “Use .png File Format” Option Active

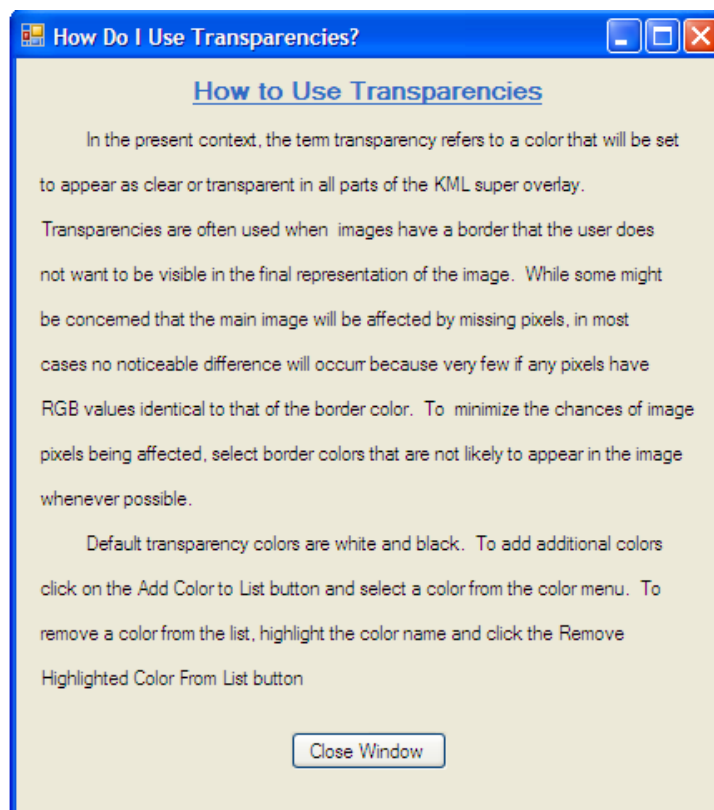


Figure 29: Help Window for the "How Do I Use Transparencies?" Link

The second section of the KML super overlay creator interface seen in Figure 30 provides the user with options for entering the latitude and longitude coordinates of the edges of the image. The default coordinate entry method is to read the latitude and longitude information from a GEO type file. The GEO file type is used to store information about an image and allows the latitude and longitude of each pixel to be determined. Files in the GEO format consist of two sections. The first section contains information the address of the image file as a Uniform Resource Locator (URL). The second section of GEO file contains a list of tie points. Each entry of the tie point list provides a name or number for the tie point, the X and Y coordinates of the tie point within the image, and the latitude and longitude of the tie point. The GEO files produced by the airborne imaging system use two tie points: one corresponding to the upper left corner of the image, and the other corresponding to the lower right corner. The latitude and longitude of the two corner points define the boundaries of the image. When the “Get Coordinates from .geo File” radio button is selected, the user is able to enter the location of a GEO file in the “Select .geo File” text box or select a GEO file using the “Browse Files” button. The “What is a .geo File?” link provides a description of the GEO file type as seen in Figure 31.

Figure 30: Coordinate Input Options Section

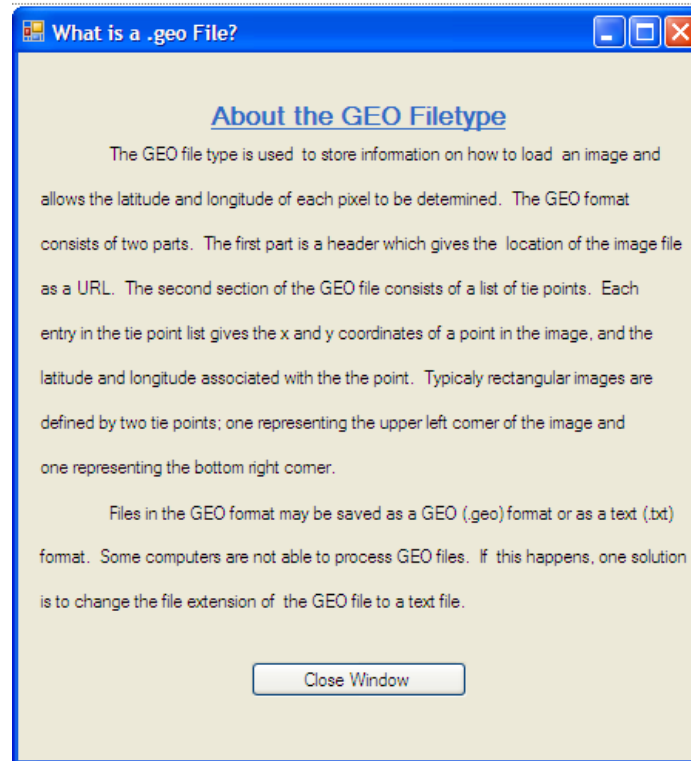


Figure 31: Help Window for the "What is a .geo File? Link

The second option for entering latitude and longitude coordinates is to enter them manually rather than loading them from a file. To enter coordinates manually, the user first selects the “Enter Coordinates Manually” radio button. Selecting the “Enter Coordinates Manually” option deactivates the “Browse Files” button and the “Select .geo File” text box. At the same time, four text boxes corresponding to the four cardinal directions become active. The user can then type the coordinate for each direction. There are two formats for entering latitude and longitude: the degrees-minutes-seconds format and the decimal format. In KML, decimal coordinates are used. Because there may be some confusion on how to enter latitude and longitude, the “How Do I Enter Latitude and Longitude” help link is provided. The “How Do I Enter Latitude and

Longitude” help window seen in Figure 32 specifies the format for entering latitude and longitude, and provides the user with a link to a website that has a converter for changing degrees-minutes-seconds format into decimal format.

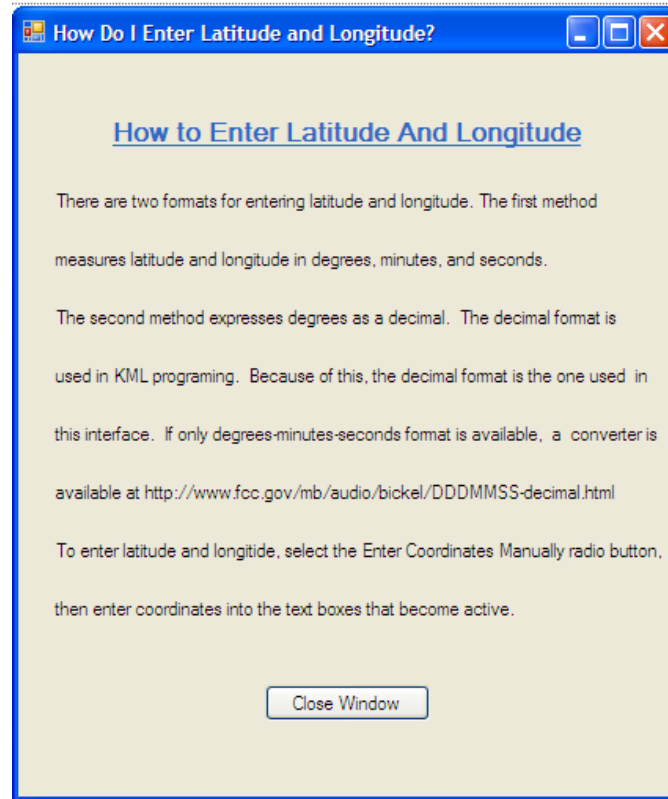


Figure 32: Help Window for the "How Do I Enter Latitude and Longitude?" Link

The third section of the KML super overlay generator titled “KML File Options” can be seen in Figure 33. The “KML File Options” section of the interface provides the user with options for saving and displaying the KML super overlay. The first control is the “Select KML File Destination” text box. The user can type in a location to save the files for the KML super overlay or select a destination using the “Browse Files” button.

A checkbox titled “Load KML File in Google Earth When Complete” gives the user the option of automatically opening the first file of the super overlay after all of the super overlay files have been created.

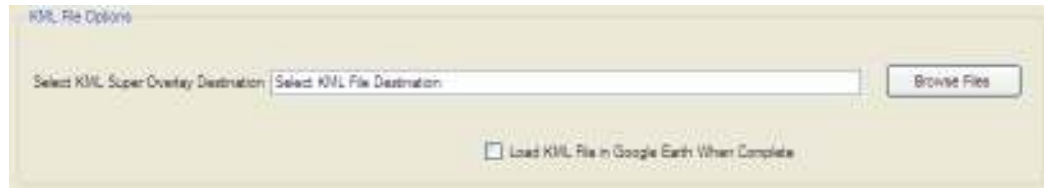


Figure 33: KML File Options Section of the KML Super Overlay Generator Interface

The final two buttons of the KML super overlay creator interface are the “Create KML Files” button and the “Clear All” button. The “Create KML Files” button generates the image pyramid and the KML super overlay according to the specifications provided by the user in the “Image Pyramid Options”, “Coordinate Input Options”, and “KML File Options” portions of the interface. The “Clear All” button clears all user input and returns all portions of the user interface to default values.

CHAPTER 6. CALIBRATION AND DATA COLLECTION

This chapter provides an overview of the calibration and data collection procedures used to collect airborne images using the OAI system. This chapter includes flight testing procedures and an overview of data flow. Image processing, importation into Google Earth, and imaging results are briefly mentioned, but are not discussed in detail.

6.1 Camera Calibration

Before flight testing of the OAI system could take place, the placement of the camera mounts was adjusted and the amount of overlap between images was measured in a calibration process. Camera calibration was performed on the ground with the camera PTU installed on a tabletop rather than in the aircraft. A preliminary adjustment of camera angles was used to achieve an overlap of approximately 100-200 pixels in both the horizontal and vertical directions. A calibration process was then used to precisely measure image overlap. Information obtained during the calibration process was used by the image processing software as a guideline for how to stitch the images.

The initial setup for camera calibration can be seen in Figure 34 below. An additional Illunis XMV 4021 camera was attached to the camera head along the horizontal axis at a distance of 27.5 inches from the center of the camera head. The seventh camera provided a view of the entire scene to which individual camera views were compared. The camera head with seventh camera attached can be seen in Figure 34. A 44 inch white square board with a 10 by 10 grid of 2 inch diameter dots was placed approximately five meters in front of the camera head. The calibration board can be seen

in Figure 35. An example of an image set produced by the seven cameras can be seen in Figure 36.

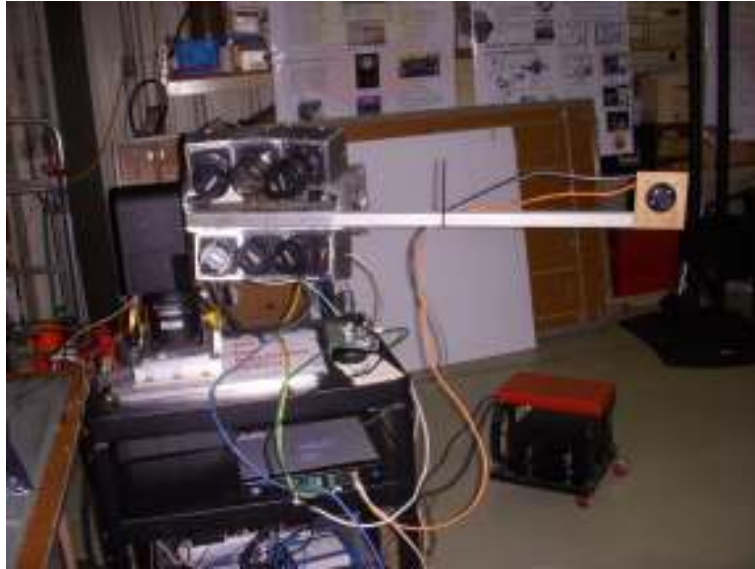


Figure 34: Camera Calibration Setup

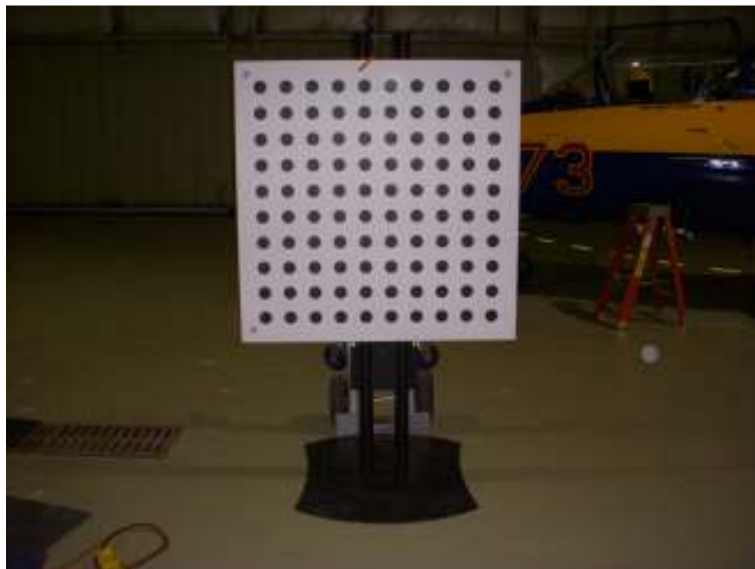


Figure 35: Camera Calibration Board

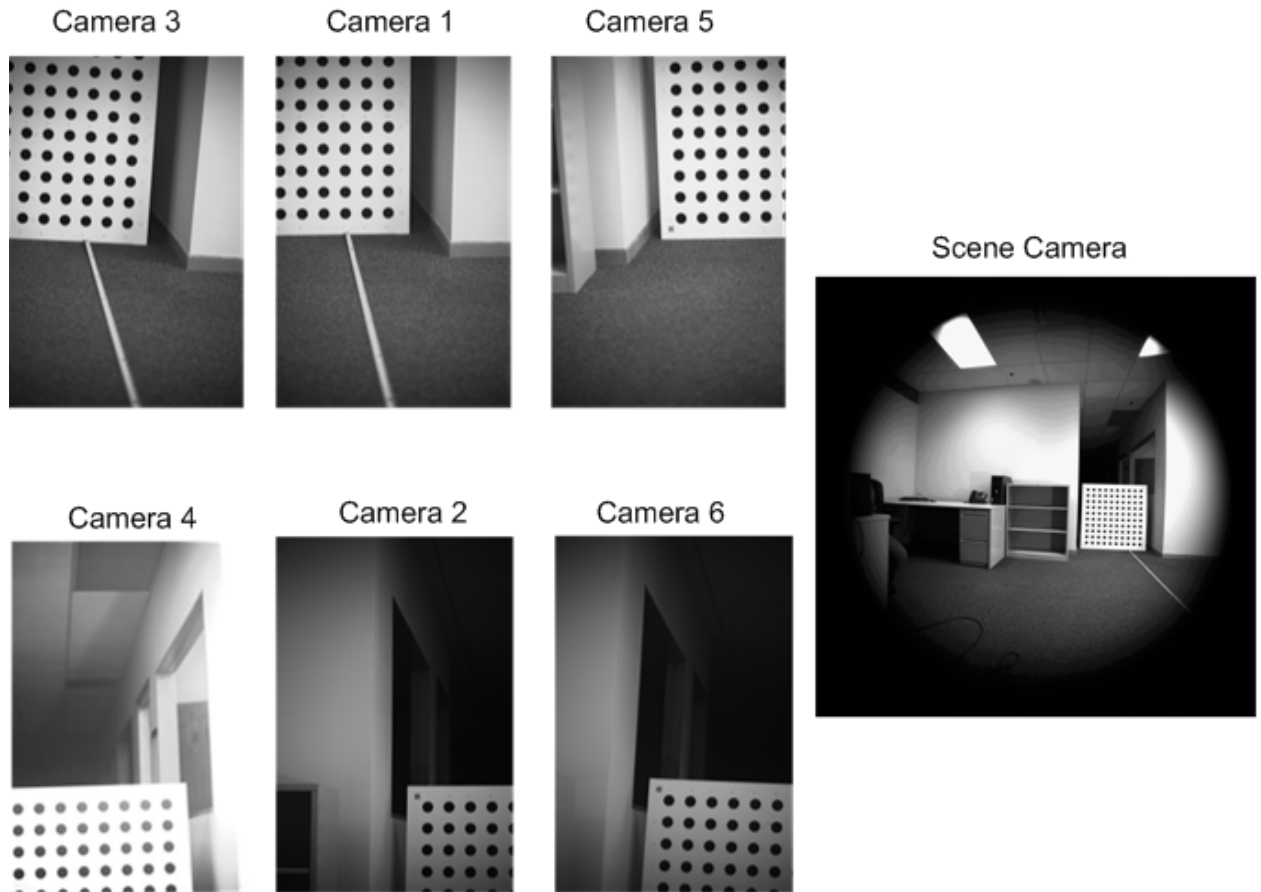


Figure 36: Example of Calibration Image Set with Six Imaging Cameras and a 7th Calibration Camera

Once the setup for camera calibration was complete, the camera mounts were adjusted within the camera head frame so that the images had approximately 100-200 pixels of overlap in both the horizontal and vertical directions. To do this, the imaging cameras were turned on so that a continuous view could be seen from each of the six cameras. Using the calibration board as a guide, the position of each camera was adjusted by hand until an acceptable overlap was achieved. The center top camera was used as a stationary reference for the other five imaging cameras. After all the imaging

cameras were in an acceptable orientation, the bolts connecting the camera mounts to the camera head were tightened, preventing any loss of calibration.

To begin the next phase of camera calibration, a GPS antenna was attached to the camera system and the PTU control software was started. The lens focus of each camera was changed to 5 meters to match the distance between the camera head and calibration board. A preliminary set of pictures was taken to check that the cameras were in focus using visual inspection of the preliminary images. Once acceptable focus was achieved, the preliminary pictures were cleared from the hard disks. Step size on the PTU control software was set to 5 degrees, meaning that one click of the manual control arrows moved the PTU 5 degrees in the selected direction. The camera head position was moved to three steps (15 degrees) up and three steps (15 degrees) to the left. The operator then issued the “start taking pictures” command to begin collecting images. After a few seconds, the pan tilt was shifted to its next position at three steps (15 degrees) up and two steps (10 degrees) to the left. One step shifts in camera head position are then issued every few seconds until the entire area had been covered. A grid showing the stopping points of the calibration pattern can be seen in Figure 37. The board was then moved to 7 meters and the process was repeated starting with refocusing the cameras. The process was then repeated twice more with the calibration board located 10 meters and 15 meters from the camera head.

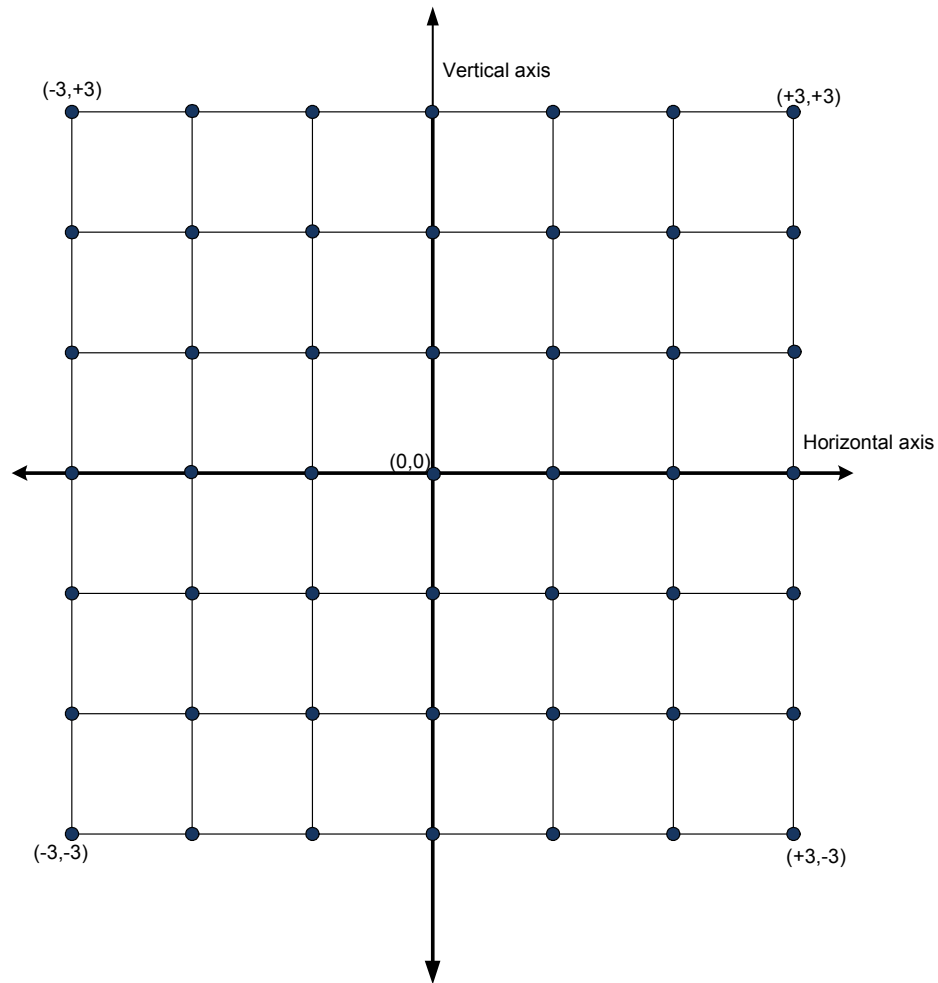


Figure 37: Calibration Scan Grid: Each Dot Represents a Point in the Scan Grid. Units are in Steps with One Step Equal to 5 Degrees of PTU Rotation

During the camera calibration process, the view of the calibration board from each of the six imaging cameras was compared to that of the seventh scene camera using the circles on the calibration board. Calibration software calculated the geometric relationships between the imaging cameras and created a set of calibration data that was used by the image stitching software. For some positions in the scan pattern, the calibration board was not visible to all imaging cameras. It was acceptable for the calibration board to not be visible by some of the imaging cameras. However, the

seventh scene camera must have a view of the entire calibration board at all times. The scan pattern at each distance can be performed in any order, but each scan pattern must be completed continuously with no movement occurring in the background scene. Any people or objects were prevented from entering the viewing range of the cameras while performing calibration.

6.2 Data Collection

Flight testing of the OAI system took place from March 2009 until August 2009. Preliminary flight testing was performed with the intent of testing equipment. The first flight test was performed without the data link or control computer installed. Images from a stationary camera head were recorded on the ELMA computer to prove that the cameras were working and that the placement of the camera head relative to the door of the aircraft was satisfactory. During subsequent flight tests, additional equipment and layers of complexity in the data collection process were added piece by piece and tested until the entire system was installed and running smoothly. Several final flight tests to collect the data used in this project were performed in late July and early August of 2009.

6.2.1 Airborne Components Procedures

The first and foremost concern of any flight testing procedures is safety. The nature of the OAI equipment brings up several safety concerns not encountered in normal aircraft operation. In order to obtain images of the ground, the OPL's A36 Beechcraft Bonanza was flown with the aft doors removed. This is an acceptable configuration for this model of aircraft (in fact the A36 is sometimes used for skydiving), but requires special safety precautions. Care was taken so that no object within the aircraft was able to fall out of the door during flight. All equipment was secured to the aircraft structure

either directly or indirectly, including small objects such as cables and sheets of paper. Preflight checks of the system included a thorough check for any loose paper, extra hardware, misplaced tools, and any other debris. Another concern when flying with the aft doors removed is temperature. Air temperature decreases with altitude, so it was important for flight crew to be dressed for the temperatures at altitude rather than ground level temperatures. Since most flying was done during the spring and summer months, a flight suit over pants was enough to keep the flight crew warm. For the first flight tests occurring in March and April, flight jackets and gloves were also worn. Interaction between OAI equipment and the outside air stream was a concern during early flight tests since the location and strength of the airstream were unknown. Preliminary flight tests indicated that in the full range of PTU operation neither the camera head nor the antenna head entered the outside air stream. In addition to these special concerns, all standard aircraft safety measures were adhered to before, during, and after flight testing. As in any aircraft operations, it was important for all persons involved in flight testing to be vigilant for any unusual sights, sounds, or smells that could indicate a problem. Thanks to the preparation, cooperation, and vigilance of all persons involved, flight testing took place without any flight safety problems.

Equipment power up and equipment checks took place on the ground before flight testing. Ground startup was done instead of in air startup because the equipment was easier to inspect visually on the ground, and any noises indicating equipment problems could be heard with the aircraft's engine off. Starting equipment on the ground also reduced the electrical demands on the aircraft because the power-intensive start up sequence could be performed with the equipment plugged into a ground power unit rather

than relying on the aircraft's electrical system. A picture of the startup configuration with ground power cart visible can be seen in Figure 38. The startup sequence began with powering up both the camera PTU and the antenna PTU. Each PTU was observed as it went through its startup sequence of moving to the software induced limits of its motion range in each direction. Researchers looked and listened for any irregularities in PTU operation during the startup sequence. Next the ELMA computer and control computer were started. Remote login to the control computer from the tablet computer was used to load the PTU control software and to confirm that the software had located both PTUs. The backs of the imaging cameras were examined to see that two lights indicating power and data connection were lit.

After all equipment was powered up and checked, the pilot and one person aircrew boarded the aircraft and went through the preflight checklist. An additional task added to the preflight list for this project was to establish radio connection with the OAI ground station personnel. Prior to each flight test, a primary and secondary radio frequency were chosen so if the primary frequency was to fail unexpectedly or conflict with another user a backup frequency could be used. After starting the aircraft's engine, ground power was disconnected by ground crew. From that point onward, the OAI equipment was supplied power by the aircraft's alternator through a bulkhead power connection. After final preflight checks by the pilot, the OAI system was ready to take off.



Figure 38: Pre-Flight Setup with Ground Power Cart Visible

After taking off and establishing a flight path, data link contact was established between the aircraft and the ground station. While it is possible to establish a data link with the aircraft on the ground, ground level obstructions such as buildings and turns on taxiways that pointed the airborne antenna away from the ground station made it impossible to maintain a data link during takeoff. Therefore, data link was established after the aircraft had begun its flight path. Once the data link was established, imaging was started by pressing the “Take Pictures” button on the airborne control computer’s control software. The “Take Pictures” button could be pressed either by the aircrew via the tablet computer or by ground control personnel via the data link. On each flight test, thousands of pictures were collected. The length of a flight test depended on how well data collection was working. The earlier flight tests were longer because both air and ground crew were troubleshooting errors in the system. Early flight tests often lasted over an hour. Later flight tests were approximately a half hour in length. Flight paths

and altitudes during data collection varied depending on cloud cover. The imaging target was usually programmed to be the location of the ground station, but occasionally other areas such as the Iowa River near the Iowa City Airport and the Iowa City Airport's runways were used. At the end of data collection, OAI equipment was shut down while in air, and the plane landed.

There were three different roles for aircraft crew, which were performed by three to five people. The roles were pilot, aircrew, and ground crew. The role of pilot was in all cases performed by Professor Thomas Schnell. The aircrew role was performed by the author or by other OPL employees. Ground crew required two or three people per flight test. Ground crew tasks were often completed by the same researchers who performed ground station tasks, but occasionally other OPL employees filled the ground crew role as needed.

Tasks for the pilot included air safety and directing the aircraft. Before flight tests, the pilot was responsible for all aircraft-related pre flight tasks. The pilot was responsible for checking weather conditions, performing aircraft safety checks, and completing the aircraft's preflight checklist. During flight the pilot was responsible for safe aircraft operation, radio contact with OPL ground personnel as well as other aircraft, and maintaining a flight path that kept the airborne data link antenna pointed towards the ground station data link antenna. The pilot was not responsible for any in-air troubleshooting or operation of the OAI equipment.

Aircrew tasks during preflight included assisting with OAI equipment startup. During preflight, the person in the aircrew role was responsible for checking that the tablet computer was connected to the control computer, and that the control computer

software was operational. During the flight test, aircrew sat in the co-pilot seat of the aircraft as the entire aft passenger compartment was full of OAI equipment. The person in the aircrew role was responsible for maintaining radio contact with ground station personnel throughout the flight test and monitoring OAI equipment for malfunction. Any airborne troubleshooting, including software input adjustments and restarting portions of the OAI system during flight, was performed by the aircrew. Restarts were facilitated by locating power switches of the pan tilts and the ELMA rack within arm's reach of the co-pilot seat. If the airborne control computer was being used to control the OAI system, it was the aircrew's job to press the "Take Pictures" button on the control software. In order to maintain safe flight, the aircrew was responsible for following pilot instructions and reporting immediately to the pilot anything that might indicate a flight hazard.

The tasks for the ground crew occurred during preflight and post-flight operations. During preflight, ground crew was responsible for rolling the aircraft out of the hangar, fueling the aircraft, and performing start up procedures for the OAI equipment. Ground crew personnel were also responsible for unplugging ground power after engine start up. Unplugging ground power was a particular safety concern because the engine of the aircraft was running during the process. A specific set of hand signals between the pilot and person unplugging the ground power are used for all of OPL's flight tests to ensure that the pilot knows when aircrew is approaching the aircraft and the aircrew knows the pilot is aware of their location at all times. After flight tests, the ground crew was responsible for putting the plane back in the hangar. During early test flights before the data link antenna was installed, ground crew personnel were also responsible for copying

OAI data from the ELMA rack hard drives to a server then clearing the ELMA rack drives.

6.2.2 Ground Station Procedures

Ground station flight test procedures began with equipment setup. The ground station data link and computer cart were stored in the OPL hangar and were rolled outside before each flight. The location of the ground station for flight testing was on a paved area at the Southeastern corner of the hangar area at the Iowa City Airport. This location was chosen because it allowed the antenna to be in an area with no aircraft traffic and no buildings to block the data link connection. At the chosen location, it was also possible to house the ground station computers inside one of the hangars, preventing screen glare and allowing the operator to stay cool during summer months. A portable radio station was also set up to communicate with the aircraft.

Prior to takeoff, the ground station equipment was powered up. Like the airborne PTUs, the ground station PTU was observed as it moved through its range of motion during the startup sequence to check for any problems. It was not possible to check the quality of the data link signal during preflight, but the connections between computers and the antennas could be checked by pinging each device. The control computer and control software were started and radio contact was established with the aircrew.

During the flight, data link connection was established between the ground based data link antenna and the airborne data link antenna. Imaging was monitored through six instances of a monitoring application that listed image files as they entered the ground station server. If all of the data channels were transmitting erratically, it was a sign of a data backup on the transmitting end of the data link and data collection was stopped

for a few seconds to allow the backed up images to pass through the data link to the ground. If a single data stream was not showing received files, a bad connection between the camera and the ELMA rack or a camera malfunction was the problem. Stitched and rotated images were produced as the images came in from the airborne equipment. After the flight test was completed, the ground station equipment was shut down and wheeled back into the OPL's hangar.

There were three roles for ground station personnel which were filled by two or three researchers. The roles were computer operator, radio operator, and data link observer. The role of computer operator was performed by a Rockwell Collins employee who had detailed knowledge of the imaging software. The radio operator task was performed either by the author or another OPL employee. The data link observer task could be performed by the radio operator, or could be assigned to another researcher. The data link observation task was performed by the author or by another OPL employee.

The ground station computer operator acted as the directing entity of the OAI system operation. The ground station computer operator controlled the airborne OAI equipment and monitored image results as they come in. During flight tests, the ground station computer operator was in charge of monitoring the six streams of incoming data and the output of the image stitching software. If any part of the data stream was malfunctioning, the ground station computer operator notified the rest of the team and advised the aircrew and ground crew on possible ways to correct problems in the system.

The radio operator's role was to facilitate communication between ground station personnel and airborne personnel. Since the ground computer operator was located in a building that was unable to receive radio signal and was too busy to take on additional

communications tasks, a radio operator was used to communicate between ground crew and air crew. The radio operator also served as a filter for information, relaying only relevant information between aircrew and ground crew in a clear, concise manner.

Because a handheld radio was used, the radio operator was able to move between the ground station PTU and the hangar where the ground station computers were located.

This allowed the radio operator to provide the aircrew with information about the orientation of the ground station antenna if data link was lost.

The final ground crew role was data link observer. The data link observer was in charge of monitoring the ground station data link for any unusual movements or noises, and to visually confirm that the ground station PTU was tracking the aircraft.

6.3 Overview of Data Flow

An overview of the OAI image collection process can be seen in Figure 39. The process begins with images of the ground being collected by the imaging head. The images pass to the ELMA rack computer, which saves a backup copy of the images, compresses the images and GPS data, then sends the compressed data to the airborne data link antenna. Next the image and GPS data are transmitted to the ground station via data link. Upon reaching the ground station, the images are decompressed and saved. Sets of six images taken at a single point in time are stitched and rotated using Rockwell Collins image processing software to achieve a large orthorectified stitched image. Finally, the large image is made into an image pyramid, and a KML super overlay is created to load the image pyramid into Google Earth for viewing.

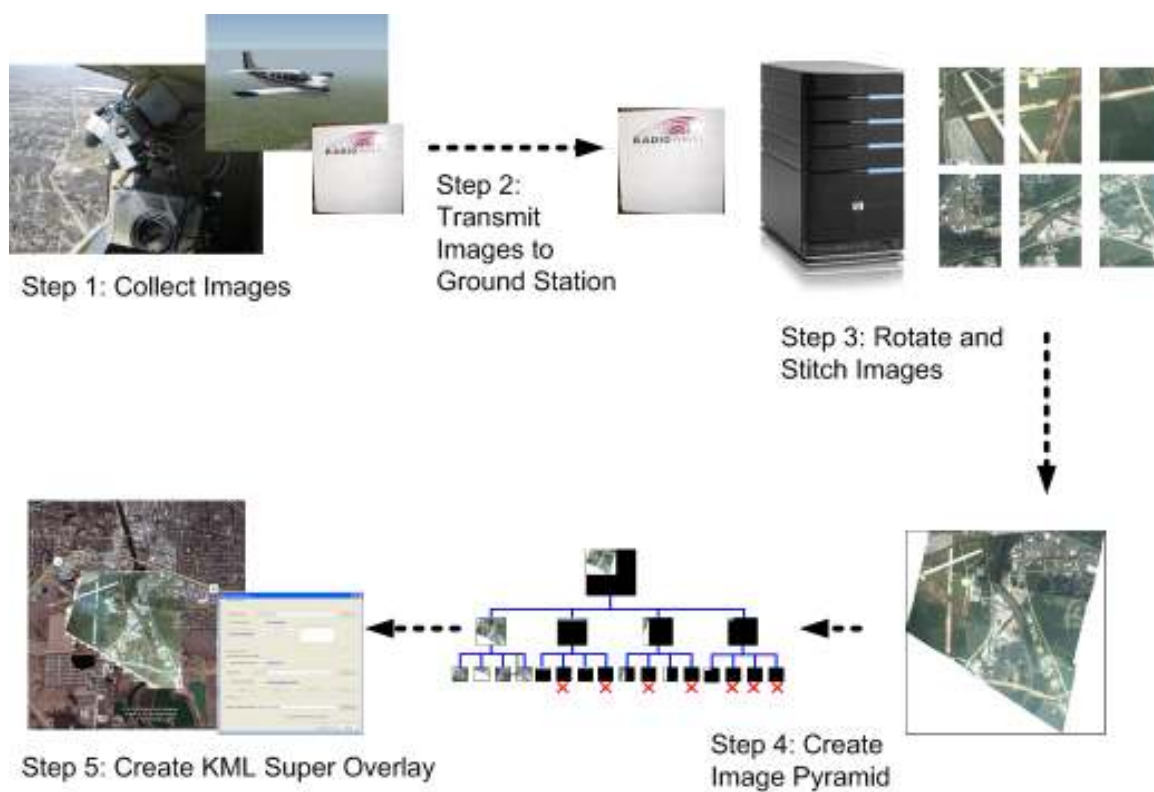


Figure 39: Steps in the image collection process

CHAPTER 7. IMAGING RESULTS

This chapter provides an overview of OAI imaging results. The first section of the chapter provides reference tables for determining the resolution of images and location of the area being imaged relative to the aircraft based on altitude and camera head position. The second section provides an assessment of the placement accuracy of OAI images into Google Earth.

7.1 Image Resolution as a Function of Altitude

Pixel size can be calculated according to Equation 1 given in Chapter 2. Knowing the pixel size allows the user to calculate the altitude and camera head position angle necessary to identify ground features of a certain size. To create a table showing pixel size as a function of altitude it is necessary to examine the geometry of the imaging head relative to the ground. The geometry of image capture can be seen in Figure 40. There are two angles shown in Figure 40. The angle labeled “b” represents the angle between the imaging head and the vertical, known as the camera position angle. The angle labeled “a” represents the fixed field of view of the camera head and is equal to 45 degrees. The altitude of the aircraft is labeled H. On the ground, l is used to denote the distance between the aircraft and the near edge of the imaging area and L is used to denote the distance from the aircraft to the far edge of the imaging area. The length of the imaging area is labeled D. Using the geometric information from Figure 40 and the number of pixels in a finished image, an equation for pixel size can be formulated. The formula for pixel size can be seen in Equation 13.

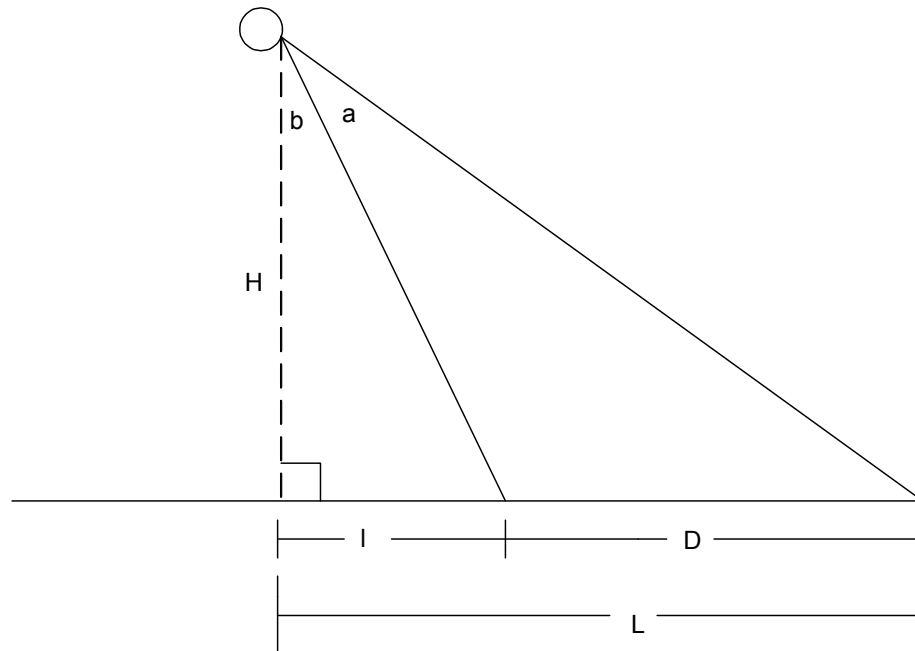


Figure 40: Geometry of Image Capture

$$S_p = H * \frac{(\tan(a+b) - \tan(b))}{N_p} = \frac{2D}{N_p}$$

Equation 13: Calculation of Pixel Size (s_p) Based on Altitude (H) Camera Head View Angle (a) Position Angle of the Camera Head (b) and Number of Pixels (N_p) or Alternatively in Terms of the Ground Distance Imaged by the Camera Head (D)

Equation 13 was used to calculate the resolution of OAI images at altitudes between 250 meters and 10,000 meters with imaging head position angles of 0, 10, 20, 30, 40, and 45 degrees. The altitude range was chosen to include the operating range of low altitude UAVs, small manned aircraft, and high altitude UAVs. Imaging head position angles were chosen to reflect the range where only the ground is included in the image. A camera head position angle of 0 degrees was included even though in the current

system the floor of the aircraft blocks the image at a camera head position angle of 0 degrees. The 0 degrees category is included because future versions of the OAI system may have the camera head mounted on the bottom of the aircraft.

The results of the image resolution calculations can be seen in Table 2 and in Figure 41. Table 2 and Figure 41 show that higher resolution is obtained by maintaining low altitude and a small camera head position angle. Results of Equation 13, which are based on a flat earth model, would indicate an infinitely large distance can be covered by each pixel if the camera head angle is equal to 45 degrees; however this does not occur in practice due to curvature of the Earth's surface. Images taken with a camera head position approaching 45 degrees or beyond 45degrees will have a portion of sky included in the image, as well as a high level of distortion.

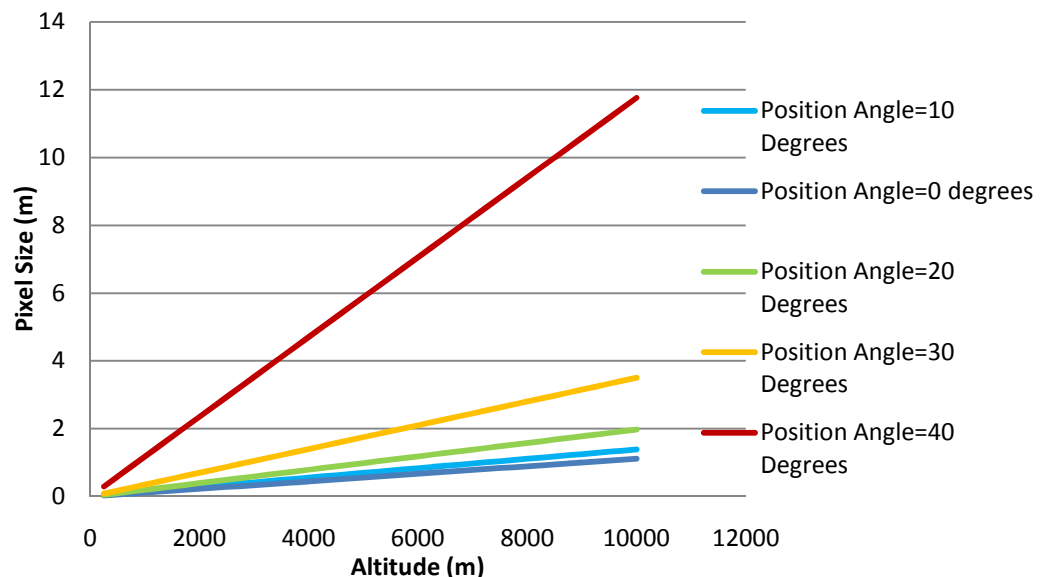


Figure 41: Image Resolution Based on Altitude and Camera Head Position Angle

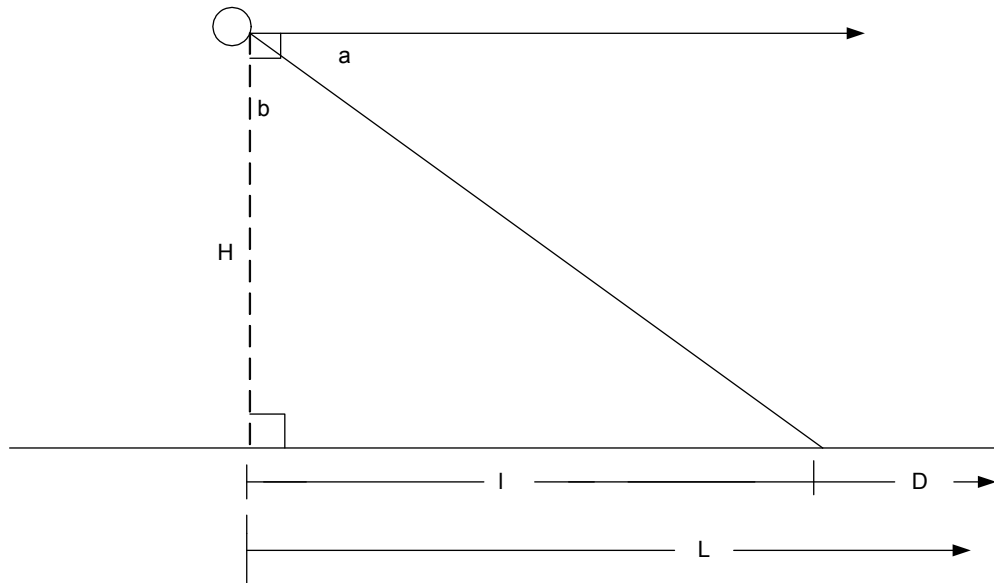


Figure 42: Illustration of a Situation Where Camera View Angle (a) Added to Camera Head Position Angle (b) is Equal to 90 Degrees. The Far Edge of the Imaging Field Never Intersects the Ground.

Table 2: Pixel Size of Image Based on Altitude and Camera Position Angle

Altitude(m)	Camera Head Position Angle (°)				
	0	10	20	30	40
	Pixel Size (m)				
250	0.03	0.03	0.05	0.09	0.29
500	0.06	0.07	0.10	0.18	0.59
750	0.08	0.10	0.15	0.26	0.88
1000	0.11	0.14	0.20	0.35	1.18
1250	0.14	0.17	0.25	0.44	1.47
1500	0.17	0.21	0.30	0.53	1.77
1750	0.19	0.24	0.35	0.61	2.06
2000	0.22	0.28	0.40	0.70	2.35
2250	0.25	0.31	0.45	0.79	2.65
2500	0.28	0.35	0.49	0.88	2.94
2750	0.31	0.38	0.54	0.96	3.24
3000	0.33	0.42	0.59	1.05	3.53
3250	0.36	0.45	0.64	1.14	3.82
3500	0.39	0.49	0.69	1.23	4.12
3750	0.42	0.52	0.74	1.31	4.41
4000	0.44	0.56	0.79	1.40	4.71
4250	0.47	0.59	0.84	1.49	5.00
4500	0.50	0.63	0.89	1.58	5.30
4750	0.53	0.66	0.94	1.66	5.59
5000	0.56	0.70	0.99	1.75	5.88
5250	0.58	0.73	1.04	1.84	6.18
5500	0.61	0.77	1.09	1.93	6.47
5750	0.64	0.80	1.14	2.02	6.77
6000	0.67	0.83	1.19	2.10	7.06
6250	0.69	0.87	1.24	2.19	7.35
6500	0.72	0.90	1.29	2.28	7.65
6750	0.75	0.94	1.34	2.37	7.94
7000	0.78	0.97	1.38	2.45	8.24
7250	0.81	1.01	1.43	2.54	8.53
7500	0.83	1.04	1.48	2.63	8.83
7750	0.86	1.08	1.53	2.72	9.12
8000	0.89	1.11	1.58	2.80	9.41
8250	0.92	1.15	1.63	2.89	9.71
8500	0.94	1.18	1.68	2.98	10.00
8750	0.97	1.22	1.73	3.07	10.30
9000	1.00	1.25	1.78	3.15	10.59
9250	1.03	1.29	1.83	3.24	10.89
9500	1.06	1.32	1.88	3.33	11.18
9750	1.08	1.36	1.93	3.42	11.47
10000	1.11	1.39	1.98	3.51	11.77

Using the geometry shown in Figure 40, the position of the near edge of the image and the far edge of the image relative to the aircraft can be calculated. The location of the near edge can be calculated according to Equation 14 and the far edge can be calculated according to Equation 15.

$$l=H*\tan(b)$$

Equation 14: Calculation of the Location of the Near Edge of the Image Relative to the Aircraft (l) Based on Altitude (H) and Camera Head Position Angle (b)

$$L=H*\tan(a+b)$$

Equation 15: Calculation of the Location of the Far Edge of the Image Relative to the Aircraft (L) Based on Altitude (H), Camera Head Field of View (a) and Camera Head Position Angle (b)

Positions of the near and far edges relative to the aircraft were calculated for altitudes between 250 meters and 10000 meters, and for camera head position angles of 0,10,20,30, and 40 degrees. The results of the calculations can be seen in Table 3. A plot of the distances to the near and far edge for each camera head position angle can be seen in Figure 43.

The graphs in Figure 43 indicate wider field of view is achieved at higher altitudes and with larger camera head position angles. A greater field of view corresponds to a lower image resolution. At a camera head position angle of 0 degrees, the edges are equal distance from the imaging head and are located on both sides of the aircraft. For all other camera head position angles, near and far edges are located on the imaging head side of the aircraft.

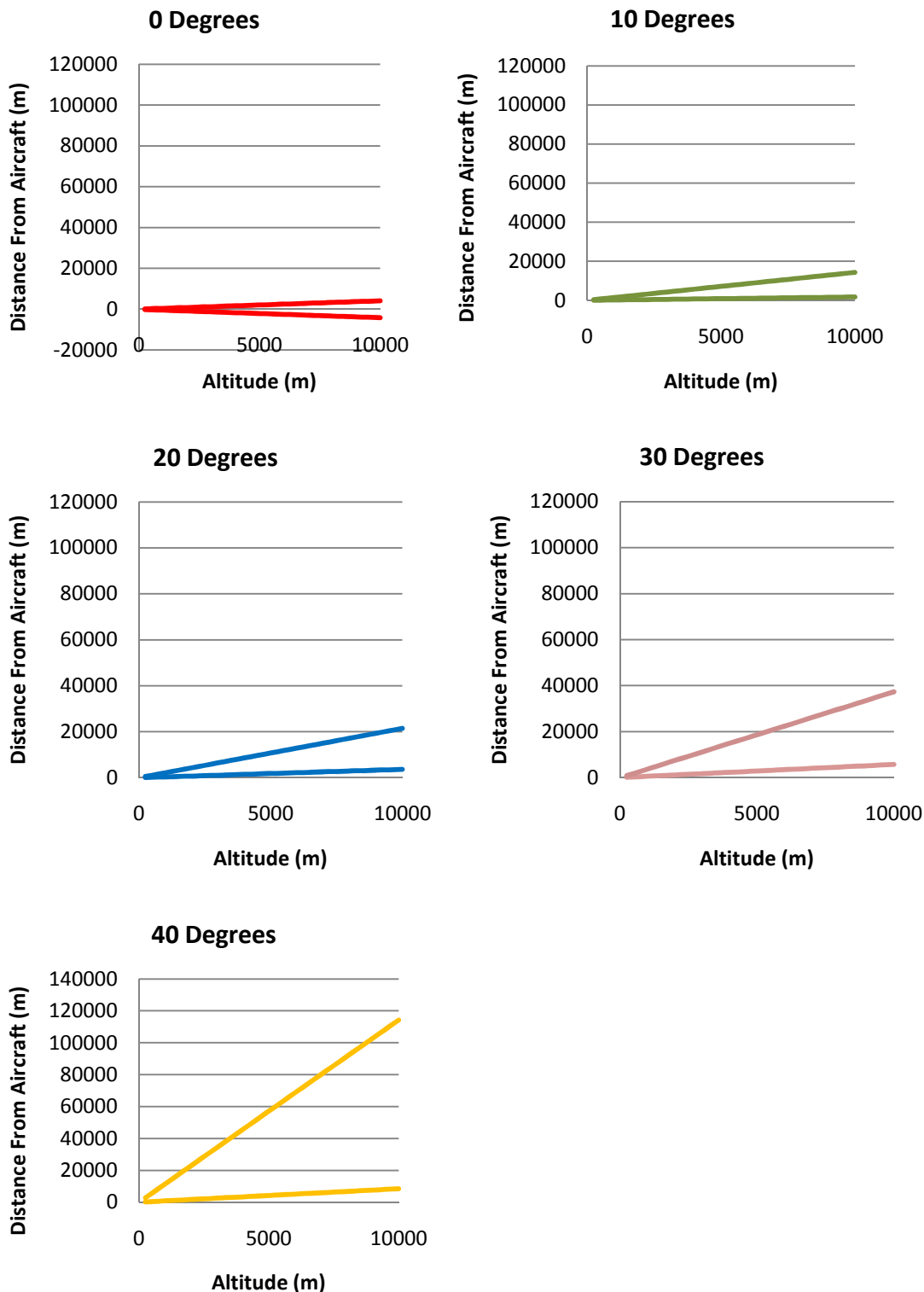


Figure 43: Distance of Near (Bottom Lines) and Far (Top Lines) Edges of Image Area Based on Altitude and Camera Head Position Angle

Table 3: Distance of Near and Far Edges of Image Area from Aircraft at Different Altitudes and Camera Head Position Angles

Altitude(m)	Camera Head Position Angle (°)									
	0		10		20		30		40	
	Near Edge	Far Edge	Near Edge	Far Edge	Near Edge	Far Edge	Near Edge	Far Edge	Near Edge	Far Edge
250	-104	104	44	357	91	536	144	933	210	2858
500	-207	207	88	714	182	1072	289	1866	420	5715
750	-311	311	132	1071	273	1608	433	2799	629	8573
1000	-414	414	176	1428	364	2145	577	3732	839	11430
1250	-518	518	220	1785	455	2681	722	4665	1049	14288
1500	-621	621	264	2142	546	3217	866	5598	1259	17145
1750	-725	725	309	2499	637	3753	1010	6531	1468	20003
2000	-828	828	353	2856	728	4289	1155	7464	1678	22860
2250	-932	932	397	3213	819	4825	1299	8397	1888	25718
2500	-1036	1036	441	3570	910	5361	1443	9330	2098	28575
2750	-1139	1139	485	3927	1001	5897	1588	10263	2308	31433
3000	-1243	1243	529	4284	1092	6434	1732	11196	2517	34290
3250	-1346	1346	573	4641	1183	6970	1876	12129	2727	37148
3500	-1450	1450	617	4999	1274	7506	2021	13062	2937	40005
3750	-1553	1553	661	5356	1365	8042	2165	13995	3147	42863
4000	-1657	1657	705	5713	1456	8578	2309	14928	3356	45720
4250	-1760	1760	749	6070	1547	9114	2454	15861	3566	48578
4500	-1864	1864	793	6427	1638	9650	2598	16794	3776	51435
4750	-1968	1968	838	6784	1729	10186	2742	17727	3986	54293
5000	-2071	2071	882	7141	1820	10723	2887	18660	4195	57150
5250	-2175	2175	926	7498	1911	11259	3031	19593	4405	60008
5500	-2278	2278	970	7855	2002	11795	3175	20526	4615	62865
5750	-2382	2382	1014	8212	2093	12331	3320	21459	4825	65723
6000	-2485	2485	1058	8569	2184	12867	3464	22392	5035	68580
6250	-2589	2589	1102	8926	2275	13403	3608	23325	5244	71438
6500	-2692	2692	1146	9283	2366	13939	3753	24258	5454	74295
6750	-2796	2796	1190	9640	2457	14475	3897	25191	5664	77153
7000	-2899	2899	1234	9997	2548	15012	4041	26124	5874	80010
7250	-3003	3003	1278	10354	2639	15548	4186	27057	6083	82868
7500	-3107	3107	1322	10711	2730	16084	4330	27990	6293	85725
7750	-3210	3210	1367	11068	2821	16620	4474	28923	6503	88583
8000	-3314	3314	1411	11425	2912	17156	4619	29856	6713	91440
8250	-3417	3417	1455	11782	3003	17692	4763	30789	6923	94298
8500	-3521	3521	1499	12139	3094	18228	4907	31722	7132	97155
8750	-3624	3624	1543	12496	3185	18764	5052	32655	7342	100013
9000	-3728	3728	1587	12853	3276	19301	5196	33588	7552	102870
9250	-3831	3831	1631	13210	3367	19837	5340	34521	7762	105728
9500	-3935	3935	1675	13567	3458	20373	5485	35454	7971	108585
9750	-4039	4039	1719	13924	3549	20909	5629	36387	8181	111443
10000	-4142	4142	1763	14281	3640	21445	5774	37321	8391	114301

7.2 Placement Accuracy

A close up of an OAI image loaded into Google Earth can be seen in Figure 44. Google Earth overlays such as street markings, street names, and icons marking businesses can be seen as overlays on the OAI image. The Google Earth features do not line up exactly with the OAI image, but are close enough to their corresponding features in the image that the objects referenced by the Google Earth overlays are readily identifiable.



Figure 44: Screen Shot from Google Earth Showing Google Earth Overlays on OAI Image

In order to quantitatively assess the placement accuracy of the OAI images, 95 ground control points (GCP) were selected from an image. Of these 95 points, only 83 were used for calculation of placement accuracy due to 7 points not appearing in the Google Earth imagery and 5 points being in areas inaccessible to the author for real-world measurements. The latitude and longitude of each GCP in the image was then compared to its position in the underlying Google Earth image and to its position in the real world as measured by a Lowrance Avionics Airmap 500 GPS and WAAS receiver. The GCPs that were selected represent features in the OIA image that were readily identifiable and have a clearly defined boundary that would not change with time. A picture showing the GCPs used can be seen in Figure 45. Most of the points selected are the corners or intersections of concrete structures such as roads, runways, and parking lots. These features make good GCPs because they create well defined points and do not change in position over time. Effort was made to evenly disperse GCPs throughout the image, but some areas have more GCPs than others due to the availability of clearly defined features.



Figure 45: GCPs Used in the Calculation of Image Placement Accuracy. Each GCP is marked with a Red X

The same method was used for measuring the location of the GCPs on the OAI image and the original Google Earth imagery. First, the GCP was located in Google Earth. Once the feature was located, the researcher zoomed in on the feature until the eye altitude value provided in the bottom left corner of the Google Earth display indicated an eye altitude between 200 and 220 meters. The cursor was placed over the GCP and the

latitude and longitude of the point as displayed at the bottom of the Google Earth screen were recorded. At the 200 meter level of zoom, the resolution of the image was sufficient for the boundaries between features to be clear, yet small cursor movements did not change the latitude and longitude readings. Using this method it was possible to obtain measurements as accurate as the image resolution and precision of the Google Earth coordinate display allows. The Google Earth latitude and longitude display provides coordinates to the nearest 0.01 seconds in the degrees-minutes-seconds format, which theoretically translates to an accuracy of 0.000003 degrees or approximately 33 centimeters. In practice, the last digit fluctuates even with a stationary cursor, making the accuracy closer to the nearest 0.05 seconds or approximately 1.65 meters. This estimate of measurement accuracy does not include any errors in placement of Google Earth imagery or OAI imagery.

To measure the latitude and longitude of each GCP in the real world, a Lowrance Avionics Airmap 500 handheld GPS and WAAS receiver unit was used. This unit has a resolution of 0.00001 degrees, or about 1 meter (Lowrance Electronics, 2003). To measure the location of each GCP with the Airmap 500, researchers remained stationary at each GCP until the numbers in the coordinates display stopped changing. The Airmap 500 software warns the user if there are not enough satellites in view to make an accurate measurement, preventing one possible source of experimental error.

Table 4 shows the average difference in latitude and longitude between the OAI image, the Google Earth image, and real world coordinate measurements. The average differences between GCPs are given in both degrees and meters.

Table 4: Image Placement Accuracy

Comparison	Average Degrees Difference Latitude	Average Degrees Difference Longitude	Average Difference Latitude in Meters	Average Difference Longitude in Meters
OAI Image v. Google Earth Image	0.00043	0.0016	48	133
OAI Image v. Real World	0.00018	0.0015	20	128
Google Earth Image v. Real World	0.00032	0.0010	36	87

The placement results for the OAI images versus the Google Earth image are in accordance with the differences in feature placement seen in Figure 44. Visual inspection of the placement of OAI images in Google Earth shows that the features in the two image sets are more closely aligned in the latitude direction than in the longitude direction, and also reveal that features are more closely aligned in some areas than in others. The distortion occurs because the image processing software does not remove all distortions caused by taking pictures with the camera head at an angle, and there are some errors introduced by camera calibration and the GPS devices. Even though the calibration software and GPS devices used for this project are highly accurate, even minuscule errors in either of these systems can be magnified due to the large distance between the imaging equipment and the ground. A probable cause of the greater placement accuracy of the longitudinal direction is that the right-to-left axis of the stitched image has less distortion than the top-to-bottom axis and happens to almost align with the longitudinal direction for this image.

The discrepancy between Google Earth image placement and real world coordinates seen in Table 4 has also been noted in (Potere, 2008), which described an average horizontal

(longitudinal) Root Mean Square Error of 39.7 meters using 436 GCPs worldwide. Since Google does not publish its image placement methods for Google Earth, the source of the error between real world coordinates and Google Earth imagery can only be speculated (Potere, 2008). Some of the error reflected in the difference between the OAI imagery and the Google Earth imagery is due to discrepancies in the placement of Google Earth imagery relative to the real world rather than error incurred by the OAI system.

CHAPTER 8. CONCLUSIONS AND FUTURE WORK

This chapter describes the major conclusions that can be drawn from this project. Short term and long term possibilities for future OAI research are discussed. An overview of possible applications for the OAI system is also provided.

8.1 Conclusions

The OAI system has proved to be successful in the collection of near real time airborne imagery. While image collection is still not truly real time, the capability of the system to load pictures into Google Earth while the imaging system is airborne has been demonstrated. Hardware designed for this project allows equipment to be securely attached to the aircraft while minimizing weight. The new camera head design greatly improved the mobility of the camera head PTU. Operations procedures for the system have been documented and crew roles have been defined.

Operating parameters of the system such as image resolution and boundaries of the imaged area relative to the aircraft have been defined for the operating range of aircraft altitude and camera head position angle. Limitations of the operating range of the camera head have been identified. Table 2 and Table 3 provide the user with a reference for mission planning.

The resolution of the system provides a means for examining a variety of target sizes. At low altitudes resolution of a few centimeters is possible. At the centimeter level of detail, tracking the movement of human beings is possible. Vehicles are visible even at higher altitudes. Using an angled camera head position allows the system to see under overhanging features like trees, bridges, and awnings. The angled viewing is a key

advantage over surveillance systems that view areas from directly above because it reveals areas that would otherwise be hidden.

Image placement in Google Earth was less accurate than expected. However, placement is accurate enough that features referred to by the Google Earth overlays for streets and place markers are readily identifiable in the OAI imagery. Much of the error in image placement is the result of small errors in calibration and GPS measurements being magnified when they are applied over the great distance between the imaging head and the ground. Imperfections in the stitching software also contribute to placement error.

8.2 Future Work

8.2.1 The Next Phase of OAI Development

The next phase of development for the airborne imaging system will be a modification of the system software. The goal of the software modification project is to achieve real time upload of images into the Google Earth viewer at a rate of two frames per second. A more thorough investigation of the mechanisms of placement error will be made in order to find ways to improve the placement accuracy of the final images. The KML super overlay generator program will be altered to operate more quickly and to automatically create KML files and load the newest image in Google Earth. The alterations to the KML super overlay generator will allow images to be viewed faster because the KML files will be created without waiting for user input. The image processing software, PTU control software, and calibration software will be altered to speed up data transmission and reduce placement error if possible. Additional flight tests are scheduled for the Sumer of 2010.

Another change to the OAI system that might be made in the near future is the addition of infrared spectrum cameras to the camera head. If a combination of visible spectrum and infrared spectrum cameras are used, both daytime and nighttime imaging will be possible.

8.2.2 Future Applications of the OAI System

. One promising application of the OAI system is its use in unmanned surveillance missions. The high resolution of the low altitude images makes the OAI system a good candidate for monitoring the movements of individuals and vehicles. The ability to program the camera PTU to continuously point the camera head at a ground target is another useful feature for surveillance operations. The orthorectified, near real time display of images in Google Earth with supplemental information in the form of Google Earth feature marking overlays makes the images readily understandable. If object identification and tracking options were added to the image processing software package, the resulting surveillance toolset could become a powerful asset for the military and law enforcement agencies. Addition of infrared spectrum cameras would further expand the utility of an OAI surveillance system by allowing for nighttime operations.

The OAI system could also be applied to natural disaster monitoring. In the summer of 2008, flooding of the Iowa River and the Cedar River caused extensive damage in Cedar Rapids, Iowa and Iowa City, Iowa. Many evacuated residents were left wondering about the state of their homes. Use of the OAI system would allow images to be captured and loaded into an online data base as a natural disaster such as the floods of 2008 occurs, allowing damage to be assessed without risk to individuals. Likewise, the OAI system could be used to monitor wildfires. The addition of infrared cameras would

be useful for firefighting use because they can be used to detect hotspots and can “see” through clouds of smoke. The high resolution of the images and possible addition of infrared cameras would make the OAI system useful in search and rescue missions as well.

8.3 Closing Remarks

The OAI system presented in this report provides users with a new way to view high resolution airborne images in real time. Parameters such as image resolution and position of imaging area have been defined for the OAI system, and can serve as a reference for future studies. Image placement error has been measured, and a second phase of development has been planned. The OAI system has many potential applications, and if further developed may one day provide important airborne imaging support for both military and civilian operations.

CHAPTER 9. WORKS CITED

- Advanced Infoneering, Inc. (n.d.). *Synthetic Flight Bag*. Retrieved 2010 23-January from Advanced Infoneering Inc.: <http://www.advancedinfoneering.com/sfb/SFB.pdf>
- Aerial Photography Services. (2008). *Equipment and Services*. Retrieved 2009 31-December from Aerial Photography Services: <http://www.aerialphs.com/>
- Aero Antenna Technology Inc. (2008 йил 28-7). AT2775-81. Aero Antenna Technology Inc.
- Airborne Surveillance. (2010). *Surveillance by the Hour*. Retrieved 2010 йил 2-January from Airborne Surveillance: <http://www.airborne surveillance.com>
- AME Info. (2005 йил 12-February). *Airborne surveillance isn't just a military role*. Retrieved 2010 5-January from AME.com: <http://www.ameinfo.com/53684.html>
- AT Electronic and Communication International. (2009 December). *UAV Product Profile*. Retrieved 2010 2-January from AT Electronic and Communication International: http://at-communication.com/upload/File/AT_UAV_Brochure.pdf
- Committee on Science and Technology. (2008 7-April). Remote Sensing Data: Applications and Benefits. *Field Hearing before the Subcommittee on Space and Aeronautics Committee on Science and Technology House of Representatives One Hundred Tenth Congress Second Session , Serial No. 110-91* . Washington DC, District of Colombia: U.S. Government Printing Office.
- Defense Update. (2005, August 23). *Dragon Eye Miniature UAV*. Retrieved January 20, 2010, from Defense Update; International Online Defense Magazine: <http://defense-update.com/products/d/dragoneyes.htm>
- Directed Perception. (n.d.). *Computer Controlled Pan-Tilt Unit Model PTU-300 Users Manual*. Retrieved 2009 19-February from Kane Computing Limited: http://www.kanecomputing.co.uk/ip_ptu_d300.htm
- Esposito, F., Rufino, G., & Moccia, A. (2007). 1st Mini-UAV Integrated Hyperspectral?Thermal Electro-Optical Payload for Forest Fire Risk Management. *Infotech@Aerospace Conference and Exhibit* (pp. 1-8). Rohnert Park, California: American Institute of Aeronautics and Astronautics.
- Esposito, F., Rugino, G., & Moccia, A. (2009). Real-Time Detection of Fire Hotspots from Mini-UAV Based, Thermal InfraRed/VIS-NIR Hyperspectral Image Data. *AIAA Infotec@Aerospace Conference* (pp. 1-18). Seattle, Washington : American Institute of Aeronautics and Astronautics.
- Finnegan, T. J. (2006 October). *Shooting the Front: Allied Aerial Reconosance and Photographic Interpretation on the Western Front-World War I*. Washington DC, District of Colombia: NDIC Press.

- Fly By Photos. (2006). *Rates and Services*. Retrieved 2009 29-December from Fly By Photos: Aerial Photography and Images: <http://www.flybyphotos.com/rates.html>
- Gao, J. (2009). *Digital Analysis of Remotely Sensed Imagery*. New York: Mc Graw Hill.
- GeoEye. (2010). *About GeoEye-1*. Retrieved 2010 3-January from GeoEye: <http://launch.geoeye.com/LaunchSite/about/>
- GlobalSecurity.org. (2005, April 26). *Eagle Eye UAV*. Retrieved January 17, 2010, from GlobalSecurity.org: <http://www.globalsecurity.org/intell/systems/eagle-eye.htm>
- Google. (2009). Google Earth 5.0.
- Google. (2010). *Google Earth Free*. Retrieved 2010 11-January from Google Earth: <http://earth.google.com/>
- Google. (2010). *Google Milestones*. Retrieved 2010 11-January from Google Corporate Information: <http://www.google.com/intl/en/corporate/history.html>
- Google. (n.d.). *Length of a Degree of Latitude and Longitude Calculator*. Retrieved March 19, 2010, from Length of a Degree of Latitude and Longitude Calculator: <http://www.csgnetwork.com/degreenllavcalc.html>
- Illunis LLC. (n.d.). *Illunis XMV-16000 Product Brief*. Retrieved 2010 14-January from Illunis: www.illunis.com
- Kumar, M., & Cohen, K. (2009). Wild Land Fire Fighting using Multiple Uninhabited Aerial. *AIAA Unmanned...Unlimited Conference*. Seattle, Washington: AIAA.
- LaFranchi, P. (2006 9-November). *MicroPilot Unveils own agricultural UAV to help shape civil market evolution*. Retrieved 2010 4-January from FlightGlobal: <http://www.flightglobal.com/articles/2006/09/11/208933/micropilot-unveils-own-agricultural-uav-to-help-shape-civil-market.html>
- Lowrance Electronics. (2003). *AirMap 500 Handheld Mapping GPS Receiver Operation Instructions*. Retrieved March 21, 2010, from <http://reviews.gpsfaq.com/downloads/lowranceairmap500manual.pdf>
- Mazzone, J. (2009). Retrieved 2009 29-December from Ace Aerial Photography: <http://www.aceaerialphoto.us/>
- Motorola. (n.d.). *MOTOROLA Canopy 2.4GHz Advantage AP with AES*. Retrieved 2010 5-February from Tescoco: <http://www.tesco.com/products/displayProductInfo.do?sku=415841&eventPage=1>
- NASA. (2008, March 1). *Aerial Camera Systems*. Retrieved March 19, 2010, from Dryden Flight Research Center: <http://www.nasa.gov/centers/dryden/research/AirSci/ER-2/cameras.html>

National Aeronautics and Space Administration. (2008). *An Overview of SeaWiFs and the SeaStar Spacecraft*. Retrieved 2010 5-January from SeaWiFs Project Information: <http://oceancolor.gsfc.nasa.gov/SeaWiFS/SEASTAR/SPACECRAFT.html>

National Aeronautics and Space Administration. (2009 18-August). *The Landsat Program*. Retrieved 2009 23-August from National Aeronautics and Space Administration: <http://landsat.gsfc.nasa.gov/about/science.html>

Northrop Grumman. (2010). *Airborne Surveillance*. Retrieved 2010 2-January from Northrop Grumman: http://www.es.northropgrumman.com/by_capability/militaryaviation/surveillance

Northrop Grumman. (2010). *Night Hunter II*. Retrieved 2010 1-January from Northrop Grumman: <http://www.northropgrumman.com/solutions/nighthunter2/index.html>

Novatel. (2005 22-December). *DL4+ Users Manual*. Retrieved 2010 16-January from <http://geodesy.eng.ohio-state.edu/course/gs609/notes/DL4%20Novatel%20GPS%20Manual.pdf>

Novatel. (2006). *DL4-Plus*. Retrieved 2010 11-January from Navtech GPS: <http://www.navtechgps.com/Downloads/DL4plus.pdf>

Novatel. (n.d.). *Updated Product Guide*. Retrieved 2010 4-February from Novatel: http://www.novatel.com/Documents/Papers/novatel_productguide_web.pdf

Panasonic. (n.d.). *Fully Rugged Toughbook 30*. Retrieved February 22, 2010, from BuyTough: http://www.buytough.com/tb_30.asp

Phidget Inc. (2009 14-August). *1203 Phidget Text LCD with PhidgetInterfaceKit 8/8/8*. Retrieved 2010 17-January from Phidget Inc: <http://www.phidgets.com/documentation/Phidgets/1203.pdf>

Potere, D. (2008). Horizontal Positional Accuracy of Google Earth's High Resolution Imagery Archive. *Sensors*, 7973-7981.

Professional Aerial Photographers Association. (2009). *History of Aerial Photography*. Retrieved 2009 15-October from Professional Aerial Photographers Association: <http://www.papainternational.org/history.html>

Rasmussen, N. D., Morse, B. S., & Taylor, C. N. (2007). A Fixed-Wing, Mini-UAV System for Aerial Search Operations. *AIAA Guidance, Navigation and Control Conference and Exhibit* (pp. 1-8). Hilton Head, South Carolina: American Institute of Aeronautics and Astronautics.

Sauser, B. (2007 31-August). *Mapping Wildfires*. Retrieved 2009 17-October from Technology Review: <http://www.technologyreview.com/computing/19324/?a=f>

United States Air Force. (2009 18-May). *United States Air Force Unmanned Systems Flight Plan 2009*. Retrieved 2010 5-January from United States Air Force:
http://www.ras.org/irp/program/collect/uas_2009.pdf

United States Congress. (1992). *United States Code Title 15 Chapter 82 Land Remote Sensing Policy*. Retrieved 2009 23-August from <http://geo.arc.nasa.gov/sge/landsat/15USCch82.html>

University College London. (2006). *Google Earth Photo Overlay Creator*. Retrieved 2010 20-2 from UCL Centre for Advanced Spatial Analysis:
<http://www.casa.ucl.ac.uk/software/photooverlaycreator.asp>

Wernecke, J. (2009). *The KML Handbook*. Upper Saddle River, NJ: Addison-Wesley.

APPENDIX A:KML SUPREOVERLAY CODE EXAMPLES

A.1 Example Top Level File for Super Overlay

```

<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <NetworkLink>
    <name>SuperOverlay: Iowa City</name>
    <Region>
      <LatLonAltBox>
        <north>41.644679</north>
        <south>41.606971</south>
        <east>-91.501985</east>
        <west>-91.551375</west>
      </LatLonAltBox>
      <Lod>
        <minLodPixels>128</minLodPixels>
        <maxLodPixels>-1</maxLodPixels>
      </Lod>
    </Region>
    <Link>
      <href>0_0_0.kml</href>
      <viewRefreshMode>onRegion</viewRefreshMode>
    </Link>
  </NetworkLink>
</kml>

```

A.2 Example of Mid-Level KML Super Overlay File

```

<?xmlversion="1.0"encoding="UTF-8"?>
<!--KMLRegionator-->
<kmlxmlns="http://earth.google.com/kml/2.1">
<Document>
  <Region>
    <Lod>
      <minLodPixels>128</minLodPixels>
      <maxLodPixels>-1</maxLodPixels>
    </Lod>
    <LatLonAltBox>
      <north>41.644679</north>
      <south>41.606971</south>
      <east>-91.501985</east>
      <west>-91.551375</west>
    </LatLonAltBox>
  </Region>
  <NetworkLink>
    <name>1_0_0</name>
    <Region>
      <Lod>
        <minLodPixels>128</minLodPixels>
        <maxLodPixels>-1</maxLodPixels>
      </Lod>
      <LatLonAltBox>
        <north>41.644679</north>
        <south>41.625825</south>
        <east>-91.526680</east>
        <west>-91.551375</west>
      </LatLonAltBox>
    </Region>
    <Link>
      <href>1_0_0.kml</href>
      <viewRefreshMode>onRegion</viewRefreshMode>
    </Link>
  </NetworkLink>
  <NetworkLink>
    <name>1_0_1</name>
    <Region>
      <Lod>
        <minLodPixels>128</minLodPixels>
        <maxLodPixels>-1</maxLodPixels>
      </Lod>
      <LatLonAltBox>
        <north>41.625825</north>
        <south>41.606971</south>
        <east>-91.526680</east>
        <west>-91.551375</west>
      </LatLonAltBox>
    </Region>
  </NetworkLink>
</Document>

```

```

        </LatLonAltBox>
    </Region>
    <Link>
        <href>1_0_1.kml</href>
        <viewRefreshMode>onRegion</viewRefreshMode>
    </Link>
</NetworkLink>
<NetworkLink>
    <name>1_1_0</name>
    <Region>
        <Lod>
            <minLodPixels>128</minLodPixels>
            <maxLodPixels>-1</maxLodPixels>
        </Lod>
        <LatLonAltBox>
            <north>41.644679</north>
            <south>41.625825</south>
            <east>-91.501985</east>
            <west>-91.526680</west>
        </LatLonAltBox>
    </Region>
    <Link>
        <href>1_1_0.kml</href>
        <viewRefreshMode>onRegion</viewRefreshMode>
    </Link>
</NetworkLink>
<NetworkLink>
    <name>1_1_1</name>
    <Region>
        <Lod>
            <minLodPixels>128</minLodPixels>
            <maxLodPixels>-1</maxLodPixels>
        </Lod>
        <LatLonAltBox>
            <north>41.625825</north>
            <south>41.606971</south>
            <east>-91.501985</east>
            <west>-91.526680</west>
        </LatLonAltBox>
    </Region>
    <Link>
        <href>1_1_1.kml</href>
        <viewRefreshMode>onRegion</viewRefreshMode>
    </Link>
</NetworkLink>
<GroundOverlay>
    <drawOrder>2</drawOrder>
    <Icon>
        <href>http://oplsver.o.pl.uiowa.edu/airport_images/total-
tiles/total_0_0_0.jpg</href>

```

```
</Icon>
<LatLonBox>
  <north>41.644679</north>
  <south>41.606971</south>
  <east>-91.501985</east>
  <west>-91.551375</west>
</LatLonBox>
</GroundOverlay>
</Document>
</kml>
```

A.3 Example of Level 1 KML File

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- KML Regionator -->
<kml xmlns="http://earth.google.com/kml/2.1">
<Document>
  <Region>
    <Lod>
      <minLodPixels>128</minLodPixels>
      <maxLodPixels>-1</maxLodPixels>
    </Lod>
    <LatLonAltBox>
      <north>41.644679</north>
      <south>41.606971</south>
      <east>-91.501985</east>
      <west>-91.551375</west>
    </LatLonAltBox>
  </Region>
  <GroundOverlay>
    <drawOrder>3</drawOrder>
    <Icon>
      <href>http://oplsrver.opl.uiowa.edu/airport_images/total-
tiles/total_1_0_0.jpg</href>
    </Icon>
    <LatLonBox>
      <north>41.644679</north>
      <south>41.625825</south>
      <east>-91.526680</east>
      <west>-91.551375</west>
    </LatLonBox>
  </GroundOverlay>
</Document>
</kml>

```


APPENDIX B: C# CODE FOR KML SUPER OVERLAY GENERATOR GUI

```

namespace OAI_Superoverlay
{
    partial class frmSuperOverlayCreator
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise,
        false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.lblPromtImage = new System.Windows.Forms.Label();
            this.btnSelectImage = new System.Windows.Forms.Button();
            this.ofdSelectImage = new System.Windows.Forms.OpenFileDialog();
            this.lblPromptCoordinates = new System.Windows.Forms.Label();
            this.lblNorth = new System.Windows.Forms.Label();
            this.lblSouth = new System.Windows.Forms.Label();
            this.lblEast = new System.Windows.Forms.Label();
            this.lblWest = new System.Windows.Forms.Label();
            this.txtNorth = new System.Windows.Forms.TextBox();
            this.txtSouth = new System.Windows.Forms.TextBox();
            this.txtEast = new System.Windows.Forms.TextBox();
            this.txtWest = new System.Windows.Forms.TextBox();
            this.chkUsePng = new System.Windows.Forms.CheckBox();
            this.btnWhyPng = new System.Windows.Forms.Button();
            this.lblTransparency = new System.Windows.Forms.Label();
            this.csdTransparencyColor = new System.Windows.Forms.ColorDialog();
            this.btnAddColors = new System.Windows.Forms.Button();
            this.lstbxTransparency = new System.Windows.Forms.ListBox();
            this.lblSelectImagePyramidDestination = new System.Windows.Forms.Label();
        }
    }
}

```

```

this.btnSelectDestination = new System.Windows.Forms.Button();
this.lblSaveKML = new System.Windows.Forms.Label();
this.btnKMLFileDestination = new System.Windows.Forms.Button();
this.btnCreateKML = new System.Windows.Forms.Button();
this.btnClearAll = new System.Windows.Forms.Button();
this.txtImageFile = new System.Windows.Forms.TextBox();
this.txtImagePyramidDestination = new System.Windows.Forms.TextBox();
this.txtKMLFileDestination = new System.Windows.Forms.TextBox();
this.chkboxLoadKML = new System.Windows.Forms.CheckBox();
this.lblListTransperencies = new System.Windows.Forms.Label();
this.grpbxImageOptions = new System.Windows.Forms.GroupBox();
this.btnAboutTransperencies = new System.Windows.Forms.Button();
this.btnRemoveColor = new System.Windows.Forms.Button();
this.grpbxCoordinates = new System.Windows.Forms.GroupBox();
this.txtGeoFile = new System.Windows.Forms.TextBox();
this.btnFindGeo = new System.Windows.Forms.Button();
this.lblBrowseGeo = new System.Windows.Forms.Label();
this.rdbtnEnterManually = new System.Windows.Forms.RadioButton();
this.rdbtnGetFromGeo = new System.Windows.Forms.RadioButton();
this.btnAboutGeo = new System.Windows.Forms.Button();
this.btnAboutLatLon = new System.Windows.Forms.Button();
this.grpbxKMLOptions = new System.Windows.Forms.GroupBox();
this.brwsrImagePyramidDestination = new System.Windows.Forms.FolderBrowserDialog();
this.brwsrKMLDestination = new System.Windows.Forms.FolderBrowserDialog();
this.ofdSelectGeo = new System.Windows.Forms.OpenFileDialog();
this.grpbxImageOptions.SuspendLayout();
this.grpbxCoordinates.SuspendLayout();
this.grpbxKMLOptions.SuspendLayout();
this.SuspendLayout();
//
// lblPromtImage
//
this.lblPromtImage.AutoSize = true;
this.lblPromtImage.Location = new System.Drawing.Point(52, 51);
this.lblPromtImage.Name = "lblPromtImage";
this.lblPromtImage.Size = new System.Drawing.Size(69, 13);
this.lblPromtImage.TabIndex = 0;
this.lblPromtImage.Text = "Select Image";
this.lblPromtImage.Click += new System.EventHandler(this.lblPromtImage_Click);
//
// btnSelectImage
//
this.btnSelectImage.Location = new System.Drawing.Point(705, 44);
this.btnSelectImage.Name = "btnSelectImage";
this.btnSelectImage.Size = new System.Drawing.Size(107, 26);
this.btnSelectImage.TabIndex = 2;
this.btnSelectImage.Text = "Browse Files";
this.btnSelectImage.UseVisualStyleBackColor = true;
this.btnSelectImage.Click += new System.EventHandler(this.btnSelectImage_Click);
//
// ofdSelectImage
//
this.ofdSelectImage.Filter = "Filter Windows Bitmaps|*.BMP|JPEG Files|*.JPG ";
this.ofdSelectImage.Title = "Select Image";
this.ofdSelectImage.FileOk += new
System.ComponentModel.CancelEventHandler(this.ofdSelectImage_FileOk);

```

```

//
// lblPromptCoordinates
//
this.lblPromptCoordinates.AutoSize = true;
this.lblPromptCoordinates.Location = new System.Drawing.Point(9, 26);
this.lblPromptCoordinates.Name = "lblPromptCoordinates";
this.lblPromptCoordinates.Size = new System.Drawing.Size(203, 13);
this.lblPromptCoordinates.TabIndex = 3;
this.lblPromptCoordinates.Text = "Enter Latitude and Longitude Coordinates";
//
// lblNorth
//
this.lblNorth.AutoSize = true;
this.lblNorth.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.lblNorth.Location = new System.Drawing.Point(50, 493);
this.lblNorth.Name = "lblNorth";
this.lblNorth.Size = new System.Drawing.Size(33, 13);
this.lblNorth.TabIndex = 4;
this.lblNorth.Text = "North";
//
// lblSouth
//
this.lblSouth.AutoSize = true;
this.lblSouth.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.lblSouth.Location = new System.Drawing.Point(248, 493);
this.lblSouth.Name = "lblSouth";
this.lblSouth.Size = new System.Drawing.Size(35, 13);
this.lblSouth.TabIndex = 5;
this.lblSouth.Text = "South";
//
// lblEast
//
this.lblEast.AutoSize = true;
this.lblEast.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.lblEast.Location = new System.Drawing.Point(470, 493);
this.lblEast.Name = "lblEast";
this.lblEast.Size = new System.Drawing.Size(28, 13);
this.lblEast.TabIndex = 6;
this.lblEast.Text = "East";
//
// lblWest
//
this.lblWest.AutoSize = true;
this.lblWest.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.lblWest.Location = new System.Drawing.Point(668, 494);
this.lblWest.Name = "lblWest";
this.lblWest.Size = new System.Drawing.Size(32, 13);
this.lblWest.TabIndex = 7;
this.lblWest.Text = "West";
//
// txtNorth
//
this.txtNorth.Enabled = false;
this.txtNorth.ForeColor = System.Drawing.SystemColors.InactiveBorder;
this.txtNorth.Location = new System.Drawing.Point(107, 490);
this.txtNorth.Name = "txtNorth";

```

```

this.txtNorth.Size = new System.Drawing.Size(100, 20);
this.txtNorth.TabIndex = 10;
this.txtNorth.Text = "00.0000";
//
// txtSouth
//
this.txtSouth.Enabled = false;
this.txtSouth.ForeColor = System.Drawing.SystemColors.InactiveBorder;
this.txtSouth.Location = new System.Drawing.Point(308, 490);
this.txtSouth.Name = "txtSouth";
this.txtSouth.Size = new System.Drawing.Size(100, 20);
this.txtSouth.TabIndex = 11;
this.txtSouth.Text = "00.0000";
//
// txtEast
//
this.txtEast.Enabled = false;
this.txtEast.ForeColor = System.Drawing.SystemColors.InactiveBorder;
this.txtEast.Location = new System.Drawing.Point(518, 490);
this.txtEast.Name = "txtEast";
this.txtEast.Size = new System.Drawing.Size(100, 20);
this.txtEast.TabIndex = 12;
this.txtEast.Text = "00.0000";
//
// txtWest
//
this.txtWest.Enabled = false;
this.txtWest.ForeColor = System.Drawing.SystemColors.InactiveBorder;
this.txtWest.Location = new System.Drawing.Point(715, 491);
this.txtWest.Name = "txtWest";
this.txtWest.Size = new System.Drawing.Size(100, 20);
this.txtWest.TabIndex = 13;
this.txtWest.Text = "00.0000";
//
// chkUsePng
//
this.chkUsePng.AutoSize = true;
this.chkUsePng.Location = new System.Drawing.Point(58, 123);
this.chkUsePng.Name = "chkUsePng";
this.chkUsePng.Size = new System.Drawing.Size(123, 17);
this.chkUsePng.TabIndex = 14;
this.chkUsePng.Text = "Use .png File Format";
this.chkUsePng.UseVisualStyleBackColor = true;
this.chkUsePng.CheckedChanged += new
System.EventHandler(this.chkUsePng_CheckedChanged);
//
// btnWhyPng
//
this.btnWhyPng.Location = new System.Drawing.Point(251, 121);
this.btnWhyPng.Name = "btnWhyPng";
this.btnWhyPng.Size = new System.Drawing.Size(127, 23);
this.btnWhyPng.TabIndex = 15;
this.btnWhyPng.Text = "Why use .png Format?";
this.btnWhyPng.UseVisualStyleBackColor = true;
this.btnWhyPng.Click += new System.EventHandler(this.btnWhyPng_Click);
//

```

```

// lblTransparency
//
this.lblTransparency.AutoSize = true;
this.lblTransparency.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.lblTransparency.Location = new System.Drawing.Point(49, 164);
this.lblTransparency.Name = "lblTransparency";
this.lblTransparency.Size = new System.Drawing.Size(142, 13);
this.lblTransparency.TabIndex = 16;
this.lblTransparency.Text = "Select Color Transpariencies";
this.lblTransparency.Click += new System.EventHandler(this.lblTransparency_Click);
//
// btnAddColors
//
this.btnAddColors.Enabled = false;
this.btnAddColors.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.btnAddColors.Location = new System.Drawing.Point(253, 159);
this.btnAddColors.Name = "btnAddColors";
this.btnAddColors.Size = new System.Drawing.Size(195, 23);
this.btnAddColors.TabIndex = 17;
this.btnAddColors.Text = "Add Color To List";
this.btnAddColors.UseVisualStyleBackColor = true;
this.btnAddColors.Click += new System.EventHandler(this.btnAddColors_Click);
//
// lstbxTransparency
//
this.lstbxTransparency.CausesValidation = false;
this.lstbxTransparency.ForeColor = System.Drawing.SystemColors.InactiveBorder;
this.lstbxTransparency.FormattingEnabled = true;
this.lstbxTransparency.Items.AddRange(new object[] {
    "Black",
    "White"});
this.lstbxTransparency.Location = new System.Drawing.Point(477, 159);
this.lstbxTransparency.Name = "lstbxTransparency";
this.lstbxTransparency.ScrollAlwaysVisible = true;
this.lstbxTransparency.Size = new System.Drawing.Size(209, 82);
this.lstbxTransparency.TabIndex = 18;
this.lstbxTransparency.SelectedIndexChanged += new
System.EventHandler(this.lstbxTransparency_SelectedIndexChanged);
//
// lblSelectImagePyramidDestination
//
this.lblSelectImagePyramidDestination.AutoSize = true;
this.lblSelectImagePyramidDestination.Location = new System.Drawing.Point(49, 82);
this.lblSelectImagePyramidDestination.Name = "lblSelectImagePyramidDestination";
this.lblSelectImagePyramidDestination.Size = new System.Drawing.Size(165, 13);
this.lblSelectImagePyramidDestination.TabIndex = 19;
this.lblSelectImagePyramidDestination.Text = "Select Image Pyramid Destination";
this.lblSelectImagePyramidDestination.Click += new
System.EventHandler(this.lblSelectImagePyramidDestination_Click);
//
// btnSelectDestination
//
this.btnSelectDestination.Location = new System.Drawing.Point(705, 76);
this.btnSelectDestination.Name = "btnSelectDestination";
this.btnSelectDestination.Size = new System.Drawing.Size(107, 26);
this.btnSelectDestination.TabIndex = 20;

```

```

this.btnSelectDestination.Text = "Browse Files";
this.btnSelectDestination.UseVisualStyleBackColor = true;
this.btnSelectDestination.Click += new System.EventHandler(this.btnSelectDestination_Click);
//
// lblSaveKML
//
this.lblSaveKML.AutoSize = true;
this.lblSaveKML.Location = new System.Drawing.Point(36, 585);
this.lblSaveKML.Name = "lblSaveKML";
this.lblSaveKML.Size = new System.Drawing.Size(183, 13);
this.lblSaveKML.TabIndex = 22;
this.lblSaveKML.Text = "Select KML Superoverlay Destination";
//
// btnKMLFileDestination
//
this.btnKMLFileDestination.Location = new System.Drawing.Point(678, 44);
this.btnKMLFileDestination.Name = "btnKMLFileDestination";
this.btnKMLFileDestination.Size = new System.Drawing.Size(107, 26);
this.btnKMLFileDestination.TabIndex = 24;
this.btnKMLFileDestination.Text = "Browse Files";
this.btnKMLFileDestination.UseVisualStyleBackColor = true;
this.btnKMLFileDestination.Click += new
System.EventHandler(this.btnKMLFileDestination_Click);
//
// btnCreateKML
//
this.btnCreateKML.Location = new System.Drawing.Point(558, 680);
this.btnCreateKML.Name = "btnCreateKML";
this.btnCreateKML.Size = new System.Drawing.Size(129, 38);
this.btnCreateKML.TabIndex = 25;
this.btnCreateKML.Text = "Create KML Files";
this.btnCreateKML.UseVisualStyleBackColor = true;
this.btnCreateKML.Click += new System.EventHandler(this.btnCreateKML_Click);
//
// btnClearAll
//
this.btnClearAll.Location = new System.Drawing.Point(696, 680);
this.btnClearAll.Name = "btnClearAll";
this.btnClearAll.Size = new System.Drawing.Size(122, 38);
this.btnClearAll.TabIndex = 26;
this.btnClearAll.Text = "Clear All";
this.btnClearAll.UseVisualStyleBackColor = true;
this.btnClearAll.Click += new System.EventHandler(this.btnClearAll_Click);
//
// txtImageFile
//
this.txtImageFile.Location = new System.Drawing.Point(225, 48);
this.txtImageFile.Name = "txtImageFile";
this.txtImageFile.Size = new System.Drawing.Size(461, 20);
this.txtImageFile.TabIndex = 27;
this.txtImageFile.Text = "Select Image";
this.txtImageFile.TextChanged += new System.EventHandler(this.txtImageFile_TextChanged);
//
// txtImagePyramidDestination
//
this.txtImagePyramidDestination.Location = new System.Drawing.Point(225, 79);

```

```

this.txtImagePyramidDestination.Name = "txtImagePyramidDestination";
this.txtImagePyramidDestination.Size = new System.Drawing.Size(461, 20);
this.txtImagePyramidDestination.TabIndex = 28;
this.txtImagePyramidDestination.Text = "Select Image Pyramid Destination";
//
// txtKMLFileDestination
//
this.txtKMLFileDestination.Location = new System.Drawing.Point(198, 48);
this.txtKMLFileDestination.Name = "txtKMLFileDestination";
this.txtKMLFileDestination.Size = new System.Drawing.Size(461, 20);
this.txtKMLFileDestination.TabIndex = 29;
this.txtKMLFileDestination.Text = "Select KML File Destination";
//
// chkbxLoadKML
//
this.chkbxLoadKML.AutoSize = true;
this.chkbxLoadKML.Location = new System.Drawing.Point(392, 638);
this.chkbxLoadKML.Name = "chkbxLoadKML";
this.chkbxLoadKML.Size = new System.Drawing.Size(249, 17);
this.chkbxLoadKML.TabIndex = 30;
this.chkbxLoadKML.Text = "Load KML File in Google Earth When Complete";
this.chkbxLoadKML.UseVisualStyleBackColor = true;
//
// lblListTransparencies
//
this.lblListTransparencies.AutoSize = true;
this.lblListTransparencies.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.lblListTransparencies.Location = new System.Drawing.Point(479, 143);
this.lblListTransparencies.Name = "lblListTransparencies";
this.lblListTransparencies.Size = new System.Drawing.Size(175, 13);
this.lblListTransparencies.TabIndex = 31;
this.lblListTransparencies.Text = "Current List of Color Transparencies";
this.lblListTransparencies.Click += new System.EventHandler(this.label1_Click);
//
// grpbxImageOptions
//
this.grpbxImageOptions.Controls.Add(this.btnAboutTransparencies);
this.grpbxImageOptions.Controls.Add(this.btnRemoveColor);
this.grpbxImageOptions.Location = new System.Drawing.Point(27, 5);
this.grpbxImageOptions.Name = "grpbxImageOptions";
this.grpbxImageOptions.Size = new System.Drawing.Size(803, 262);
this.grpbxImageOptions.TabIndex = 32;
this.grpbxImageOptions.TabStop = false;
this.grpbxImageOptions.Text = "Image Pyramid Options";
this.grpbxImageOptions.ForeColorChanged += new
System.EventHandler(this.chkUsePng_CheckedChanged);
this.grpbxImageOptions.Enter += new System.EventHandler(this.groupBox1_Enter);
//
// btnAboutTransparencies
//
this.btnAboutTransparencies.Location = new System.Drawing.Point(226, 212);
this.btnAboutTransparencies.Name = "btnAboutTransparencies";
this.btnAboutTransparencies.Size = new System.Drawing.Size(195, 23);
this.btnAboutTransparencies.TabIndex = 35;
this.btnAboutTransparencies.Text = "How Do I Use Transparencies?";
this.btnAboutTransparencies.UseVisualStyleBackColor = true;

```

```

        this.btnAboutTransparencies.Click += new
System.EventHandler(this.btnAboutTransparencies_Click);
//
// btnRemoveColor
//
this.btnRemoveColor.Enabled = false;
this.btnRemoveColor.ForeColor = System.Drawing.SystemColors.ButtonShadow;
this.btnRemoveColor.Location = new System.Drawing.Point(226, 183);
this.btnRemoveColor.Name = "btnRemoveColor";
this.btnRemoveColor.Size = new System.Drawing.Size(195, 23);
this.btnRemoveColor.TabIndex = 34;
this.btnRemoveColor.Text = "Removed Hilighted Color from List";
this.btnRemoveColor.UseVisualStyleBackColor = true;
this.btnRemoveColor.Click += new System.EventHandler(this.button1_Click);
//
// grpboxCoordinates
//
this.grpboxCoordinates.Controls.Add(this.txtGeoFile);
this.grpboxCoordinates.Controls.Add(this.btnFindGeo);
this.grpboxCoordinates.Controls.Add(this.lblBrowseGeo);
this.grpboxCoordinates.Controls.Add(this.rdoBtnEnterManually);
this.grpboxCoordinates.Controls.Add(this.rdoBtnGetFromGeo);
this.grpboxCoordinates.Controls.Add(this.btnAboutGeo);
this.grpboxCoordinates.Controls.Add(this.btnAboutLatLon);
this.grpboxCoordinates.Controls.Add(this.lblPromptCoordinates);
this.grpboxCoordinates.Location = new System.Drawing.Point(27, 282);
this.grpboxCoordinates.Name = "grpboxCoordinates";
this.grpboxCoordinates.Size = new System.Drawing.Size(803, 246);
this.grpboxCoordinates.TabIndex = 33;
this.grpboxCoordinates.TabStop = false;
this.grpboxCoordinates.Text = "Coordinate Input Options";
//
// txtGeoFile
//
this.txtGeoFile.ForeColor = System.Drawing.SystemColors.InfoText;
this.txtGeoFile.Location = new System.Drawing.Point(172, 113);
this.txtGeoFile.Name = "txtGeoFile";
this.txtGeoFile.Size = new System.Drawing.Size(461, 20);
this.txtGeoFile.TabIndex = 42;
this.txtGeoFile.Text = "Select .geo File";
this.txtGeoFile.TextChanged += new System.EventHandler(this.txtGeoFile_TextChanged);
//
// btnFindGeo
//
this.btnFindGeo.ForeColor = System.Drawing.SystemColors.ControlText;
this.btnFindGeo.Location = new System.Drawing.Point(676, 110);
this.btnFindGeo.Name = "btnFindGeo";
this.btnFindGeo.Size = new System.Drawing.Size(107, 26);
this.btnFindGeo.TabIndex = 41;
this.btnFindGeo.Text = "Browse Files";
this.btnFindGeo.UseVisualStyleBackColor = true;
this.btnFindGeo.Click += new System.EventHandler(this.btnFindGeo_Click);
//
// lblBrowseGeo
//
this.lblBrowseGeo.AutoSize = true;

```



```

this.lblBrowseGeo.ForeColor = System.Drawing.SystemColors.ControlText;
this.lblBrowseGeo.Location = new System.Drawing.Point(20, 116);
this.lblBrowseGeo.Name = "lblBrowseGeo";
this.lblBrowseGeo.Size = new System.Drawing.Size(80, 13);
this.lblBrowseGeo.TabIndex = 40;
this.lblBrowseGeo.Text = "Select .geo File";
//
// rdoBtnEnterManually
//
this.rdoBtnEnterManually.AutoSize = true;
this.rdoBtnEnterManually.Location = new System.Drawing.Point(31, 164);
this.rdoBtnEnterManually.Name = "rdoBtnEnterManually";
this.rdoBtnEnterManually.Size = new System.Drawing.Size(154, 17);
this.rdoBtnEnterManually.TabIndex = 39;
this.rdoBtnEnterManually.Text = "Enter Coordinates Manually";
this.rdoBtnEnterManually.UseVisualStyleBackColor = true;
this.rdoBtnEnterManually.CheckedChanged += new
System.EventHandler(this.rdoBtnEnterManually_CheckedChanged);
//
// rdoBtnGetFromGeo
//
this.rdoBtnGetFromGeo.AutoSize = true;
this.rdoBtnGetFromGeo.Checked = true;
this.rdoBtnGetFromGeo.Location = new System.Drawing.Point(31, 61);
this.rdoBtnGetFromGeo.Name = "rdoBtnGetFromGeo";
this.rdoBtnGetFromGeo.Size = new System.Drawing.Size(167, 17);
this.rdoBtnGetFromGeo.TabIndex = 38;
this.rdoBtnGetFromGeo.TabStop = true;
this.rdoBtnGetFromGeo.Text = "Get Coordinates from .geo File";
this.rdoBtnGetFromGeo.UseVisualStyleBackColor = true;
this.rdoBtnGetFromGeo.CheckedChanged += new
System.EventHandler(this.radioButton1_CheckedChanged);
//
// btnAboutGeo
//
this.btnAboutGeo.Location = new System.Drawing.Point(226, 57);
this.btnAboutGeo.Name = "btnAboutGeo";
this.btnAboutGeo.Size = new System.Drawing.Size(173, 24);
this.btnAboutGeo.TabIndex = 37;
this.btnAboutGeo.Text = "What is a .geo file?";
this.btnAboutGeo.UseVisualStyleBackColor = true;
this.btnAboutGeo.Click += new System.EventHandler(this.btnAboutGeo_Click);
//
// btnAboutLatLon
//
this.btnAboutLatLon.Location = new System.Drawing.Point(226, 160);
this.btnAboutLatLon.Name = "btnAboutLatLon";
this.btnAboutLatLon.Size = new System.Drawing.Size(233, 24);
this.btnAboutLatLon.TabIndex = 36;
this.btnAboutLatLon.Text = "How Do I Enter Latitude and Longitude?";
this.btnAboutLatLon.UseVisualStyleBackColor = true;
this.btnAboutLatLon.Click += new System.EventHandler(this.btnAboutLatLon_Click);
//
// grpBxKMLOptions
//
this.grpBxKMLOptions.Controls.Add(this.btnKMLFileDestination);

```

```

this.grpbxKMLOptions.Controls.Add(this.txtKMLFileDestination);
this.grpbxKMLOptions.Location = new System.Drawing.Point(27, 534);
this.grpbxKMLOptions.Name = "grpbxKMLOptions";
this.grpbxKMLOptions.Size = new System.Drawing.Size(803, 137);
this.grpbxKMLOptions.TabIndex = 10;
this.grpbxKMLOptions.TabStop = false;
this.grpbxKMLOptions.Text = "KML File Options";
this.grpbxKMLOptions.Enter += new System.EventHandler(this.grpbxKMLOptions_Enter);
//
// brwsrImagePyramidDestination
//
this.brwsrImagePyramidDestination.HelpRequest += new
System.EventHandler(this.brwsrImagePyramidDestination_HelpRequest);
//
// oftSelectGeo
//
this.oftSelectGeo.FileName = "openFileDialog1";
this.oftSelectGeo.Filter = "\"Geo Files (*.geo)|*.geo|Text files (*.txt)|*.txt|All files (*.*)|*.*\"";
//
// frmSuperOverlayCreator
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(842, 731);
this.Controls.Add(this.lblListTransperencies);
this.Controls.Add(this.chkbxLoadKML);
this.Controls.Add(this.txtImagePyramidDestination);
this.Controls.Add(this.txtImageFile);
this.Controls.Add(this.btnClearAll);
this.Controls.Add(this.btnCreateKML);
this.Controls.Add(this.lblSaveKML);
this.Controls.Add(this.btnSelectDestination);
this.Controls.Add(this.lblSelectImagePyramidDestination);
this.Controls.Add(this.lstbxTransparency);
this.Controls.Add(this.btnAddColors);
this.Controls.Add(this.lblTransparency);
this.Controls.Add(this.btnWhyPng);
this.Controls.Add(this.chkUsePng);
this.Controls.Add(this.txtWest);
this.Controls.Add(this.txtEast);
this.Controls.Add(this.txtSouth);
this.Controls.Add(this.txtNorth);
this.Controls.Add(this.lblWest);
this.Controls.Add(this.lblEast);
this.Controls.Add(this.lblSouth);
this.Controls.Add(this.lblNorth);
this.Controls.Add(this.btnSelectImage);
this.Controls.Add(this.lblPromtImage);
this.Controls.Add(this.grpbxImageOptions);
this.Controls.Add(this.grpbxCoordinates);
this.Controls.Add(this.grpbxKMLOptions);
this.Name = "frmSuperOverlayCreator";
this.Text = "KML Super Overlay Creator";
this.Load += new System.EventHandler(this.Form1_Load);
this.grpbxImageOptions.ResumeLayout(false);
this.grpbxCoordinates.ResumeLayout(false);

```

```

        this.grpbxCoordinates.PerformLayout();
        this.grpbxKMLOptions.ResumeLayout(false);
        this.grpbxKMLOptions.PerformLayout();
        this.ResumeLayout(false);
        this.PerformLayout();
    }

#endregion

private System.Windows.Forms.Label lblPromtImage;
private System.Windows.Forms.Button btnSelectImage;
private System.Windows.Forms.OpenFileDialog ofdSelectImage;
private System.Windows.Forms.Label lblPromptCoordinates;
private System.Windows.Forms.Label lblNorth;
private System.Windows.Forms.Label lblSouth;
private System.Windows.Forms.Label lblEast;
private System.Windows.Forms.Label lblWest;
private System.Windows.Forms.TextBox txtNorth;
private System.Windows.Forms.TextBox txtSouth;
private System.Windows.Forms.TextBox txtEast;
private System.Windows.Forms.TextBox txtWest;
private System.Windows.Forms.CheckBox chkUsePng;
private System.Windows.Forms.Button btnWhyPng;
private System.Windows.Forms.Label lblTransparency;
private System.Windows.Forms.ColorDialog csdTransparencyColor;
private System.Windows.Forms.Button btnAddColors;
private System.Windows.Forms.ListBox lstbxTransparency;
private System.Windows.Forms.Label lblSelectImagePyramidDestination;
private System.Windows.Forms.Button btnSelectDestination;
private System.Windows.Forms.Label lblSaveKML;
private System.Windows.Forms.Button btnKMLFileDestination;
private System.Windows.Forms.Button btnCreateKML;
private System.Windows.Forms.Button btnClearAll;
private System.Windows.Forms.TextBox txtImageFile;
private System.Windows.Forms.TextBox txtImagePyramidDestination;
private System.Windows.Forms.TextBox txtKMLFileDestination;
private System.Windows.Forms.CheckBox chkbxLoadKML;
private System.Windows.Forms.Label lblListTransperencies;
private System.Windows.Forms.GroupBox grpbxImageOptions;
private System.Windows.Forms.GroupBox grpbxCoordinates;
private System.Windows.Forms.GroupBox grpbxKMLOptions;
private System.Windows.Forms.FolderBrowserDialog brwsrImagePyramidDestination;
private System.Windows.Forms.FolderBrowserDialog brwsrKMLDestination;
private System.Windows.Forms.Button btnRemoveColor;
private System.Windows.Forms.Button btnAboutTransparencies;
private System.Windows.Forms.Button btnAboutLatLon;
private System.Windows.Forms.Button btnAboutGeo;
private System.Windows.Forms.RadioButton rdobtnGetFromGeo;
private System.Windows.Forms.RadioButton rdobtnEnterManually;
private System.Windows.Forms.TextBox txtGeoFile;
private System.Windows.Forms.Button btnFindGeo;
private System.Windows.Forms.Label lblBrowseGeo;
private System.Windows.Forms.OpenFileDialog offtSelectGeo;
    }
}

```