Georeferencing Unmanned Aerial Systems Imagery via Registration with Geobrowser Reference Imagery

THESIS

Presented in Partial Fulfillment of the Requirements for the Degree Master of Science in the Graduate School of The Ohio State University

By

Robert Nevins

Graduate Program in Civil Engineering

The Ohio State University

2017

Master's Examination Committee:

Professor Dorota Grejner-Brzezinska, Advisor
Professor Charles Toth
Assistant Professor Rongjun Qin
Abstract

Unmanned aerial systems are developing into increasingly competitive platforms for aerial image surveying in a variety of applications. Easy-to-use and relatively inexpensive, their utility is however reliant on georeferencing their imagery with respect to an earth-based coordinate system. Traditionally this requires the time-consuming and potentially cost-increasing introduction of ground control points into the scene of interest. Other techniques, such as direct georeferencing using integrated Global Navigation Satellite System receivers and inertial navigations systems, struggle to achieve comparable accuracy due to weight and cost limitations faced with highest-accuracy instrumentation.

In this work, unmanned aerial system imagery was georeferenced via registration with satellite imagery downloaded from online geobrowser image databases, specifically Environmental Systems Research Institute World Imagery. This method allows the potential elimination of all fieldwork related to ground control point distribution and surveying, taking advantage instead of instantaneous access daytime, cloud-free satellite imagery provided by geobrowsers. Registration was performed both using pixel-based template matching, and successive application of pixel-based and feature-based keypoint detection and matching techniques. Test imagery was collected by unmanned aerial systems over a parking lot and surrounding area in central Ohio. Root mean square error results were calculated for both pixel-based and successive pixel-based and feature-based
registration with respect to 8 ground control points measured independently by Global Navigation Satellite System survey. Using 0.3 meter reference imagery, sub-pixel accuracy was achieved with successive pixel-based and feature-based registration. Possible applications include unmanned aerial systems mapping for endeavors such as precision agriculture, and unmanned aerial systems navigation.
To my grandparents
Acknowledgments

I would like to thank my advisor, Dr. Dorota Grejner-Brzezinska, and Dr. Charles Toth for their academic and professional support and guidance throughout my study. I would also like to thank Dr. Rongjun Qin, Dr. Alper Yilmaz, and the other faculty members who have helped facilitate my thesis work. In addition, I would like to thank The Ohio State University and the Department of Civil, Environmental, and Geodetic Engineering for the financial support to pursue my degree.

Finally, thank you to my family, friends, and colleagues at the Satellite Positioning and Inertial Navigation Laboratory for their continuous support and friendship.
Vita

June 2011 .......................................................Leesville Road High School

May 2015 .......................................................B.A. Mathematics, Physics, Ancient Studies,
St. Olaf College

2015 to 2016 ..................................................University Fellow, The Ohio State University

2017 to present ..............................................Graduate Associate, Department of Civil,
Environmental, and Geodetic Engineering,
The Ohio State University

Publications


Fields of Study

Major Field: Civil Engineering
# Table of Contents

Abstract ........................................................................................................................................... ii

Acknowledgments ......................................................................................................................... v

Vita ............................................................................................................................................ vi

List of Tables ................................................................................................................................. x

List of Figures ................................................................................................................................. xi

List of Acronyms ............................................................................................................................. xiii

Chapter 1: Introduction .................................................................................................................. 1

1.1 Overview .................................................................................................................................. 1

1.2 Related Work .......................................................................................................................... 2

1.3 Contributions .......................................................................................................................... 7

Chapter 2: Imagery and Image Preprocessing ............................................................................. 11

2.1 Experiment Description .......................................................................................................... 11

2.2 Orthomosaicking with Pix4D ............................................................................................... 11

2.3 Esri Reference Imagery .......................................................................................................... 13

2.3.1 Default Product ................................................................................................................ 14

2.3.2 Level 2b Image Generation .............................................................................................. 14
List of Tables

Table 1. RMSE results for first-order coarse registration, and first- and second-order fine registration. ................................................................. 50
List of Figures

Figure 1. A flowchart depicting the registration algorithm ............................................... 8
Figure 2. The UAS orthomosaic calculated using Pix4D. The GCP locations are indicated
by red circles. .................................................................................................................... 13
Figure 3. Top: reference edge image without filtering preprocessing. Bottom: reference
dge image, with bilateral filtering prior to edge detection ............................................. 19
Figure 4. From left to right: reference edge image parking lot lines pre-closing, post-
closing, and orthomosaic parking lot lines. ................................................................. 24
Figure 5. flowchart depicting the coarse registration modified template matching
algorithm .......................................................................................................................... 28
Figure 6. Unrotated image on the left, compared to the rotated image on the right, with
no-data pixels introduced along both axes to preserve the rectangular array shape. The red
box overlay indicates the dimensions of the left tile ......................................................... 37
Figure 7. Error plot of GCPs following coarse registration and first-order georeferencing.
Error bars are magnified by a factor of 5. ....................................................................... 46
Figure 8. Error plot of GCPs following fine registration and first-order georeferencing.
Error bars are magnified by a factor of 5. ....................................................................... 47
Figure 9. Error plot of GCPs following fine registration and second-order georeferencing.
Error bars are magnified by a factor of 5. ....................................................................... 48
Figure 10. GCP locations as calculated from GPS survey, coarse registration, and first- and second-order fine registration. ................................................................. 49
List of Acronyms

- ASIFT: Affine-Scale-Invariant Feature Transform
- DEM: Digital Elevation Model
- DoG: Difference-of-Gaussians
- Esri: Environmental Systems Research Institute
- EXIF: Exchangeable Image File Format
- FAIR-SURF: Fully Affine Invariant Speeded Up Robust Features
- FLANN: Fast Approximate Nearest Neighbor Search Library
- GCP: Ground Control Point
- GDAL: Geospatial Data Abstraction Library
- GeoTIFF: Geographic Tagged Image File Format
- GPS: Global Positioning System
- INS: Inertial Navigation System
- LoG: Laplacian of Gaussian
- RMSE: Root Mean Square Error
- ROHM: Robust-Oriented Hausdorff Measure
- RPC: Rational Polynomial Coefficient
- SGM: Semi-Global Matching
- SIFT: Scale-Invariant Feature Transform
- SURF: Speeded Up Robust Features
- UAS: Unmanned Aerial Systems
- VRS: Virtual Reference Station
Chapter 1: Introduction

1.1 Overview

Unmanned Aerial Systems (UAS) have been demonstrated to be competitive remote sensing platforms for a variety of applications, including agricultural, viticultural, and various civil engineering applications [1,2,3]. The low operating cost, high spatial resolution, and relative ease of use achievable by UAS makes image data collection a relatively simple task when compared to the challenges faced in acquiring aircraft or satellite image data. This utility of UAS imagery is however reliant upon georeferencing the imagery in an earth-based coordinate system. Traditionally this requires the manual identification of control points between images, using either natural landmarks or via the introduction of ground control points (GCPs) into the area of interest. These methods can result in time-consuming in situ and/or post-processing work.

Alternatively, another frequently utilized method for georeferencing aerial imagery is direct georeferencing [4, 5]. By integrating Global Positioning System (GPS) and Inertial Navigation System (INS) measurements, the full exterior orientation of an imaging system can be measured, and each pixel of each image can be georeferenced while reducing or eliminating the need for GCPs. While direct georeferencing accuracy can be high, its practicality is limited for use with UAS, as the sensor weight and cost can prevent the use of highest-accuracy integrated GPS/INS systems. With the inclusion of
One possible solution to the aforementioned UAS georeferencing issues is to register UAS imagery with georeferenced imagery obtained from online earth image databases, or geobrowsers, such as Environmental Systems Research Institute (Esri) World Imagery or Google Earth. While other satellite imagery could be used, databases such as these provide instantaneous access to cloud-free, daytime, georeferenced imagery, which compliments the virtually at-a-whim image collection capability of UAS.

1.2 Related Work

Two main categories of image registration algorithms exist, which include area-, or pixel-based, and feature-based methods [8]. In pixel-based registration algorithms, images are compared on a pixel-by-pixel basis using a correlation metric to find the optimal correspondence. Typically, the entirety of the images, or subsets of the images, will be registered in pixel-based algorithms. Feature-based methods, on the other hand, select a number of keypoints, or features, in the images. A matching algorithm is then used to obtain keypoint pairs between images. Based on these matches, a geometrical transformation can then be applied to project one image into the coordinate system of the other.

Both pixel-based and feature-based registration techniques have been used for the purpose of georeferencing UAS imagery. Pixel-based registration of UAS images with reference imagery has been successfully implemented as a component of a UAS navigation filter in [9,10]. Other pixel-based registration algorithms have been implemented for aircraft navigation as well, such as robust-oriented Hausdorff measure.
(ROHM) matching in [11]. One benefit of pixel-based registration methods is that they can be more robust than feature-based registration—since the UAS imagery and reference image to which it is being registered are often taken at different times, using the entirety of the image can reduce the influence of features (like cars or pedestrians) that may be present in one image but not the other, or that may have changed location.

One important issue that must be addressed when registering UAS imagery with georeferenced imagery from other sources (amongst other applications) is that of image scale. While a pixel in an image taken from a UAS may have a ground sample distance on the order of centimeters, reference imagery from an aircraft or satellite is often of decimeter or meter-level resolution. For pixel-based registration methods, scale discrepancies are often addressed by resampling the UAS imagery at a number of different scales, and then assessing the location and scale that provides the best match with the reference image.

Feature-based registration techniques also have various ways of dealing with scale discrepancies in their keypoint detection and matching algorithms. One of the most ubiquitous methods is scale-invariant feature transform (SIFT), which has been shown to be one of the best performing keypoint descriptors compared to various other techniques [12,13]. As its name would suggest, SIFT is very robust in its ability to reproduce keypoints in images of varying scale. SIFT functions by scale-space filtering, using a technique known as Difference-of-Gaussians (DoG) [14]. DoG is a computationally faster approximation of using Laplacian of Gaussian (LoG) operators, which are used to detect regions of contrasting intensity, or blobs, at varying simulated image scales. This
is accomplished by smoothing the image by means of convolution with a Gaussian kernel of varying widths $\sigma$.

The DoG algorithm instead is calculated by smoothing an image with Gaussian kernels of two different $\sigma$ values, and taking their difference across a variety of different Gaussian pyramid octaves. Keypoints can then be detected by means of an extrema search over space and scale, enabling reproducible keypoints to be detected in images regardless of whether their scales agree or not. Keypoint matching in the SIFT algorithm is based on a 128-dimensional vector that is computed as a descriptor of the 16x16 pixel neighborhood of each keypoint. Matching can then be performed to find the keypoints between images whose descriptors are most similar.

SIFT has been widely implemented in UAS image registration algorithms, both as presented in [14] and with various other performance improvements. Improvement efforts have focused on the one hand on improving performance speed through various means [15, 16]. This has also led to the development of other feature detection algorithms, such as the popular Speeded Up Robust Features algorithm (SURF) [17]. On the other hand, efforts have been made to improve the accuracy of not only the matches calculated by these algorithms, but of the resulting geometric transformation as well.

One of the primary methods of ensuring that keypoint matches correspond to a reasonable geometric transformation is Random Sample Consensus [18]. In this algorithm, a random subset of the keypoint matches are selected as “hypothetical inliers,” from which a transformation model is computed. Then, the rest of the keypoint matches are tested against this model, with the matches that are well described by the model being
added to the “consensus set.” This set may then be used to re-estimate the transformation model, and recompute the consensus set.

This algorithm may be applied iteratively, beginning with a random subset each time, until a transformation model is accepted based on the size of its consensus set. RANSAC has been applied in conjunction with different feature-based registration algorithms, with the potential to significantly increase registration accuracy due to its ability to reduce the overall number of false matches and outliers [19,20].

Despite the robustness provided by algorithms such as SIFT to changes in image viewpoint, such as rotation and scale, there are some instances that arise where matching keypoints may not be detected. Larger changes in camera orientation, which may occur during a UAS flight, can prevent persistent keypoints across images from being detected. To make keypoint detection more robust to large changes in camera orientation, algorithms such as fully affine invariant SURF (FAIR-SURF) and affine-SIFT (ASIFT) have been developed [21, 22]. By simulating a variety of latitude and longitude camera orientation angles, in addition to the rotation, scaling, and translation parameters accounted for in the SIFT algorithm, the resulting features can be reliably detected in images with much larger changes in camera orientation than could be managed using other algorithms. While the ASIFT algorithm is more computationally complex, it has been successfully applied to problems of UAS image stitching and georeferencing [23,24].

One important application of the aforementioned image registration techniques in the field of UAS image processing is for image mosaicking and point cloud generation [25]. Provided that a sufficient number of images from a UAS flight are available with
sufficient overlap, registration techniques can be implemented to detect matching points in multiple images. Using a large number of matching points, models can then be created for the camera optical parameters, as well as locations and orientations during image capture. Based on these models, as well as ground control and/or GPS/INS information as available, the images can then be stitched together in regions of overlap to form a single, larger image, or mosaic. Orthorectification of the mosaic is often subsequently performed as well, which removes the effects of image perspective and terrain variations to produce a geometrically-correct, uniform-scale mosaic, or orthomosaic. In addition to the orthomosaic, another key output from the UAS imagery workflow is the point cloud, which is a 3D vector representation of the UAS data, and can approach pixel-level density [26].

A variety of registration techniques can and have been used for mosaicking and point cloud generation, including cross correlation, least squares matching, and others [27]. One particularly effective and widely-used technique is Semi-Global Matching (SGM) [28]. SGM functions by pixelwise matching of mutual information, combined with 2D image approximation approach based on multiple 1D smoothness constraints crossing each pixel [29]. This is referred to as “energy” minimization and includes three terms, the first of which accounts for disparity image pixel matching costs, the second of which accounts for small disparity changes to accommodate curved or slanted surfaces, and the third of which accounts for discontinuities. SGM is a favored approach due to both its accuracy and speed—its accuracy exceeds that of local methods and approaches that of global methods, but does so considerably faster than global methods [28,29]. This has led to widespread implementation of SGM in a variety of photogrammetric image...
processing software, including SimActive, Hexagon, and Pix4D, which was used during the course of this research [30].

1.3 Contributions

This thesis implements a two-step algorithm for automatically georeferencing UAS imagery by means of registration with the Esri World Imagery database. First, a coarse registration was performed using a pixel-based matching algorithm between edge images generated from the Esri reference imagery and the UAS imagery. Then, a fine registration was performed using a feature-based matching algorithm based on keypoints generated from the Good Features to Track algorithm [31]. Results of the georeferencing accuracy are calculated both after implementation of the coarse registration alone, and after sequential implementation of the coarse and fine registration algorithms. These results allow for an accuracy comparison of a pixel-based registration algorithm with a composite pixel-based and feature-based registration algorithm, while also evaluating the accuracy potential of image registration and georeferencing between UAS and geobrowser imagery. An overview of the registration process detailed in this thesis is shown in figure 1, with a brief step-by-step explanation following immediately thereafter.
Figure 1. A flowchart depicting the registration algorithm.
a) In the first step, unmanned aerial system imagery is mosaicked and orthorectified using the software Pix4D. See section 2.2 for more details.

b) Reference imagery is downloaded from the Environmental Systems Research Institute World Imagery layer for the registration process. See section 2.3 for more details.

c) Unmanned aerial system orthomosaic and reference imagery are smoothed using a bilateral filter. See section 3.1 for more details.

d) Orthomosaic edge image and reference edge image are generated using Canny edge detector. See section 3.2 for more details.

e) A morphological closing is applied to the reference edge image. See section 3.3 for more details.

f) Template matching is performed to find the optimal scaling, rotation, and translation values to register the orthomosaic edge image with the reference edge image. See section 3.4 for more details.

g) Grayscale orthomosaic is transformed according to the scaling, rotation, and translation values calculated in f). See chapter 4 for more details.

h) Grayscale reference image is subset to the extent of the transformed Grayscale orthomosaic from g). See chapter 4 for more details.

i) Keypoints are detected in the reference image and orthomosaic using the Good Features to Track algorithm. See section 4.1 for more details.

j) Scale-Invariant Feature Transform descriptors are calculated for each keypoint detected in i). See section 4.2 for more details.
k) Orthomosaic keypoint descriptors are matched to reference image keypoint descriptors using a brute-force algorithm. See section 4.4 for more details.

l) Object-space geographic coordinates are calculated for each orthomosaic keypoint using the spatial location information from the reference image metadata. See section 5.3 for more details.

m) A geometric transformation is applied to the orthomosaic using the ArcGIS software, based on the known image-space coordinates and object-space coordinates calculated in l) for each tie point. See section 5.4 for more details.

n) Root mean square error is calculated from the ground control point location as projected from m) and as measured by Global Positioning System survey. See chapter 6 for more details.
Chapter 2: Imagery and Image Preprocessing

Two main sources of imagery were used for this thesis. To perform the necessary registration experiments, UAS imagery of a scene of interest was obtained, as well as a georeferenced satellite reference image of the same scene and surrounding area with which to register the UAS imagery.

2.1 Experiment Description

UAS imagery for this thesis was obtained from a Nikon D800 camera, flown on an octocopter over a parking lot and the surrounding area in central Ohio [26]. 248 images were captured at 4912 (cols) by 7360 (rows) pixel resolution, processed in 8-bit 3-channel resolution. A total of 8 GCPs were introduced to the scene, and their locations, as well as the locations of several natural landmarks, were independently measured by GPS survey. These GCPs permitted the calculation of an error metric after the registration algorithms were performed, to calculate the geolocation error of the resulting georeferenced UAS imagery.

2.2 Orthomosaicking with Pix4D

Rather than attempt to register each individual image taken by the Nikon D800, the images were first mosaicked and orthorectified into a single larger orthomosaic of the entire scene of interest. This is a much more robust approach than image-by-image registration between UAS imagery and reference imagery, since it allows all features in
the geometrically-corrected orthomosaic to be simultaneously matched with reference imagery, rather than attempt to individually register each UAS image to the reference image and attempt mosaicking afterwards.

To mosaic the UAS images, the software suite Pix4D was used [32]. Pix4D contains a variety of computer vision and photogrammetry algorithms, with the ability to create 3D maps and models from overlapping input imagery. The basic Pix4D algorithm for generating composite maps can be broken down into 5 steps [32,33]. In the first step, a feature-based matching algorithm is used to find keypoint matches between images. Next, a bundle block adjustment is performed using the keypoint matches, as well as navigation and orientation data from the UAS. In the third step, 3D coordinates for the keypoints are calculated in the WGS 1984 reference system from UAS GPS data, if it is available [34]. Otherwise, a local mapping frame may be used. A digital elevation model (DEM) is then generated based on a triangulated irregular network based on interpolation of the keypoint 3D coordinates. Finally, in the fifth step every image pixel is projected and the georeferenced orthomosaic is generated using the DEM.

Although image mosaicking can be performed in Pix4D without any additional georeferencing data, there are a variety of ways that the accuracy can be improved. GPS navigation data and image orientation data, obtained from GPS receiver and INS instrumentation aboard the UAS, as well as independently measured GCPs introduced to the scene can all be included to dramatically improve accuracy. For this thesis, since orientation data was not included by default in the Exchangeable Image File Format (EXIF) metadata recorded at the time of image capture, and since the objective was to register the orthomosaic without GCPs, only GPS location data was included. GPS data
was recorded at the time of each image capture using Single Point Positioning, with 3D accuracy on the order of 10 meters. This was sufficient to utilize the “standard” accuracy setting in Pix4D, defined as horizontal accuracy of 5 meters and vertical accuracy of 10 meters; rather than “low,” accuracy setting, defined as horizontal accuracy of 50 meters and vertical accuracy of 100 meters [32]. This setting results in a much more accurate orthomosaic, with meter-level root mean square error (RMSE) before introduction of ground control or registration. The resulting orthomosaic, as well as the location of the 8 GCPs, are shown in figure 2.

Figure 2. The UAS orthomosaic calculated using Pix4D. The GCP locations are indicated by red circles.

2.3 Esri Reference Imagery

The other image product for the registration process, the reference image, was obtained from the Esri World Imagery database. This product contains imagery for the world, as well as high-resolution imagery in certain areas of the world and the United
States, with resolution as fine as 0.03 meters. A variety of imagery sources and satellites are included to populate the World Imagery layer, which is updated regularly. Nevertheless, depending on the location and resolution, image collection dates can vary dramatically, with some images being out of date by many years.

2.3.1 Default Product

The highest resolution imagery available in the Esri World Imagery layer corresponding to the location of the UAS imagery was 0.3 meter imagery obtained from WorldView-3, collected on February 27, 2012. WorldView-3 is a commercial imaging satellite, owned by DigitalGlobe, which includes the ability to provide panchromatic and pansharpened multispectral imagery at 31 centimeter resolution, with default multispectral imagery collected at 1.24 meter resolution [35]. The imagery is near-orthographic, due to its vertical collection from orbit, but documentation for this scene does not indicate further processing was applied to achieve a true orthographic projection. Imagery corresponding to the UAS flight zone was downloaded as a 577 x 258 pixel image in Geographic Tagged Image File Format (GeoTIFF), using nearest-neighbor interpolation with 12 centimeter pixel size [36]. In this format, the geospatial information corresponding to the Esri reference imagery was saved as well. This includes details such as coordinate systems, projections, datums, reference ellipsoids, and other details that allow the image to be projected in a given coordinate system. During the registration/georeferencing algorithm, this information was used to assign geospatial location data to the UAS orthomosaic, after the registration processes were performed.

2.3.2 Level 2b Image Generation
According to the Esri metadata, the 2D geolocation accuracy of the 0.3 meter resolution layer that the reference image was taken from was 2.7 meters. While locally the geolocation error was less than this, there was still over a meter of error between natural landmarks in the reference image and their GPS-measured locations. To support the investigation, a zero-order polynomial shift, or simply a 2D translation, was applied to better align the natural landmarks with their GPS-measured locations, as a means of generating “level 2b” reference imagery. Level 2a and 2b are labels commonly applied to various different stages of the satellite imagery correction and georeferencing process [37,38].

Level 2a imagery is directly georeferenced, in that it has been projected into an earth-based coordinate system based on estimations of where the satellite was located and its attitude when the image was acquired. This is considered an entry-level map product, and is representative of the state of the reference image as it was originally downloaded from Esri. Level 2b imagery is an improvement of level 2a imagery via the introduction of GCPs to further improve the accuracy of the level 2a projection. The geolocation accuracy of level 2b imagery is typically equal to or better than the spatial resolution of the image.

Other methods of improving the geolocation accuracy of level 2a imagery are commonly used, such as rigorous sensor models or Rational Polynomial Coefficients (RPCs), which can achieve subpixel-level accuracy [39]. RPCs are favored in some cases, because they provide relationships between image-space and object-space coordinates, without the need to disclose what may be confidential rigorous sensor models to the user of the imagery. Large numbers of GCPs are also frequently used,
alongside higher-order geometric transformations, to further improve geolocation accuracy. However, since default multispectral bands of WorldView-3 imagery are at 1.24 meter resolution, over a meter of geolocation error can persist even after processing with RPCs [40]. Moreover, with the significant time span between the Esri image date and UAS imagery and GCP collection date, visible physical scene changes were noticeable affecting the surveyed natural landmarks. For example, parking lot line corner locations had been measured by GPS survey as natural landmarks on the day of the UAS flight, but the lot had visibly been repainted in a different layout between the time of reference image collection and UAS flight date. Thus, a shift with a single, robust natural landmark GCP was applied to the Esri reference image, dramatically increasing the geolocation accuracy of the reference image, without the need to include geolocation information from other landmarks that were suspected to have changed over time.
Chapter 3: Coarse Registration

To register the reference image and orthomosaic, OpenCV version 3.2.0 was used. OpenCV is a free, open-source computer vision software library by Intel [41]. Due to its robust collection of computer vision and machine learning algorithms and relative computational efficiency, OpenCV is a competitive peer to many commercial computer vision packages available today. Additionally, OpenCV is available on a variety of interfaces, including C++, Java, Python, and Matlab. For this thesis, OpenCV was run using Python 3.6.

Since several algorithms in the OpenCV library, such as Canny edge detection and Good Features to Track require single-channel input images, the reference image and orthomosaic were imported to OpenCV in grayscale [42,31]. While the reference image is easy to manipulate due to its raster size of 577x258 pixels and 720 kilobytes, the orthomosaic at full resolution of 40925x18307 pixels and 1.13 gigabytes was too large for OpenCV to manipulate. Accordingly, the orthomosaic was downsampled to a ground sample distance of approximately 15 centimeters, before being imported in OpenCV for the registration process.

3.1 Bilateral Filtering

To reduce the noise of the reference image and orthomosaic, a bilateral filter was applied to both images due to its ease of implementation, alongside its effectiveness at
preserving edge features while removing unwanted noise [43]. The principal behind bilateral filtering is to combine both domain and range filtering, that is to consider both the spatial and photometric information within a given neighborhood for each pixel. Bilateral filtering of an image \( I \) to a resulting filtered image \( h \) can be described according to the equation

\[
h(x) = k^{-1}(x) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(\xi)c(\xi,x)s(I(\xi),I(x)) \, d\xi
\]

with the normalization

\[
k(x) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} c(\xi,x)s(I(\xi),I(x)) \, d\xi.
\]

Here, \( c(\xi,x) \) is a measure of the geometric distance between the neighborhood center \( x \) and a nearby point \( \xi \). Similarly, \( s(I(\xi),I(x)) \) is a measure of the photometric similarity between the aforementioned points \( x \) and \( \xi \).

The benefit in combining both geometric distance and photometric similarity in the bilateral filter, particularly for the coarse registration algorithm, is the ability of this filter to preserve edges better than a filter relying only on geometric distance can. While a mean or Gaussian filter, for example, applies a given weight to each pixel in the filter’s kernel based on its location with respect to the center pixel, regardless of photometric variability, a bilateral filter can apply different levels of smoothing based on the neighborhood of each pixel as the kernel is passed over it. An area of low photometric variability, such as a grass field, thus receives more smoothing to maximize noise reduction, while an area of high photometric variability, such as a painted white line in an
asphalt parking lot, receives less smoothing to preserve the edges corresponding to the paint/asphalt transitions. Figure 3 demonstrates the resulting edge images with and without prior bilateral filtering of the original reference image, with filtering resulting in significant noise reduction in regions such as the northern and eastern grass fields in the image, and minimal loss of robust edges in the central parking lot and built-up regions.

Figure 3. Top: reference edge image without filtering preprocessing. Bottom: reference edge image, with bilateral filtering prior to edge detection.
3.2 Canny Edge Detection

After a bilateral filter was applied to both the reference image and orthomosaic, the edges of both images were detected using the Canny edge detector algorithm [42]. Often viewed as the “standard” algorithm for edge detection, the Canny algorithm has been utilized since it was first publicized in the 1980’s, and still maintains a high level of performance amidst more recently developed edge detection algorithms [44]. The Canny algorithm functions in four main steps, which include image smoothing, gradient calculation, non-maximal suppression, and hysteresis thresholding.

3.2.1 Image Smoothing

Before the actual edge detection procedure is performed, according to the original algorithm in [42], a two-dimensional Gaussian filter is applied to the image of interest. However, OpenCV’s implementation of the Canny algorithm does not include an integrated Gaussian smoothing, relying instead upon the user to apply any filtering beforehand. Since both the reference image and orthomosaic were previously smoothed using a bilateral filter, no additional smoothing was performed before implementing the rest of the Canny algorithm.

3.2.2 Gradient calculation

After the image of interest has been smoothed, the next step in the Canny algorithm is to take the gradient of the image. Since edges occur at regions of rapid intensity change in the image, taking the gradient of the image allows isolation of edge features to extrema in the gradient image. This is done by filtering the image with a 3x3 Sobel kernel in the horizontal and vertical directions [45]. The edge gradient is then computed as
\[ g = \sqrt{g_x^2 + g_y^2} \]  

where \( g \) is the net edge gradient, \( g_x \) is the gradient in the \( x \) direction and \( g_y \) is the gradient in the \( y \) direction. The angle \( \theta \) of the gradient is calculated as

\[ \theta = \tan^{-1}\left(\frac{g_y}{g_x}\right) \]

This angle is then rounded to one of four orientations. From vertical, these orientations correspond to: 0/180 degrees, +45/+225 degrees, -45/-225 degrees, and +90/-90 degrees.

3.2.3 Non-Maximal Suppression

Having calculated the gradient magnitude and direction, the next step in the Canny edge detection algorithm is non-maximal suppression, to ensure that edges aren’t unnecessarily “thick” due to pixels near the edge being classified as edge pixels themselves. This is accomplished by looking at a 3x3 neighborhood for every pixel in the image. The gradient magnitude of the central pixel is compared to that of the two nearest pixels on either side of the central pixel, in the direction parallel to the previously calculated central pixel orientation. If the gradient magnitude of the central pixel is greater than that of both these pixels, it is retained for the next step. If not, then it is suppressed as a non-edge pixel.

3.2.4 Hysteresis thresholding

The final step in the Canny algorithm is hysteresis thresholding. In this stage, a lower bound and an upper bound are used to determine which of the pixels retained by the non-maximal suppression are edge pixels and which are not. If a pixel has a gradient magnitude value greater than the upper bound, then it is classified as an edge pixel. Similarly, if its value is smaller than the lower bound, it is classified as a non-edge pixel.
If the pixel’s gradient value is between the lower and upper bound, then its classification depends on its neighbors—if the pixel is adjacent to an edge pixel, then it too will be classified as an edge pixel, but if it is not adjacent to an edge pixel, then it will be classified as a non-edge pixel.

Canny edge images were generated according to the aforementioned algorithm for both the reference image and the orthomosaic. The reference edge image was generated using the reference image at full spatial resolution, as obtained from Esri. The orthomosaic edge image, on the other hand, was generated using the higher resolution (twice that of the reference image) orthomosaic image. This was done to maximize the number of edge features detected by the Canny algorithm, which might not be picked up on a lower-resolution image. The resulting orthomosaic edge image was then resized to the same dimensions as the reference image using OpenCV’s resize function. The Inter_Area interpolation method was selected for the downsampling [46]. This method works by adjusting the interpolation window based on the biaxial scale factors used to downsample the image, and is documented as a preferred method for image decimation due to its resistance to aliasing effects. The net result is a reasonably-scaled orthomosaic image with more plentiful edge features.

3.3 Closing

The final step before applying the template matching algorithm was to apply a closing to the reference edge image. This was done primarily to consolidate individual features that were detected as double edges back into a single edge feature. For example, a white paint line in a parking lot would usually be represented as two edges in the Canny edge image, corresponding to the paint/asphalt intensity change on either side of the
painted line. See the left-most tile in figure 4 below for an example of this, which shows the edge image of two parking lot lines. However, the corresponding physical line in the orthomosaic would be represented as more or less a single, thicker edge due to the interpolation when its edge image was downsampled to near the resolution of the reference image. This is shown in the right-most tile in figure 4. Thus, a closing to “fill in” these parking lot lines and other features served two purposes for the subsequent template matching algorithm. First, matching the roughly single edge features in the orthomosaic to a “filled-in” edge feature in the reference image, rather than forcing the template matching algorithm to try to match the orthomosaic edge feature to an edge on either side of the actual feature in the reference image serves to increase the accuracy of the template matching globally. Second, by increasing the total number of nonzero pixels in each edge feature, the overall weight of the feature is increased in comparison to the zeroed “no data” pixels in the image. Since the objective is to achieve a good match between the nonzero edge features, with the zeroed non-edge pixels aligning primarily by consequence of this, this result ultimately enhances the template matching accuracy in the next stage.

An example is shown in figure 4 of parking lines in the reference edge image both before and after closing, as well as parking lines in the orthomosaic edge image. Note also that each individual parking lot line in the reference edge image is significantly thicker than the parking lot lines in the orthomosaic. Ordinarily, parking lot lines are about 4 inches, or just over 10 centimeters wide. While the UAS has no problem at this resolution, the 0.3 meter resolution of the Esri reference imagery means that parking lot lines are at best about three times wider in the reference edge image than in the
orthomosaic edge image. This corresponds to a width of a few pixels in the interpolated 12 centimeter ground sample distance reference image, as seen in the first two panels in figure 4.

Figure 4. From left to right: reference edge image parking lot lines pre-closing, post-closing, and orthomosaic parking lot lines.

At its core, a closing is a morphological transformation which is just the composition of two other morphological transformations, which are an erosion and a dilation. A dilation is computed using two elements, which are an image and a kernel. In this case, the kernel was a 5x5 pixel window. For each pixel in the image of interest, the center of the kernel is aligned with the image pixel, and the image pixel is assigned the maximum value of all the pixels that are overlapped by the kernel. According to the size of the kernel used, this results in the enlargement and “filling in” of nonzero edge features in the reference edge image.
An erosion is computed similarly to a dilation, however instead of replacing the central image pixel with the maximum value of the pixels overlapped by the kernel, the minimum value is used. This results in a “thinning” of the nonzero edge features in the reference image.

The benefit in using a closing, where an erosion is composed with a dilation, is that the “filling in” effect of the dilation on the nonzero edge features persists, while the enlargement from the dilation and the thinning from the erosion of said features more or less cancels out. Referring again to figure 4, we see the effect of a closing on painted parking lot lines, where the area between the two paint/asphalt transitions, as indicated by the red arrows, is now filled in with nonzero pixels, but the zeroed non-edge pixels immediately outside these exterior lines remain mostly zeroed. Thus the strength of the overall edge feature is enhanced, without a significant loss of positional precision.

3.4 Template Matching

With the reference edge image and orthomosaic edge image generated, a modified template matching algorithm was used to register the orthomosaic edge image with the reference edge image. OpenCV has a function matchTemplate available to calculate the window within an image that best matches a given template, based on a user-specified matching method. However, this function operates via a brute-force translation comparison of the template at every available location in the image. Thus, other important degrees of freedom, primarily rotation and scaling, are not considered by default in the matchTemplate algorithm. Before rotation and scaling could be considered, however, a zero-intensity border was added to the reference edge image, so that the
orthomosaic edge image would never be larger than the reference edge image, despite the rotation and scaling it was subject to.

In order to consider these additional degrees of freedom, a brute-force algorithm was implemented by which the template (the orthomosaic edge image) was subjected to a range of rotational and independent biaxial scaling values, in addition to the translation values searched by the default matchTemplate algorithm. Since this is a computationally expensive method, a coarse initial template matching was performed, searching $x$ and $y$ scaling values 25 percent above and below the initial orthomosaic edge image dimensions in increments of 0.5 percent, and rotation angles 5 degrees above and below the initial orthomosaic edge image orientation, in increments of 0.1 degrees. Following the coarse template matching, a fine template matching was performed, searching $x$ and $y$ scaling values 5 percent above and below the optimized scaling values from the coarse template matching in increments of 0.1 percent, and 1 degree above and below the optimized orientation from the coarse template matching in increments of 0.02 degrees. This corresponds to a bin size of less than or equal to approximately 10 centimeters for both the scaling and rotation ranges, or approximately one-third of the ground sample distance of the imagery provided by WorldView-3. For biaxial scaling, this bound is calculated as the per-increment change in ground sample distance based on the change in image dimensions used to represent the roughly 80x30 meter scene of interest. For rotation, this is calculated as an upper bound on the per-increment change in pixel geolocation, based on the change in geolocation of the farthest points from image center, the four corner pixels of the image, as subjected to one increment of rotation.
Figure 5 shows a flowchart depicting the algorithm used to calculate the optimal orthomosaic scaling, rotation, and translation values to register the orthomosaic edge image with the reference edge image, based on the aforementioned lower and upper bounds for each parameter. It shows three nested operations on the orthomosaic, which are evaluating the X-scale, evaluating the Y-scale, and evaluating the rotation parameter. For each combination of these three parameters, translational template matching is performed, and the score of the best match is stored, along with its corresponding parameter values. Each of these parameters is initialized at its minimum value, and at each decision operation, these parameters will be increased by one step size until they are at their maximum value. Then, the maximized parameter is reset to its minimum value, and its parent parameter is increased by one step size. This is repeated, until the first parameter, the X-scale, reaches its maximal value. At this point, the global highest template matching score is identified, and the parameter values corresponding to it are saved as the optimal orthomosaic orientation parameters.
Figure 5. Flowchart depicting the coarse registration modified template matching algorithm.
For the actual matchTemplate function at the heart of the brute-force algorithm, matching was performed utilizing the cross-correlation method. The cross-correlation result $R$ of a template $T$ at a location $(x, y)$ on an image $I$ is given by

$$R(x, y) = \sum_{x', y'} (T(x', y') \cdot I(x + x', y + y'))$$

where $x', y'$ are the translation components along the $x$ and $y$ dimensions, respectively. Cross-correlation is not commonly used without a normalization factor due to the risk of its maximum yielding a false registration point [47,48]. This is due to the cross-correlation formula disproportionately weighting pixels of higher intensities over pixels of lower intensities. However, for matching the reference edge image and orthomosaic edge image, which are binary apart from fractional interpolated values in the orthomosaic edge image due to rescaling, this is actually advantageous. Rather than equally weighting an edge to edge match and a non-edge to non-edge match, only edge matches receive a positive weighting, due to non-edge pixels having zero intensity. Since additional non-edge border pixels must be introduced edge images, as previously discussed, to enable template matching to be performed, utilizing the cross-correlation formula prevents these additional pixels from affecting the matching outcome.
Chapter 4: Fine Registration

Rather than continuing to use Canny edge images for implementation of the fine registration algorithm, the original grayscale reference image was used, and the grayscale orthomosaic was used, after it had been scaled and rotated according to the results of the coarse registration. This was done so that features could be detected based on the original grayscale pixel intensities, rather than their binary values in the edge images. Additionally, the reference image was subsetted so that only the region identified as the best match for the orthomosaic in the coarse registration was used for the fine registration.

4.1 Good Features to Track

The Good Features to Track algorithm was then implemented to detect keypoints in the reference image and orthomosaic [31]. The Good Features to Track algorithm is used to detect image features, such as corners, salt-and-pepper textures, and other texture patterns, which can be reliably tracked between grayscale images.

Feature selection in the Good Features to Track algorithm is based on the affine motion model relating image intensities between given images I and J and a window in image I:

\[ J(Ax + d) = I(x) \]  \hspace{1cm} 4.1

Where \( d \) is the translation of the feature window’s center, and
\[ A = 1 + D \]

Comprises the 2x2 identity matrix \( 1 \) as well as the deformation matrix:

\[ D = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix} \]

Determining the motion parameters is performed by minimizing the dissimilarity residual

\[ \epsilon = \int \int_W [J(Ax + d) - I(x)]^2 w(x) \, dx \]

where \( W \) is the given feature window and \( w(x) \) is a weighting function, which can simply be unity, or can be a “Gaussian-like” function to more strongly weight the center region of the window. Following the derivation in [31], feature selection is determined by solving a special case of the resulting linear 6x6 system:

\[ Tz = a \]

where

\[ T = \int \int_W \begin{bmatrix} x^2 g_x^2 & x^2 g_x g_y & xy g_x^2 & xy g_x g_y & x g_x^2 & x g_x g_y \\ x^2 g_x g_y & x^2 g_y^2 & xy g_x g_y & xy g_y^2 & x g_x g_y & x g_y^2 \\ xy g_x ^2 & xy g_x g_y & y^2 g_x^2 & y^2 g_x g_y & y g_x g_y & y g_x g_y \\ xy g_x g_y & xy g_y^2 & y^2 g_x g_y & y^2 g_y^2 & y g_x g_y & y g_y^2 \\ x g_x^2 & x g_x g_y & y g_x^2 & y g_x g_y & g_x^2 & g_x g_y \\ x g_x g_y & x g_y^2 & y g_x g_y & y g_y^2 & g_x g_y & g_y^2 \end{bmatrix} w \, dx \]

is a matrix computed from one image with \( g_x, g_y \) denoting elements of the spatial gradient of the image intensity:

\[ g = \left( \frac{\partial J}{\partial x}, \frac{\partial J}{\partial y} \right)^T \]

and where
\[ z = \begin{bmatrix} d_{xx} \\ d_{yx} \\ d_{xy} \\ d_{yy} \\ d_x \\ d_y \end{bmatrix} \tag{4.8} \]

is a vector of the unknown entries of the deformation \( D \) and displacement \( d \), and

\[ a = \int_W \left[ I(x) - J(x) \right] w \, dx \tag{4.9} \]

is an error vector depending on the per-pixel intensity difference in the window \( W \) between the two images \( I \) and \( J \). The matrix \( T \) can then be partitioned as:

\[ T = \int_W \begin{bmatrix} U & V \end{bmatrix} w \, dx \tag{4.10} \]

where \( U \) is a 4x4 matrix, \( Z \) is a 2x2 matrix, and \( V \) is a 4x2 matrix.

However, rather than solve equation 4.5, it is preferable to set the entries of the deformation matrix \( D \) to zero, since on the one hand this motion must be small between successive image frames across which tracking is being performed, and on the other hand the interaction between \( D \) and the displacement \( d \) through the matrix \( V \) in equation 4.10 can lead to additional errors in the displacement if the deformation parameters were to be solved for as well. Thus it is preferable to solve

\[ Zd = e \tag{4.11} \]

where \( e \) contains the last two entries of vector \( a \). This system is used to evaluate whether or not a particular window is a strong feature in a given image according to the criterion
\[
\min(\lambda_1, \lambda_2) > \lambda
\]

where \(\lambda_1\) and \(\lambda_2\) are the eigenvalues of \(Z\) and \(\lambda\) is a predefined threshold that may be specified by the user. This requirement ensures that the feature is both above the image noise level and well-conditioned, that is the feature is neither of constant intensity nor patterned unidirectionally.

In OpenCV, the Good Features to Track implementation performs a few other steps as well. After the corner quality is evaluated at every pixel in an image, non-maximal suppression is performed as detailed in section 3.2.3. Additionally, a minimum quality level can be designated, corresponding to \(\lambda\) in equation 4.12, such that features whose quality is below a certain fraction of the strongest feature’s quality are rejected. Finally, a minimum distance can be designated, such that features are discarded if a stronger feature is located within a radius around the feature of interest less than the minimum distance.

4.2 SIFT Descriptors

Having used the Good Features to Track algorithm to detect keypoints in the reference image and orthomosaic, descriptors for the keypoints were then computed using the SIFT compute algorithm as described in [14]. The first step in descriptor computation for a given keypoint is to sample the image gradient magnitudes and orientations within a window surrounding the keypoint location. This is performed in a 16 x 16 neighborhood around the keypoint, which is divided into 16 different 4 x 4 windows. Then, for each of these 4 x 4 windows, an 8 bin orientation histogram is created. This results in a 128 element vector representing the histograms for each of the 16 total 4x4 windows per keypoint neighborhood, which is later used to match keypoints between images.
4.3 Caveats of GFTT Keypoint – SIFT Descriptor Integration

One of the main benefits of using the SIFT algorithm is scale invariance of the detected features – features detected in an image at one resolution level can usually be detected in the same image at reasonably different resolutions as well. Since the scale invariance of the SIFT algorithm as implemented in OpenCV is ensured within the keypoint detection process and Difference-of-Gaussians algorithm, and not the descriptor generation process, it is important to note that the feature detection/matching process used in this thesis does not ensure the scale invariance of the features [14]. Additionally, rotation invariance is achieved by a normalization process performed in the keypoint detection process, meaning that the features detected in this algorithm are not rotation invariant either. However, since the images of interest in this thesis are already coarsely registered to pixel-level accuracy before the feature detection/matching algorithm is applied, scale and rotation invariance are not necessary requirements. In fact, by removing the need to consider keypoints of different scales and rotations than the values accepted by the coarse registration, potential false matches can be avoided.

4.4 Keypoint Matching

Once the keypoints and descriptors have been calculated for both the reference image and the orthomosaic, the next step was to identify matching keypoints between the images. To ensure greatest accuracy, a brute force matching algorithm was used, which compares each descriptor from one image with every descriptor from the other image to determine the best match. Although of $O(mn)$ complexity, where $m$ is the number of reference image keypoints and $n$ is the number of orthomosaic keypoints, the relatively small size of the reference image and orthomosaic prevented computational difficulty.
However, for real-time applications or processing images of higher resolution, a brute-force approach may prove too computationally expensive. Other, less computationally expensive methods for keypoint matching are further discussed in section 7.2.

Using the brute-force method, the best match was determined using the L2 norm to find the descriptor of minimum distance away from each descriptor in the first image. Cross-check was also utilized to further ensure the accuracy of the results. Using cross-check, a descriptor $I$ in the reference image matched with a descriptor $J$ in the orthomosaic will be accepted only if descriptor $J$ is the best match for descriptor $I$, while descriptor $I$ is also the best match for descriptor $J$. Finally, since the reference image and orthomosaic are coarsely aligned to pixel-level accuracy before the feature detection/matching algorithm is applied, keypoint matches that differ in location between the reference image and orthomosaic by more than a few pixels were discarded as well. Thus, only the highest quality matches are retained to register the orthomosaic.
Chapter 5: Georeferencing the Orthomosaic

Having discussed the algorithms used for coarse and fine registration of the orthomosaic, the next step before overall registration accuracy could be assessed was to georeference the orthomosaic according to the geospatial location of the reference image pixels selected for registration in the previous registration algorithms.

5.1 Image Space Tie Points: Coarse Registration

For the coarse registration algorithm, the four corners of the orthomosaic were used as tie points for georeferencing. The matchTemplate algorithm used in the coarse registration algorithm returns the location of the top left corner of the template, which in this case is the rotated and rescaled orthomosaic edge image, in the image space of the reference edge image. Based on the dimensions of the rotated and rescaled orthomosaic edge image, the locations of the other three corners are easily calculable as well. However, these corner locations correspond to the corners of the orthomosaic after being rotated and rescaled in the brute-force matching algorithm, which can introduce extra zero, no-data pixels as a result. This is because when rotating the image, the rotated coordinates of the image may exceed the original dimensions of the rectangular image array. Figure 6 demonstrates the dimension increase along both axes and no-data pixels that must be introduced to a rotated image to maintain its data type of a rectangular array, by means of a greatly exaggerated rotation of the orthomosaic.
Figure 6. Unrotated image on the left, compared to the rotated image on the right, with no-data pixels introduced along both axes to preserve the rectangular array shape. The red box overlay indicates the dimensions of the left tile.

This in turn alters the image dimensions, which combined with the actual rotation and rescaling of the orthomosaic edge image, means that the geophysical location of a particular pixel in the rescaled and rotated image is not the same as the geophysical location of the pixel with the exact same image-space coordinates in the original orthomosaic edge image.

To calculate the location of the four corners of the original orthomosaic in the image space of the reference edge image, firstly the dimensions of the rescaled, but not rotated orthomosaic edge image are used to locate the four corners of the aforementioned orthomosaic edge image, as centered about the origin of a 2-dimensional coordinate plane. Then, each corner at a location \((x, y)\) is rotated by the same angle \(\theta\) as calculated in the brute-force matching algorithm to the coordinates \((x', y')\) according to the simple rotation equations:
\[ x' = x \cos \theta - y \sin \theta \]
\[ y' = x \sin \theta + y \cos \theta \]

5.1

A simple translation is the only remaining step to obtain the original orthomosaic edge image corner coordinates in the image space of the reference edge image. The \( x \) and \( y \) translation values are simply the location of the center of the rotated and rescaled orthomosaic \((C_x, C_y)\), which are calculated according to the simple formulas

\[ C_x = L_x + (R_x - L_x)/2 \]
\[ C_y = L_y + (R_y - L_y)/2 \]

5.2

where \((L_x, L_y)\) are the coordinates of the top left corner, and \((R_x, R_y)\) are the coordinates of the bottom right corner. Thus, the coordinates for the corners of the original orthomosaic edge image \((X, Y)\) in the image space of the reference image are given by

\[ X = x' + C_x \]
\[ Y = y' + C_y \]

5.3

where \(x'\) and \(y'\) are the coordinates calculated from equation 5.1, and \(C_x\) and \(C_y\) are the coordinates calculated from equation 5.2. In this manner, the coordinates of all four corners of the original orthomosaic edge image were calculated in the image space of the reference edge image.

The other important transformation was calculating the location of the four corners of the orthomosaic edge image in the image space of the Pix4D orthomosaic, since the ultimate goal was to georeference the full resolution orthomosaic, not the downsampled orthomosaic that was manipulated using OpenCV. For the coarse registration tie points, this was simple since the corners simply correspond to the extreme points of the 40925 x 18307 Pix4D orthomosaic. With the coordinates of the coarse registration tie points known in both the image space of the Pix4D orthomosaic and the
image space of the reference image, the coarse registration tie points need only to have their object-space coordinates determined, which is discussed in section 5.3.

5.2 Image Space Tie Points: Fine Registration

For the fine registration algorithm, all of the keypoint matches accepted by the
brute-force matcher and spatial deviation constraint were used as tie points for
georeferencing. As was the case for the coarse registration tie points, their exact location
must be calculated in the full resolution Pix4D orthomosaic, as well as in the image space
of the downsampled orthomosaic based on their locations in the image space of the
rotated and rescaled orthomosaic. This was accomplished by applying a transformation
matrix $M$ to the coordinates of the tie points in the image space of the rotated and rescaled
orthomosaic, where $M$ is given by

$$M = \begin{bmatrix}
    s_x \cos \varphi & -s_x \sin \varphi & x_c \\
    s_y \sin \varphi & s_y \cos \varphi & y_c
\end{bmatrix}  \tag{5.4}$$

and where $s_x$ and $s_y$ are the reciprocals of the $x$ and $y$ scaling parameters calculated from
the coarse registration algorithm, $\varphi$ is the inverse rotation angle calculated from the
coarse registration algorithm, and $x_c$ and $y_c$ are translations in the $x$ and $y$ direction
respectively. $x_c$ and $y_c$ correspond to the difference in location of the central pixel $(C_x, C_y)$
in the orthomosaic before rotation and rescaling, and the central pixel of the rotated and
rescaled orthomosaic $(c_x, c_y)$ after having the scaling and rotation transformation in
equation 5.4 applied. Mathematically, $c_x$ and $c_y$ are calculated by

$$\begin{bmatrix}
    c_x \\
    c_y
\end{bmatrix} = \begin{bmatrix}
    s_x \cos \theta & -s_x \sin \theta \\
    s_y \sin \theta & s_y \cos \theta
\end{bmatrix} \begin{bmatrix}
    C_x \\
    C_y
\end{bmatrix} \tag{5.5}$$

Thus, $x_c$ and $y_c$ are given by
\[
\begin{bmatrix}
    x_c \\
y_c
\end{bmatrix} = \begin{bmatrix}
    c_x - c_x \\
c_y - c_y
\end{bmatrix}
\]

5.6

The coordinates of each tie point in the image space of the original OpenCV orthomosaic are then calculated by the matrix multiplication of

\[
M \cdot \begin{bmatrix}
x \\
y \\
1
\end{bmatrix}.
\]

5.7

Using \(M\) from equation 5.4, and where \((x, y)\) is the location of the tie point of interest in the image space of the rotated and rescaled orthomosaic. The corresponding coordinates in the image space of the reference image are calculated by simply adding the translation offset used to subset the reference image for the fine registration to the coordinates calculated by equation 5.7.

With the tie point coordinates now known in the image space of the reference image, a simple scaling conversion is used to transform the coordinates of the tie points from the image space of the downsampled orthomosaic in OpenCV to the image space of the full-scale Pix4D orthomosaic. This is represented by

\[
X_p = \left(\frac{P_x}{O_x}\right)x
\]

\[
Y_p = \left(\frac{P_y}{O_y}\right)y
\]

5.8

where \((X_p, Y_p)\) are the coordinates of the tie point of interest in the image space of the Pix4D orthomosaic, \(P_x\) and \(P_y\) are the \(x\) and \(y\) dimensions, respectively, of the Pix4D orthomosaic, \(O_x\) and \(O_y\) are the \(x\) and \(y\) dimensions, respectively, of the OpenCV orthomosaic, and \((x, y)\) are the coordinates of the tie point of interest in the image space of the OpenCV orthomosaic.
5.3 Geospatial Data Abstraction Library

After calculating the coordinates of the tie points for both the coarse and fine registration in the image space of the Pix4D orthomosaic and in the image space of the reference image, the next step was to calculate the coordinates of the tie points in object space, that is their coordinates on the surface of the earth. This was accomplished by using the Geospatial Data Abstraction Library (GDAL) to interpret the geospatial location information contained within the reference image GeoTIFF data.

GDAL is an open-source library, first developed in 1998, with the capability of reading and writing a variety of spatial data formats [49]. GDAL is utilized both as an end product, and as a utility for proprietary geospatial application development. Implemented in C++, GDAL is also available in other languages such as Python, allowing straightforward integration with the rest of the Python code in this thesis.

For this thesis, the GetGeoTransform function was used, which returns the coefficients for a transformation between pixel/line image space and geographic/projected coordinate object space from the reference image GeoTIFF data. The coefficients fetched include: pixel west/east resolution, pixel north/south resolution, longitude and latitude of the top-left corner of the image, and two rotation parameters.

Using these coefficients, the object-space geographic coordinates \((X_g, Y_g)\) can be calculated from the image-space reference image coordinates for each tie point according to the equations:

\[
X_g = a + x \cdot b + y \cdot c \\
Y_g = d + x \cdot e + y \cdot f
\]

5.9
where \((x,y)\) are the image-space coordinates, \(b\) is the pixel width, and \(f\) is the pixel height. \(c\) and \(e\) are rotation parameters, which are zero in this case since the imagery is north-up. The upper left corner of the upper left pixel is at position \((a, d)\).

5.4 Image Projection in ArcGIS

With the coordinates of each tie point calculated in both the image space of the Pix4D orthomosaic and the object-space geographic coordinates, the next step was to project the Pix4D orthomosaic into the geographic reference system. The WGS 1984 reference system was used for all of these projections, which were performed using ArcGIS.

ArcGIS is a broadly-used geographic information system, with a wide variety of tools available for map, image, and geographic data manipulation. Of particular import for this thesis was the Warp function. This function applies a transformation to an input image, based on user-provided source and target coordinates, and user-selected transformation type. Each tie point for the Warp function in ArcGIS is entered as a vector

\[
[x\ y\ X_g\ Y_g]
\]

where \((x,y)\) are the coordinates of the tie point of interest in the image space of the reference image, and \((X_g, Y_g)\) are the object-space longitude and latitude, respectively, corresponding to that tie point.

Regarding the transformation type, both first-order and second-order polynomial transformations were used [50]. In two dimensions, a polynomial equation of degree \(n\) can be given by

\[
p(x, y) = \sum_{k}^{n} a_k x^i y^j
\]
with coefficients $a$, and where $i$ and $j$ are permuted subject to $i + j \leq k$. Polynomial equations are fit to an image based on a least-squares adjustment of the tie points as described by equation 5.10. A first-order polynomial warp can correct biaxial translation, scaling, as well as rotation and an obliquity. Second-order polynomials correct these parameters, as well as accounting for biaxial torsion and convexity corrections. First-, second-, and third-order polynomials are frequently used for georeferencing purposes, however higher-order polynomials require more tie points and can introduce greater distortions if tie points are not well-distributed throughout the image.

For the coarse registration algorithm, which calculated only 4 tie points corresponding to the orthomosaic corners, only a first-order transformation was applied. This avoided over-parameterization, as the coarse registration algorithm itself only considered translation, scaling, and rotation parameters. The fine registration algorithm, on the other hand, calculated upwards of 50 tie points corresponding to the various matched features, allowing both first- and second-order polynomial transformations to be independently applied. The accuracy of these various methods was then calculated and compared, as detailed in chapter 6.
Chapter 6: Experimental Results

A total of 8 GCPs were placed in various locations throughout the scene imaged by the orthocopter-mounted Nikon D800. The location of these GCP’s was measured by GPS survey as ground truth, using Virtual Reference Stations (VRS) to achieve an estimated 3D accuracy of 2.5 centimeters or better [51]. To evaluate the overall accuracy of the image registration algorithms, the location of the GCPs was measured in the Pix4D orthomosaic after it was georeferenced according to the coarse and fine registration algorithms. RMSE between the location of the georeferenced GCPs and their GPS-measured locations was used as a measurement of the overall accuracy of these algorithms.

6.1 RMSE Results

Plots of the error for each GCP are shown in figures 7 through 9, and the calculated location of each GCP using each method is shown in figure 10. Results are plotted for the first-order coarse registration, as well as the first- and second-order fine registration results. For visualization, results are projected into Universal Transverse Mercator Zone 17 North, and a linear offset of -313,070 meters easting and -4,440,970 meters northing was applied to center results about the origin. These plots show the difference between GCP location as measured by GPS and as projected in the noted
georeferenced image, with an arrow in the direction of the error. The error arrows have been enlarged by a factor of 5 for improved readability.
Figure 7. Error plot of GCPs following coarse registration and first-order georeferencing. Error bars are magnified by a factor of 5.
Figure 8. Error plot of GCPs following fine registration and first-order georeferencing. Error bars are magnified by a factor of 5.
Figure 9. Error plot of GCPs following fine registration and second-order georeferencing. Error bars are magnified by a factor of 5.
Figure 10. GCP locations as calculated from GPS survey, coarse registration, and first- and second-order fine registration.
As seen in figure 7, the coarse registration algorithm results in roughly uniform error for each GCP. Globally, there is no notable systematic error either. For the fine registration in figure 8 and 9 however, the extreme GCPs vertically in both directions show more error than the inner GCPs, both using the first- and second-order transformations. A contributing factor to this is that the fine registration algorithm did not detect any keypoint matches to use as tie points near these areas. Results are shown numerically in table 1 in two categories: first, using all of the GCPs, and second, using only “internal” GCPs, which are the GCPs which were bounded above and below by a tie point, in both their northing and easting.

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE (Meters)</th>
<th>All GCPs</th>
<th>Internal GCPs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Coarse Registration</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Order Polynomial</td>
<td>0.59</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td><strong>Fine Registration</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First Order Polynomial</td>
<td>0.27</td>
<td>0.18</td>
<td></td>
</tr>
<tr>
<td>Second Order Polynomial</td>
<td>0.38</td>
<td>0.32</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. RMSE results for first-order coarse registration, and first- and second-order fine registration.

As seen in table 1, with respect to the 0.3 meter Esri reference imagery, the coarse registration algorithm was able to achieve global pixel-level georeferencing accuracy. The fine registration algorithm showed substantial improvement, particularly using the first-order polynomial transformation, which achieved subpixel-level georeferencing accuracy both using all GCPs, and using only the internal GCPs. However, the internal GCP georeferencing accuracy was significantly better than that of all the GCPs. Often,
higher-order polynomials are able to achieve superior georeferencing RMSE due to their ability to accommodate more complex geometric image distortions [52]. However, higher-order polynomials also risk over-parameterizing the data, and are more dependent on homogenous distribution of a larger number of tie points throughout the image. Sparse tie points can lead to excessive image distortion and higher RMSE when applying higher-order polynomial transformations, as found in [52]. The edges of the UAS orthomosaic however featured relatively uniform terrain, particularly grassland where no tie points were detected, while tie points were detected only in the parking lot and built-up area closer to the center of the image. This weak spatial distribution of tie points may likely have been a contributing factor to the higher RMSE attained from the second-order polynomial transformation.
Chapter 7: Conclusions and Future Work

In this thesis, imagery from the Esri World Imagery geobrowser was used as reference imagery for UAS image registration, to address the feasibility of using geobrowser data for automatic registration. Georeferencing accuracy was calculated for both pixel-based and sequential pixel- and feature-based registration algorithms, as both have been implemented in literature for the purpose of registering UAS imagery.

7.1 Conclusions

Results indicate that the inclusion of feature-based registration can obtain superior registration accuracy compared with solely pixel-based registration, with the caveat that accuracy suffers in regions of sparse feature density, particularly for the feature-based registration methods. In such regions, varying UAS altitude or using a sensor with a wider field-of-view may be beneficial, in order to increase the total spatial coverage, and thus number of features in each image. Accuracy improvements from this method could vary greatly between different scenes of interest however, depending on the corresponding land use/land cover. Using reference imagery obtained closer to the date of UAS imagery collection, whenever possible, would also minimize loss of matching features due to physical scene changes. Nevertheless, subpixel-accuracy georeferencing was demonstrated using feature-based registration and geobrowser reference imagery,
with the additional caveat that the spatial resolution, as well as geolocation accuracy of geobrowser imagery may vary based on location.

While there are drawbacks associated with using geobrowser imagery for the purpose of georeferencing UAS imagery, such as the aforementioned variations in spatial resolution, temporal resolution, and so forth, there are many positives to this method as well. First of all, no fieldwork in the form of GCPs is required, allowing for faster processing and potentially lower cost in collecting and georeferencing UAS imagery. Sub-pixel accuracy of 0.18 meters RMSE was demonstrated, which is considered appropriate for many applications in precision agriculture [53]. The accuracy improvement observed from incorporating a feature-based registration algorithm upon the results of an edge image template matching registration algorithm may also be able to provide more accurate results to UAS navigation filters currently using only the latter [9,10]. Moreover, the coarse-fine registration algorithm as discussed in this thesis is not restricted to using geobrowser imagery, and if desired, could be applied using reference satellite imagery with more accurate georeferencing, and taken much closer to the date of UAS imagery.

7.2 Future Work

While successful in terms of achieving the best possible results, some of the structures used in this thesis could be improved in terms of their computational speed for real-world applications. Particularly, the coarse registration template matching algorithm as implemented is a brute-force search over a range of rotation, 2D scaling, and 2D translation values. This prohibits real-time, high-precision registration due to run times
that can be several hours long on non-specialized machines, depending upon the desired accuracy. However, there are a variety of techniques that could be explored to increase processing speed. First, a more rigorous coarse-to-fine pruning template matching algorithm could be implemented, for the purpose of increasing speed, as well as ensuring that the absolute best match is found in every situation [54]. Other techniques, such as fast convolutions using Fourier transforms, have been demonstrated to increase computation speed as well, particularly for large images [55].

In addition to template matching, a few other algorithms, such as the brute-force descriptor matching algorithm, could potentially be reduced in complexity. As discussed in section 4.4, brute-force descriptor matching is of $O(mn)$ complexity, where $m$ is the number of reference image keypoints and $n$ is the number of orthomosaic keypoints. Structures such as KD-trees, which are binary search trees used to store points of K dimensionality, can be used to decrease computational complexity [56]. KD-trees can be constructed with $O(n \log n)$ complexity, assuming a KD-tree is built for the orthomosaic keypoints, and keypoint descriptors can then be matched with $O(m \log n)$ complexity. While the KD-tree and many derivatives thereof have been used extensively, other methods have been developed and implemented with further reduced computational complexity, notably including the Fast Approximate Nearest Neighbor Search Library (FLANN) [57]. FLANN offers the benefit of automatic algorithm selection and configuration for nearest-neighbor matching, with results that can achieve an order of magnitude faster computation compared to other methods. While there are potential drawbacks to approximate nearest-neighbor techniques, such as achieving “near-optimal”
accuracy compared to the optimal accuracy achieved by brute-force and other techniques, the gains in computation speed make real-time applications for large images much more feasible.

Finally, real-world applications of the algorithms in this thesis would benefit substantially from the ability to process the full resolution UAS orthomosaic, not necessarily for the edge image template matching, but at least for the feature-based registration algorithm. As mentioned in chapter 3, OpenCV was unable to manipulate the full resolution UAS orthomosaic due to its large size. Using a different library or program with the ability to manipulate the full orthomosaic could significantly increase the georeferencing accuracy, as the full spatial resolution of the UAS imagery could be exploited for feature detection and localization, rather than just downsampled imagery.
References


