12-26-2014

Terrain Referenced Navigation Using SIFT Features in LiDAR Range-Based Data

Matthew T. Leines

TERRAIN REFERENCED NAVIGATION USING SIFT FEATURES IN LIDAR RANGE-BASED DATA

THESIS

Matthew T. Leines, Capt, USAF

AFIT-ENG-MS-14-D-47

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-14-D-47

TERRAIN REFERENCED NAVIGATION USING SIFT FEATURES IN LIDAR RANGE-BASED DATA

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Matthew T. Leines, BSECE, MSECE

Capt, USAF

December 2014

AFIT-ENG-MS-14-D-47

TERRAIN REFERENCED NAVIGATION USING SIFT FEATURES IN LIDAR RANGE-BASED DATA

Matthew T. Leines, BSECE, MSECE
Capt, USAF

Approved:

/signed/	11 Dec 2014
John F. Raquet, PhD (Chairman)	Date
/signed/	20 Nov 2014
Kyle J. Kauffman, PhD (Member)	Date
/signed/	20 Nov 2014
Alan L. Jennings, PhD (Member)	Date

AFIT-ENG-MS-14-D-47

Abstract

The use of GNSS in aiding navigation has become widespread in aircraft. The long term accuracy of INS are enhanced by frequent updates of the highly precise position estimations GNSS provide. Unfortunately, operational environments exist where constant signal or the requisite number of satellites are unavailable, significantly degraded, or intentionally denied. This thesis describes a novel algorithm that uses scanning LiDAR range data, computer vision features, and a reference database to generate aircraft position estimations to update drifting INS estimates. The algorithm uses a single calibrated scanning LiDAR to sample the range and angle to the ground as an aircraft flies, forming a point cloud. The point cloud is orthorectifed into a coordinate system common to a previously recorded reference of the flyover region. The point cloud is then interpolated into a Digital Elevation Model (DEM) of the ground. Range-based SIFT features are then extracted from both the airborne and reference DEMs. Features common to both the collected and reference range images are selected using a SIFT descriptor search. Geometrically inconsistent features are filtered out using RANSAC outlier removal, and surviving features are projected back to their source coordinates in the original point cloud. The point cloud features are used to calculate a least squares correspondence transform that aligns the collected features to the reference features. Applying the correspondence that best aligns the ground features is then applied to the nominal aircraft position, creating a new position estimate. The algorithm was tested on legacy flight data and typically produces position estimates within 10 meters of truth using threshold conditions.

Table of Contents

Page

List of Figures

Figure Page

Figure Page

Figure Page

List of Tables

List of Acronyms

Acronym	Definition
AFIT	Air Force Institute of Technology
ALS	Airborne Laser Scanning
ANT	Autonomy and Navigation Technology
ASPN	All-Source Positioning and Navigation
DCM	Direction Cosine Matrix
DEM	Digital Elevation Model
DoG	Difference of Gaussian
ECEF	Earth Centered Earth Fixed
ENU	East North Up
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
ICP	Iterative Closest Point
IERS	International Earth Rotation Service
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
ITRS	International Terrestrial Reference System
LiDAR	Light Detection And Ranging
LaDAR	Laser Detection And Ranging
LCCP	Lambert Conformal Conic Projection
LLh	Latitude Longitude and Height
MLS	Mean Least Squares
MRSE	Mean Radial Spherical Error
MSE	Mean Sum of Errors

Acronym	Definition
MSL	Mean Sea Level
NAD83	North American Datum of 1983
NED	North East Down
NOAA	National Oceanic and Atmospheric Administration
NGA	National Geospatial-Intelligence Agency
NGS	National Geodetic Survey
NIMA	National Imagery and Mapping Agency
OSIP	Ohio Statewide Imagery Program
PCA	Principal Component Analysis
RANSAC	Random Sample Consensus
RMSE	Root Mean Squared Error
RPY	Roll Pitch Yaw
SIFT	Scale Invariant Feature Transform
SITAN	Sandia Inertial Terrain Aided Navigation
SLAM	Simultaneous Localization and Mapping
SPCS	State Plane Coordinate System of 1983
SSE	Sum of Squared Error
TERCOM	Terrain Contour Matching
TERPROM	Terrain Profile Matching
TIN	Triangular Irregular Network
TRN	Terrain Referenced Navigation
TMP	Transverse Mercator Projection
UAV	Unmanned Ariel Vehicle
WGS84	World Geodetic System of 1984

TERRAIN REFERENCED NAVIGATION USING SIFT FEATURES IN LIDAR RANGE-BASED DATA

I. Introduction

Global Navigation Satellite Systems (GNSS) have found uses in a variety of applications over the last decade. Predominately used for providing high precision location and timing information, GNSS capable devices are now used nearly universally in aircraft, ground vehicles, smart phones, and are critical even in farming, banking and power transmission [106]. In aircraft navigation, GNSS allows precise flight schedules and coordination, weather and collision avoidance, and safe flying and landings in inclement weather. For trajectory estimation, frequent high precision GNSS updates can be used to correct small cumulative errors from the Inertial Navigation System (INS) sensor measurements.

Modern navigation is possible because of the widespread use of INS onboard aircraft. The INS contains accelerometers and gyroscopes, sensors which measure specific force and angular rotation rate respectively. Processing INS sensor data creates an estimate of the aircraft trajectory, defined as the aircrafts position and orientation history. The trajectory provided by the INS in this way is one type of dead reckoning. The INS provides this estimate by relying only on its own past trajectory history and internal sensors, and because of this it is popular for safety, security and military concerns.

While dead reckoning navigation information is very useful, it is not always used as a singular trajectory source. Small measurement errors resulting from drift, thermal noise, and even manufacturing defects will produce small errors in the INS estimation [110]. Over time INS errors grow at a rate dependant on the quality of the device unless the INS data is compared with another trajectory information source to create an error estimation. Modern GNSS commonly fulfills this role with high precision. Despite widespread use, GNSS does have some operational limits [45] and can be degraded by poor satellite geometry or local geography. Examples of degraded or blocked GNSS signal environments include dense foliage, indoors, under water, underground, under a bridge, canyons (near to a mountain, in low lands, or in an actual canyon), and urban canyons (between buildings in a city). More concerning are environments where GNSS is blocked or degraded by human activities. A region can be accidentally jammed by local broadcasts, deliberately jammed, or even spoofed [125]. These cases are of more relevant concern for aircraft, where strong geometry and signal strength would normally be expected and are then suddenly lost or become unreliable.

1.1 Benefits of LiDAR

In GNSS degraded scenarios or to simply reduce reliance on GNSS, a reliable trajectory estimation must be provided to the INS to prevent error growth. The use of Light Detection And Ranging (LiDAR) data is attractive because of its growing popularity in mapping, potentially high precision, and availability of data products. The proposed method focuses on the uses of range information provided from Airborne Laser Scanning (ALS) sensors and is inspired by approaches dealing with LiDAR mapping and feature-based object recognition.

1.2 Research Objectives

Determining an aircraft position in a GNSS denied environment by utilizing ALS is partitioned into 3 primary research objectives:

1. Develop an algorithm for using airborne LiDAR range information for positioning.
2. Test the developed algorithm using real flight and reference data.
3. Identify key algorithm performance parameters.

1.3 Assumptions

To implement the position estimation method proposed in this thesis, several assumptions are made. The aircraft is assumed to be equipped with a single scanning LiDAR facing nadir and its scanning sweep is coincident with the aircraft wings. The aircraft is also assumed to be equiped with an INS with possible drift errors in its position estimate information, any errors in the attitude information are considered negligible. It is assumed that each LiDAR range sample is able to be time tagged with the position and attitude of the aircraft at that time. It is also assumed that a reference database is available to access previously acquired LiDAR point clouds of the intended flyover region.

1.4 Thesis Overview

This thesis details a Terrain Referenced Navigation (TRN) method based on range data collected from scanning LiDAR sensors and processed using Scale Invariant Feature Transform (SIFT) features. The details of the proposed method is presented in the proceeding chapters. Chapter 2 presents background information related to the proposed method in several parts; the mathematical notation used, reference coordinate systems, coordinate system transforms, computer vision algorithms such as SIFT feature extraction, traits of LiDAR sensors, topographic registration using ALS, the field of TRN, and concludes with a description of the most current LiDAR-based TRN approaches. Chapter 3 details each step of algorithm for LiDAR range-based TRN developed in this thesis. Chapter 4 tests the developed algorithm using real flight and reference data under 2 different initial condition cases, and further explores the sensitivity of the algorithm's performance to changes in 3 threshold conditions on estimated position accuracy and availability. This thesis concludes with a summary of the test results and a short description of possible algorithm improvements and future follow-on work. An appendix is also provided showing each of the 66 image matches used in the algorithm tests.

II. Background

This chapter covers the various components of navigation theory, features, LiDAR sensors, and remote sensing of topography, necessary to understand the applications of TRN. The chapter concludes with a brief discussion of related works in the field of TRN and several papers of special interest to this research effort.

2.1 Mathematic Notation

This section reviews basic mathematical notation commonly utilized in navigation involving coordinate frames and coordinate frame transforms [11, 110, 124].

2.1.1 Basic Notation.

The following basic notation will be held throughout the dissertation:

- **Scalars:** Scalars will be represented in *italics*; (e.g. x or X).

- **Vectors:** Vectors will be noted by lower case letters in *bold*. The scalar elements of a vector will be noted at first use and arranged in brackets. E.g, **p** is a three element vector containing component scalars X, Y, and Z, shown as

$$\mathbf{p} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.1}$$

- **Matrices:** Matrices will be noted by upper case letters in *bold*. The scalar elements of a matrix will be denoted with a double subscript indicating row and column number respectively and arranged in brackets. E.g, **P** is a $2x3$ element matrix consisting of A_{ij} elements, shown as

$$\mathbf{P} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{bmatrix}. \tag{2.2}$$

- **Transpose:** The transpose of a vector or matrix will be identified by a superscript T. E.g, if

$$\mathbf{p} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}, \text{then} \qquad \mathbf{p}^T = \begin{bmatrix} X & Y & Z \end{bmatrix}. \tag{2.3}$$

- **Estimated:** An variable resulting from an estimation of random variables will be noted with a *hat*. E.g, $\hat{\mathbf{p}}$.

- **Sets:** A matrix made of multiple instances of vector data will be represented with a *bar*. E.g, a set of N coordinates

$$\bar{\mathbf{p}} = \begin{bmatrix} X_1 & X_2 & ... & X_N \\ Y_1 & Y_2 & ... & Y_N \end{bmatrix}. \tag{2.4}$$

- **Reference Frames:** Variables will frequently be expressed in their current reference frame, noted as a superscript. Scalar components of a vector expressed in a reference frame will share a similar subscript. E.g. $\mathbf{p}^a$ is a vector represented in the a frame, shown as

$$\mathbf{p}^a = \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix}. \tag{2.5}$$

- **Direction Cosine Matrices:** A Direction Cosine Matrix (DCM) is a matrix operator $\mathbf{C}$ representing a transformation from the subscript's frame into the superscript's frame, given that the two frames are right handed and orthogonal. The DCM $\mathbf{C}_b^a$ would be a rotation from the b frame into the a frame. DCM are discussed in more detail in the following section.

2.1.2 Direction Cosine Matrices and Euler Angles.

Coordinate transformations between reference frames in this research are performed with a compact notation called a DCM. A DCM is a $3x3$ matrix where the scalar component elements represent the cosine of the angle between the axis in one frame projected into another, written as:

$$\mathbf{C}_b^a = \begin{bmatrix} c_{11} & c_{12} & c_{13} \\ c_{21} & c_{22} & c_{23} \\ c_{31} & c_{32} & c_{33} \end{bmatrix}. \tag{2.6}$$

The DCM $\mathbf{C}_b^a$ can be used to transform a coordinate vector $\mathbf{p}$ from frame b into frame a with the following notation:

$$\mathbf{p}^a = \mathbf{C}_b^a \mathbf{p}^b. \tag{2.7}$$

DCM have the following useful properties:

$$\begin{aligned} Det(\mathbf{C}_b^a) &\equiv \|\mathbf{C}_b^a\|_2 = 1 \\ \mathbf{C}_a^b &= (\mathbf{C}_b^a)^{-1} = (\mathbf{C}_b^a)^T \\ \mathbf{C}_a^c &= \mathbf{C}_b^c \mathbf{C}_a^b \end{aligned} \tag{2.8}$$

if the two reference frames are both orthogonal and right-handed [110].

A coordinate transform using a DCM can also be represented as a series of single axis rotations performed in order. The components of these rotations are referred to as Euler angles. As an example, if a reference frame b can be related to a new frame a by a rotation ψ about the z axis, followed by a rotation θ about the y axis, and concluded with a final rotation ϕ about the x axis, then the resulting DCM would be written as:

$$\mathbf{C}_b^a = \mathbf{C}_x \mathbf{C}_y \mathbf{C}_z \tag{2.9}$$

where each rotation can be expressed in matrix form as:

$$\text{rotation } \psi \text{ about the } z\text{-axis, } \mathbf{C}_z = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.10}$$

$$\text{rotation } \theta \text{ about the } y\text{-axis, } \mathbf{C}_y = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.11}$$

$$\text{rotation } \phi \text{ about the } x\text{-axis, } \mathbf{C}_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{2.12}$$

and will hold to the properties listed in Equation (2.8) [110].

2.1.3 Reference Frame Coordinate Systems.

When discussing location and navigation, it is important to be very clear in how a location is defined. There are many ways to describe a location mathematically, so a brief description of the standard coordinate reference frames used in this research are detailed in this section. All of these reference frames have orthogonal and right-handed axis sets in $\Re^3$ space. Additional reference frames and further details than those presented can be found in [11, 71, 110].

2.1.3.1 Earth Centered Earth Fixed Frame.

The Earth Centered Earth Fixed (ECEF) reference frame has its origin at the defined center of the Earth. A position coordinate in ECEF or *e*-frame, will be expressed as

$$\mathbf{p}^e = \begin{bmatrix} X_p & Y_p & Z_p \end{bmatrix} \tag{2.13}$$

and is shown in Figure 2.1.

The *X* axis lies on the intersection of the Greenwich meridian plane and the Earth equatorial plane. The *Y* axis lies on the equatorial plane at 90 degrees east of the Greenwich

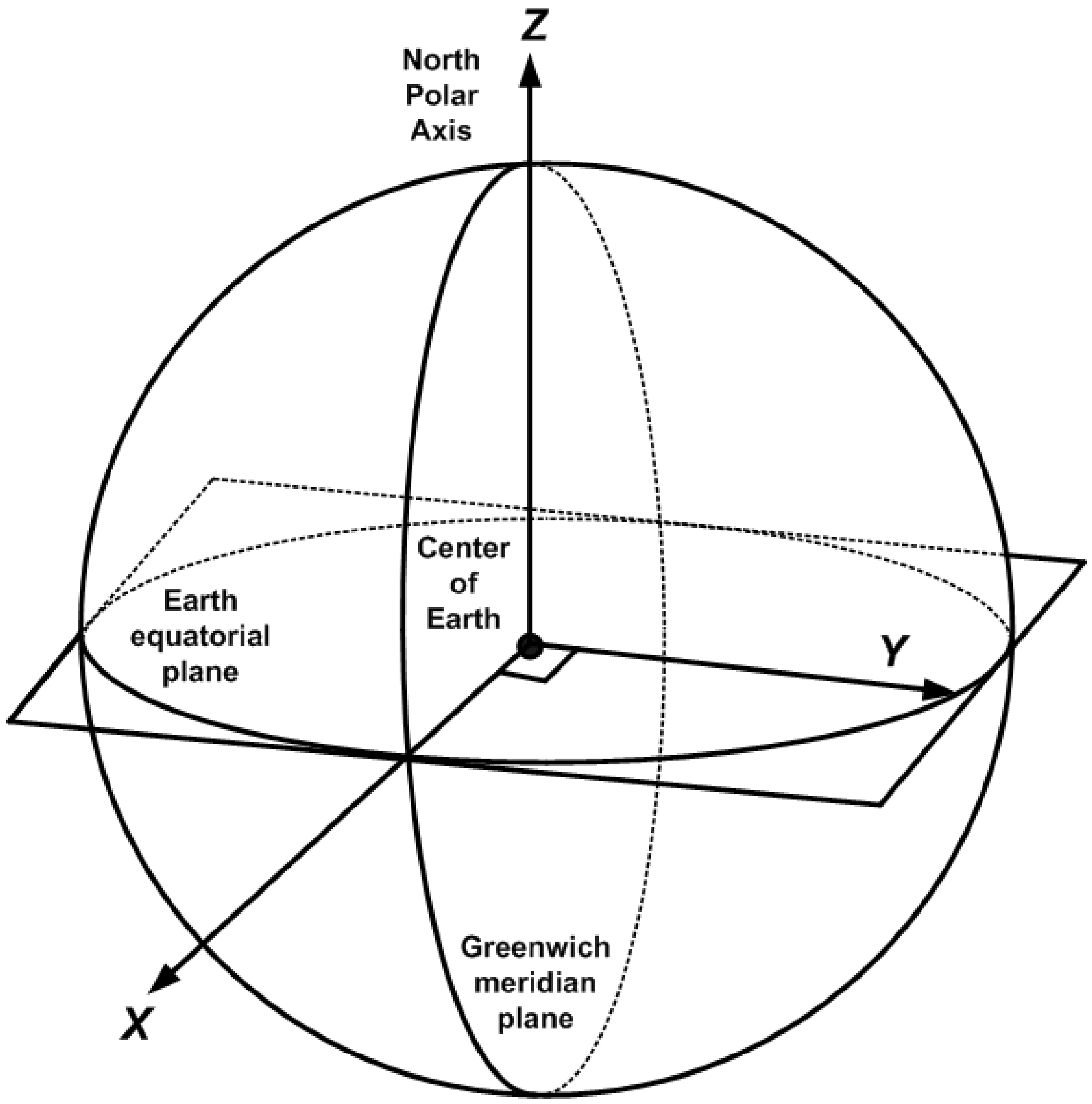


Figure 2.1: The Earth Centered Earth Fixed Reference Frame

meridian. These planes are fixed to the Earth's sphere as it rotates around the *Z* axis that runs through the north polar axis. In this system, a coordinate point is measured in meters from the center of the Earth, and being a true cartesian coordinate system, offers some simplification in navigation computations. This system is also called a geocentric system, in that its origin is coincident with the calculated center of the earth.

2.1.3.2 Local Navigation Frame.

The local navigation reference frame has its origin at a predefined location, typically a surface point on Earth. The navigation frame is a geographic system and is best used for navigation in a small area of interest similar to a local map. Definition of the origin has some impact on the accuracy of the navigation frame over long distances. A general approach is to choose an origin with some use with the application, for example the take off point on a runway may be a good choice for a short local area flight as the airport is a well known location and appears on a variety of maps. Using a static origin does lend itself to growing errors as the distance grows large. As a purely cartesian frame, the local navigation frame does not account for the curvature of the earth, and a strait and level flight would see the distance to the ground change even as altitude remains steady. To remedy this, the position in a local navigation frame could be calculated with its origin centered on the current aircraft position. This creates a navigation frame position that remains current with the curvature of the earth, but must be calculated at each needed time. Some formal local navigation systems have origins (examples in Section 2.2.3) specified to meet certain regional accuracy requirements.

There are many variations on what local navigation frame coordinates are defined as, but most can be reduced into two common systems. First is the North East Down (NED) system, representing the three axis with an origin at the predefined point. NED is more commonly used in airborne oriented navigation, with the definition of down being the distance to the origins altitude in the direction of the local gravity vector. The second, East North Up (ENU), system is more common to terrestrial navigation, though both systems can be used interchangeably. ENU shares the definitions of the north and east axis, although they are swapped to maintain a right-handed axis set. The up direction refers to elevation above the origins' altitude in the direction against the local gravity vector. Both systems are shown in Figure 2.2. In this research, a position in any local navigation frame will be a vector given in the specific navigation frame system, with origin $\mathbf{p}_0$ shown as

$$\mathbf{p}_0^{NED} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{2.14}$$

$$\mathbf{p}^{NED} = \begin{bmatrix} X_{NED} & Y_{NED} & Z_{NED} \end{bmatrix}^T \tag{2.15}$$

and

$$\mathbf{p}_0^{ENU} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{2.16}$$

$$\mathbf{p}^{ENU} = \begin{bmatrix} X_{ENU} & Y_{ENU} & Z_{ENU} \end{bmatrix}^T \tag{2.17}$$

respectively. Transforming between a given NED into a ENU frame, or ENU into NED, can be defined by a single DCM

$$\mathbf{C}_{NED}^{ENU} = \mathbf{C}_{ENU}^{NED} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2.18}$$

where

$$\mathbf{p}^{ENU} = \mathbf{C}_{NED}^{ENU}\mathbf{p}^{NED} \tag{2.19}$$

$$\mathbf{p}^{NED} = \mathbf{C}_{ENU}^{NED}\mathbf{p}^{ENU}. \tag{2.20}$$

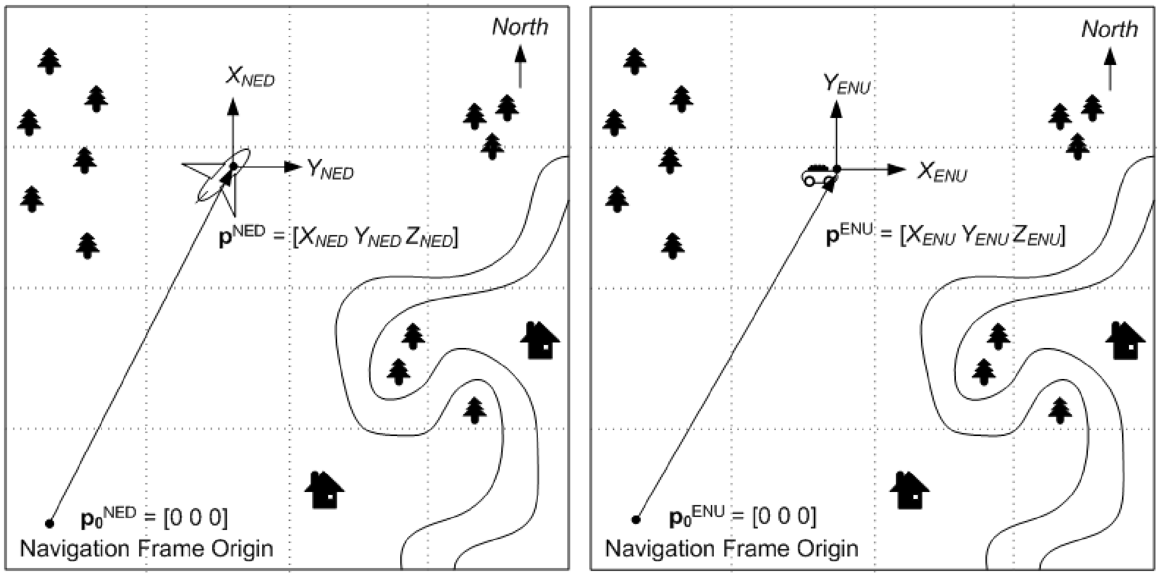


Figure 2.2: Top down view of local navigation frame with origin. Note that the rotation between NED and ENU causes NED north to align with the X axis, while in ENU north aligns with the Y axis.

2.1.3.3 Body Frame.

The body reference frame represents location related to the structure of the navigating vehicle. The origin of the body frame is often shown as the approximate center of mass of the vehicle, but more accurately it is collocated with the origin of the INS. The axis of the body frame are arranged with X in the heading direction, Y is out the right side, and Z points out the bottom of the vehicle. The body reference frame is rigidly attached to the structure of the vehicle, and is useful in defining locations of sensors and equipment that are also attached to the vehicle. With respect to other reference frames, the body frame origin makes a useful single point representation of the vehicle. An example of the body frame as attached to an aircraft is given in Figure 2.3. A position coordinate in the body

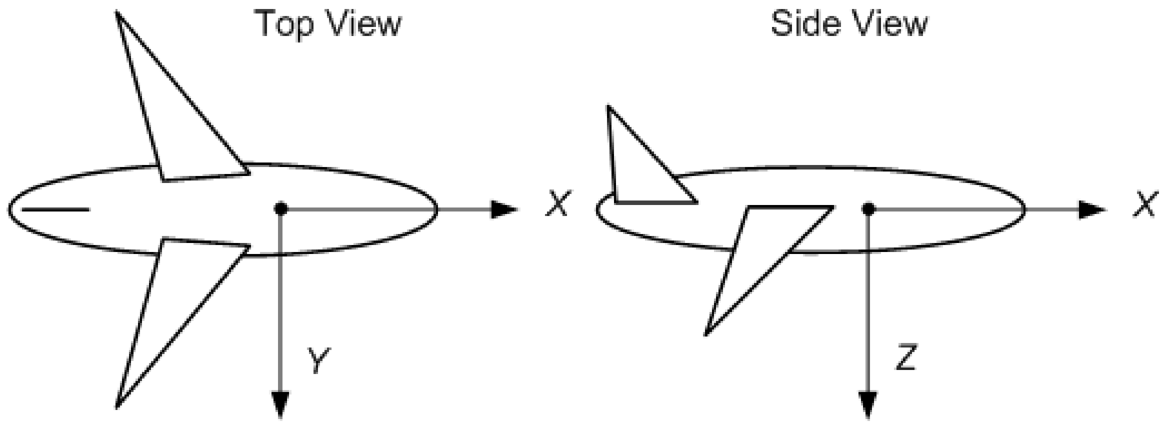


Figure 2.3: Top and side views of the body frame as attached to an aircraft.

frame will be a vector $\mathbf{p}^b$ with scalar components given as

$$\mathbf{p}^b = \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix}. \tag{2.21}$$

2.1.3.4 Sensor Frame.

The sensor reference frame refers to a device that collects data needed for use in other frames. Sensors are assumed rigidly attached to a location in the body frame, but their collected data must be further processed from the sensors point of view. For example a flash LiDAR or camera attached to the belly of an aircraft would record images of the ground, and to define what the sensor sees, a reference frame specific to the device is rigidly attached. Hardware features or limitations (e.g, focal length, aperture, wavelength, etc...) environmental conditions, and noise and error sources native to the sensor are all represented in the sensor frame. The sensor frame origin of many imaging devices is located at the lens focal point [124], and with respect to scanning LiDAR the reflection point on the scanning mirror [93]. This research predominately uses a scanning LiDAR for the sensor as shown in Figure 2.4, and the physical device is basically an electronics box with a glass pane covering the source, rotating mirrors and detector. The Z axis will be through the center of the aperture, the X axis aligns with the scanning plane created by the mirrors, and the Y axis completes the orthogonal right-handed frame. A range sample from a scanning LiDAR will be a vector $\mathbf{p}^s$ with origin scalar components given in cartesian coordinates as

$$\mathbf{p}^s = \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} \tag{2.22}$$

or in polar coordinates as

$$\mathbf{p}^s = \begin{bmatrix} R \\ \theta \end{bmatrix} \tag{2.23}$$

where R_s is the distance, or the slant range, to the ground point from the LiDAR, and θ is the scan, look, or fire angle of the LiDAR [93].

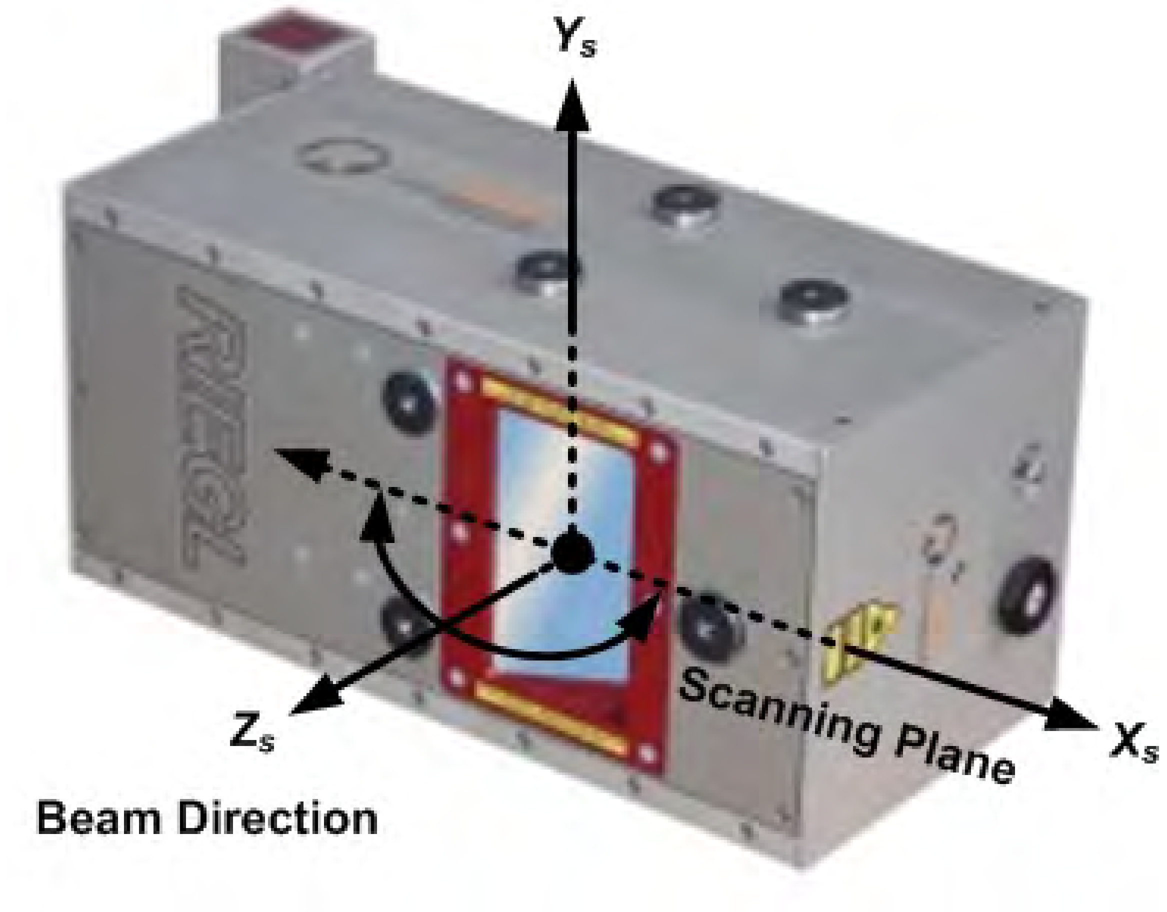


Figure 2.4: Example of a scanning LiDAR electronics box with the attached sensor frame axis [84].

2.2 Transformations Between Coordinate Frames

Translating vector position information from one coordinate frame to another to resolve navigation problems requires coordinate transformations. Many of these transforms are in common use with standard definitions and will be defined in this section. Transforms involving local navigation frames and sensor frames tend to be more application specific, they will be introduced in this section but defined as used later in Section 2.5. This section will cover basic terminology and usage of the various types of transforms utilized in this research. Much greater detail on these topics can be found in the following texts [26, 71, 110].

2.2.1 Datums.

A datum is a reference of Earth, useful for calculating positions. Use of datums is critical to understanding how to calculate positions or even to understand GNSS coordinates. The following sections explain the two datum types and how to determine a 3D surface position utilizing them.

2.2.1.1 Ellipsoid Earth Models.

While geocentric ECEF coordinates are useful to describe a point on or inside the earth they can be cumbersome for human understanding of points directly on Earth's surface. Since Earth is somewhat elliptical in shape, and rather lumpy at that, there are several systems available to better account for Earths actual shape. These elliptical estimates are referred to as geodetic systems and are described by the use of a datum, each resulting in a ECEF reference system with a origin point differing from the geocentric definition in Section 2.1.3.1.

The definition of the ECEF is measured in cartesian coordinates but geodetic systems are in datum specific elliptical coordinates, specifically Latitude Longitude and Height (LLh). A position coordinate in a frame expressed in a geodetic reference system will be a vector given in the datum's frame with scalar components given as

$$\mathbf{p}^{datum} = \begin{bmatrix} \varphi \\ \lambda \\ h \end{bmatrix} \tag{2.24}$$

to represent a position in Latitude and Longitude in radians and the height in meters above the ellipsoid respectively. An example of geodetic coordinates is shown in Figure 2.5.

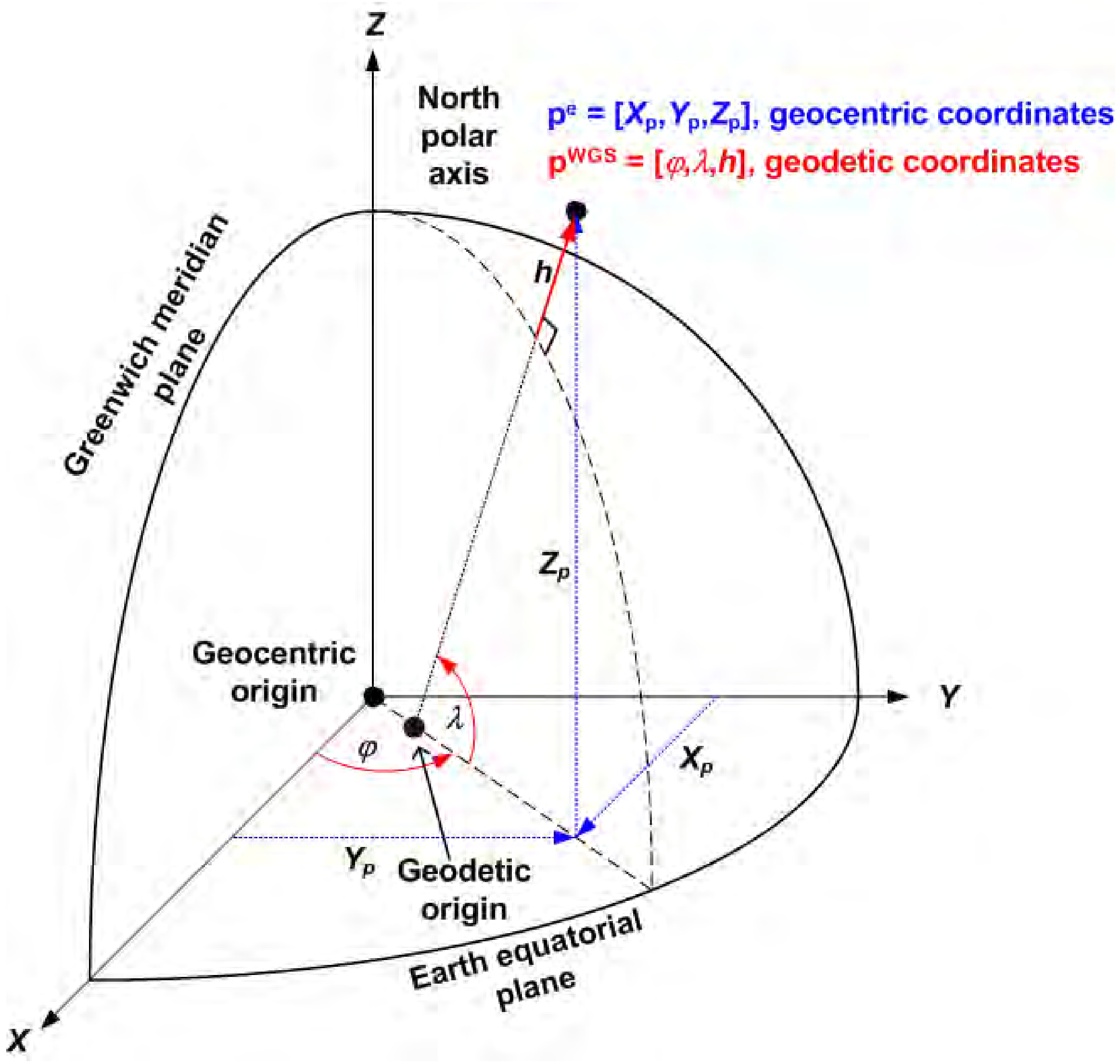


Figure 2.5: Geocentric and geodetic representation of the same point on Earth's surface [71]

There are many datums available to chose from [69], some are defined to best describe the mean sea level over the entire earth, others are used to improve accuracy over the global datums in smaller regions or even individual countries. Of particular interest to this research are the World Geodetic System of 1984 (WGS84), NAD83, and International Terrestrial Reference System (ITRS) datums. WGS84 and ITRS are both global datums that define a world wide sea level ellipsoid, but where developed by National Geospatial-Intelligence Agency (NGA)[1] and International Earth Rotation Service (IERS)[2] respectively, differing slightly in exact definition [99]. WGS84 is of special interest as it is most commonly used to express Global Positioning System (GPS) coordinates. NAD83 is an example of a datum that defines a global ellipsoid that is only accurate over a specific region, in this case the datum is best over the North American Continent. Datums can be completely described by the semi-major axis a of the ellipsoid in meters, the unitless ellipsoid rate of flattening f, and derived ellipsoid eccentricity e [99]. The three datums of interest are defined in Table 2.1.

Table 2.1: Ellipsodial Datum Definitions [71, 99]

Datum	a	f	e
NAD83	6378137.0	$\frac{1}{298.257222101}$	0.0818191910428
WGS84	6378137.0	$\frac{1}{298.257223563}$	0.0818191908426
ITRS	6378136.49	$\frac{1}{298.25645}$	0.0818192967682

Conversions from a geodetic coordinate system to ECEF is fairly strait forward following

$$\mathbf{p}^e = \begin{bmatrix} X_p \\ Y_p \\ Z_p \end{bmatrix} = \begin{bmatrix} (R_n + h)\cos\varphi\cos\lambda \\ (R_n + h)\cos\varphi\sin\lambda \\ \left(R_n(1 - e^2) + h\right)\sin\lambda \end{bmatrix} \tag{2.25}$$

where

$$R_n = \frac{a}{\left(1 - e^2\sin^2\varphi\right)^2} \tag{2.26}$$

[1]National Imagery and Mapping Agency (NIMA) was incorporated into the NGA in 2003

[2]IERS was renamed into the International Earth Rotation and Reference Systems Service, keeping the old acronym, in 2003

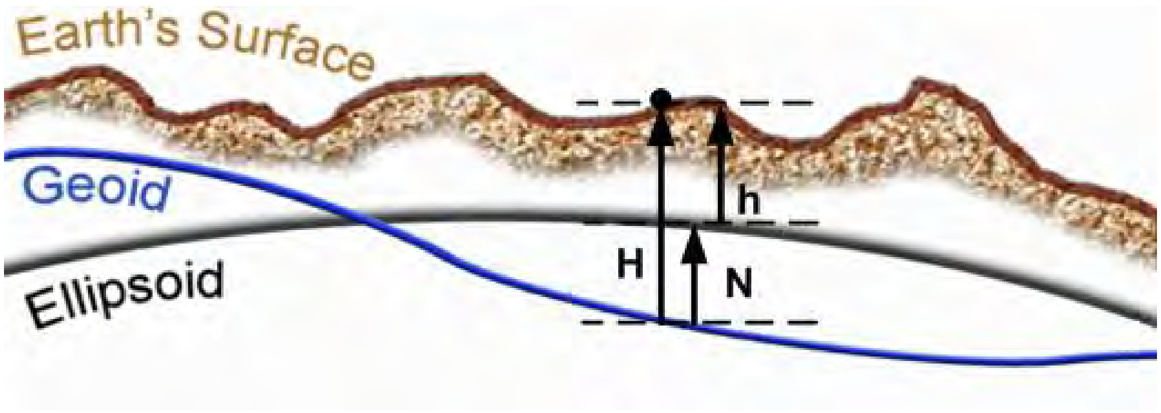


Figure 2.6: Difference between datum provided ellipsoid height *h* and the orthometric geoid height *H* [79].

is the normal radius [71]. Converting from ECEF to geodetic requires a more complicated approximate but closed form method performed iteratively and will not be derived here. For details of that transform consult [71].

2.2.1.2 Geoid Earth Models.

Another type of Earth model is called the geoid. The geoid model captures the general lumpy shape of the Earth using local gravity measurements. In this way the geoid model creates a Mean Sea Level (MSL) surface over the Earth, as if the average sea level flowed up and into the land [71, 110]. Ellipsoid models are generally a best fit to geoid MSL over the ellipsoid's designed region of interest. The relationship of the ellipsoid and geoid models to the earths surface (topography) is shown in Figure 2.6.

When using an ellipsoid datum, the height value *h* represents the distance from the ellipsoid to the surface point. But the distance above MSL is what is actually desired, and that is provided by finding the distance from the geoid to the surface point, the orthometric height *H*. This is accomplished by using the geoid model of the region, defining the difference between geoid and a particular datum ellipsoid heights *N* for point defined by the ellipsoid latitude and longitude [76]. The orthometric height is found simply by adding

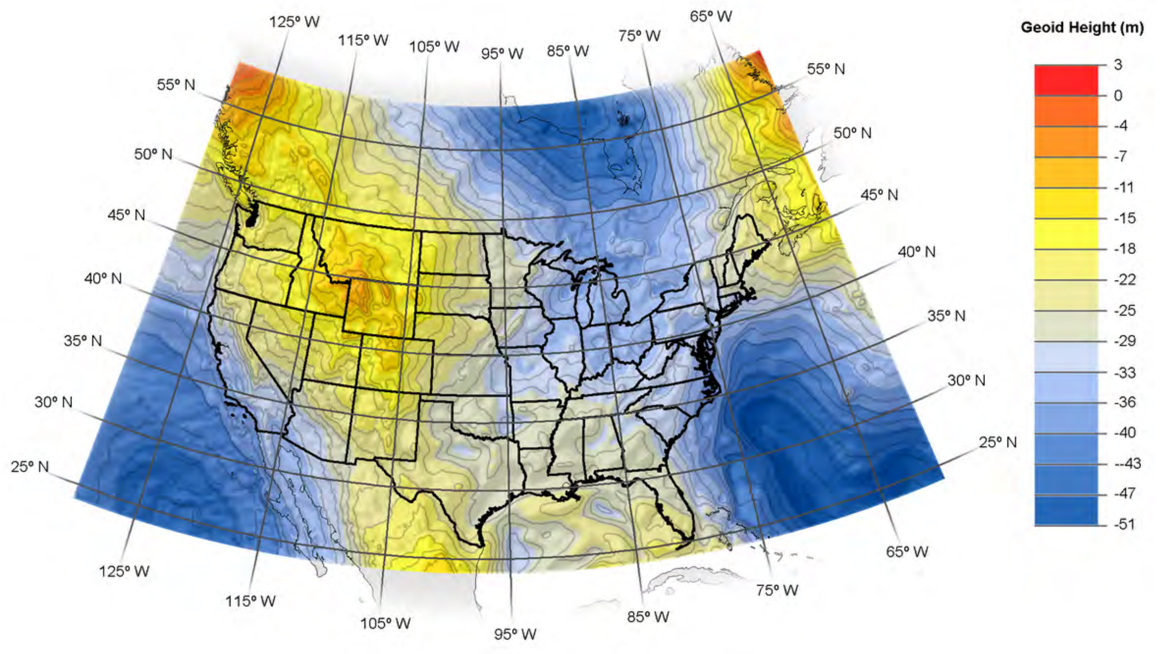


Figure 2.7: The difference in height between the the NAD83 datum ellipsoid and the United States geoid for 2012 [107].

the ellipsoid height to the signed geoid/ellipsoid separation

$$H = h + N. \tag{2.27}$$

A current map of the geoid/ellipsoid separation distances from the NAD83 ellipsoid is seen in Figure 2.7.

2.2.1.3 Datum Differences.

The differences between the datums of interest in Table 2.1 in terms of exact latitude and longitude coordinates can in general be considered small (under a meter in most cases) for applications outside of high precision needs. While tectonic motions cause point drift conditions in the accuracy between datums over time, the tectonic velocities over the passage of time can be accounted for through the Helmert transformation [16, 99–101] (available as free software [98, 108]) and other more complex but direct transforms

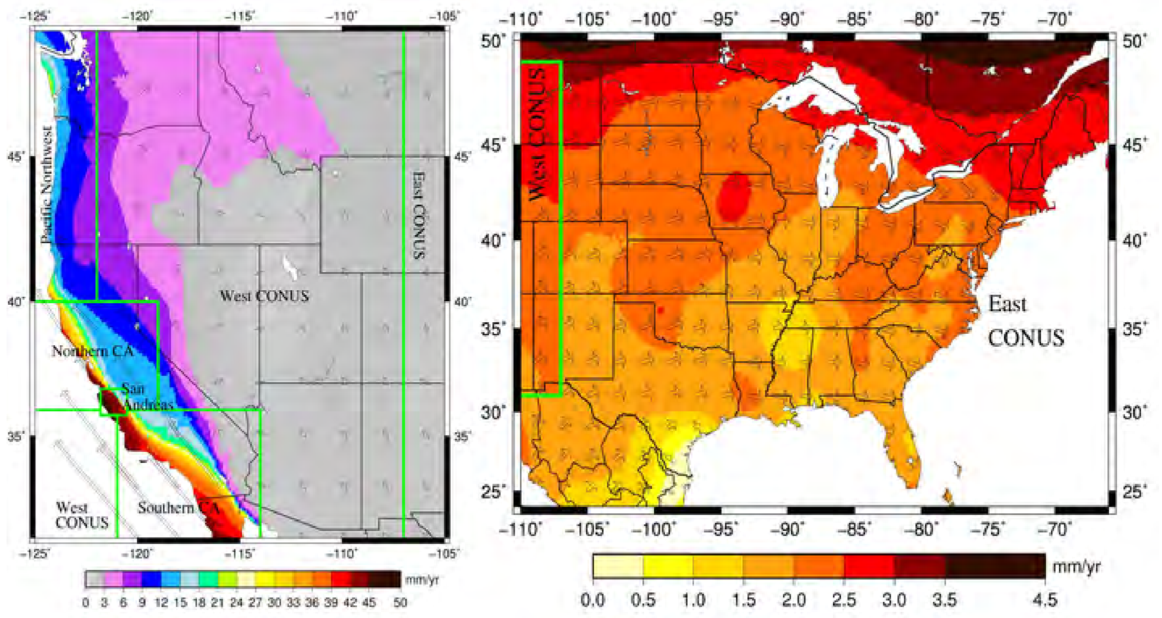


Figure 2.8: Tectonic drift rates (horizontal velocities) for the United States relative to NAD83 in 2011 [108].

[50, 96]. Tectonic velocities vary regionally, but for use in this research over the midwest United States, this rate is in mm per year [98, 100] shown in Figure 2.8. More currently, the organizations that define WGS84(G1674) and ITRS(2008) (NGA and IERS respectively) coordinate their efforts when the datums are updated, this has resulted in the two datums aligned to each other at better than 1cm [69, 116] and through ITRS, transformations between WGS84 and NAD83 are identical with an uncertainty of 2 meters [69, 98].

2.2.2 Moving Between Frames.

With movement between ECEF and datums defined, transformations between the various coordinate frames defined in Section 2.1.3 can be performed to move coordinates between frames.

2.2.2.1 Geodetic to Navigation Frame.

Movement between ECEF and geodetic coordinates was defined in Section 2.2.1.1. Movement from ENU to a geodetic datum frame is a standard transformation requiring a −90 degree rotation λ about the x axis (latitude) followed by the same rotation φ about the z axis (longitude) of the selected origin point for the ENU frame [71, 124]. Using Equation (2.12) and Equation (2.10) the DCM is constructed

$$\mathbf{C}_e^{ENU} = \begin{bmatrix} -\sin\varphi & -\cos\varphi & 0 \\ -\cos\varphi & -\sin\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \sin\lambda & -\cos\lambda \\ 0 & \cos\lambda & \sin\lambda \end{bmatrix} \tag{2.28}$$

$$\mathbf{C}_e^{ENU} = \begin{bmatrix} -\sin\varphi & -\sin\lambda\cos\varphi & \cos\lambda\cos\varphi \\ \cos\varphi & -\sin\lambda\sin\varphi & \cos\lambda\sin\varphi \\ 0 & \cos\lambda & \sin\lambda \end{bmatrix} \tag{2.29}$$

and the DCM to move from NED to that datum's frame can be created from Equation (2.29) combining Equation (2.18) and Equation (2.8). Both the DCM for moving from the datum's frame to ENU or NED frames are formed by taking the inverse or transpose of Equation (2.29) using Equation (2.8).

2.2.2.2 Navigation to Body Frame.

To track a vehicle as it moves through a navigation frame space, changes in the attitude of its body frame axis with respect to the static navigation frame axis must be followed. Attitude changes are represented as the Roll Pitch Yaw (RPY) rotations around the the body frame axis:

$$\text{roll rotation } \phi \text{ about the } X\text{-axis, } \mathbf{C}_r = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix} \tag{2.30}$$

$$\text{pitch rotation } \theta \text{ about the } Y\text{-axis, } \mathbf{C}_p = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \tag{2.31}$$

$$\text{yaw rotation } \psi \text{ about the } Z\text{-axis, } \mathbf{C}_y = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.32}$$

These rotations as applied to the body frame are shown in Figure 2.9.

To rotate from the NED frame into the body frame uses the DCM [71]

$$\mathbf{C}_b^{NED} = \mathbf{C}_r\mathbf{C}_p\mathbf{C}_y \tag{2.33}$$

$$\mathbf{C}_b^{NED} = \begin{bmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \sin\phi\cos\theta \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\phi\cos\theta \end{bmatrix} \tag{2.34}$$

and performing the transformation on a position in the body frame

$$\mathbf{p}^{NED} = \mathbf{C}_b^{NED}\mathbf{p}^b. \tag{2.35}$$

Similarity the DCM to move from the body frame to the ENU frame can be created from Equation (2.34) combining Equation (2.18) and Equation (2.8). The DCM for moving from the NED frames to the body frame is formed by taking the inverse or transpose of Equation (2.34) using Equation (2.8), and similarly for ENU to body frame.

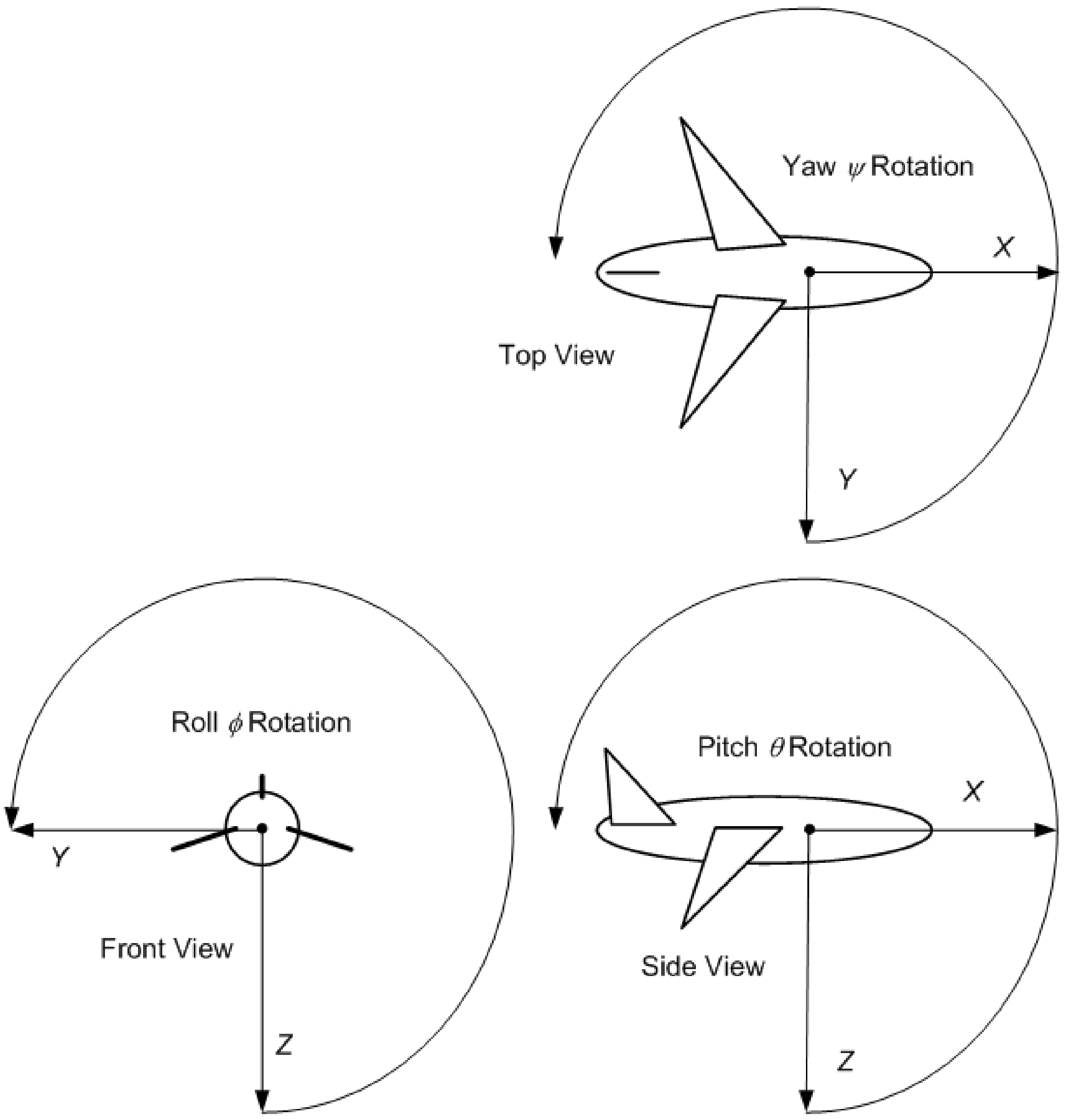


Figure 2.9: Orientation changes via RPY rotation angles on a body frame.

2.2.3 Projections.

Local navigation reference frames are often application specific. A system used by National Geodetic Survey (NGS), National Oceanic and Atmospheric Administration (NOAA) and others with mapping interests in North America is the State Plane Coordinate

System of 1983 (SPCS). SPCS is a map projection system based off of NAD83 [102]. SPCS divides up the United States into multiple zones per state as seen in Figure 2.10. Each zone is mapped depending on orientation by either a Transverse Mercator Projection (TMP) (vertical zones) or Lambert Conformal Conic Projection (LCCP) (horizontal zones). As the curve of the earth passes through the projection plane for a particular zone it creates two parallels where the projection will be exact (scale of the projection equals 1). Between the parallels there will be a central meridian where the the projection scale adjustment is greatest Figure 2.11. Zone size, parallels, scale restrictions, and navigation frame origin are all chosen by the State to preserve a particular level of accuracy, e.g. [109]. These individual State zones are what become the local navigation frame map with a ENU orientation system.

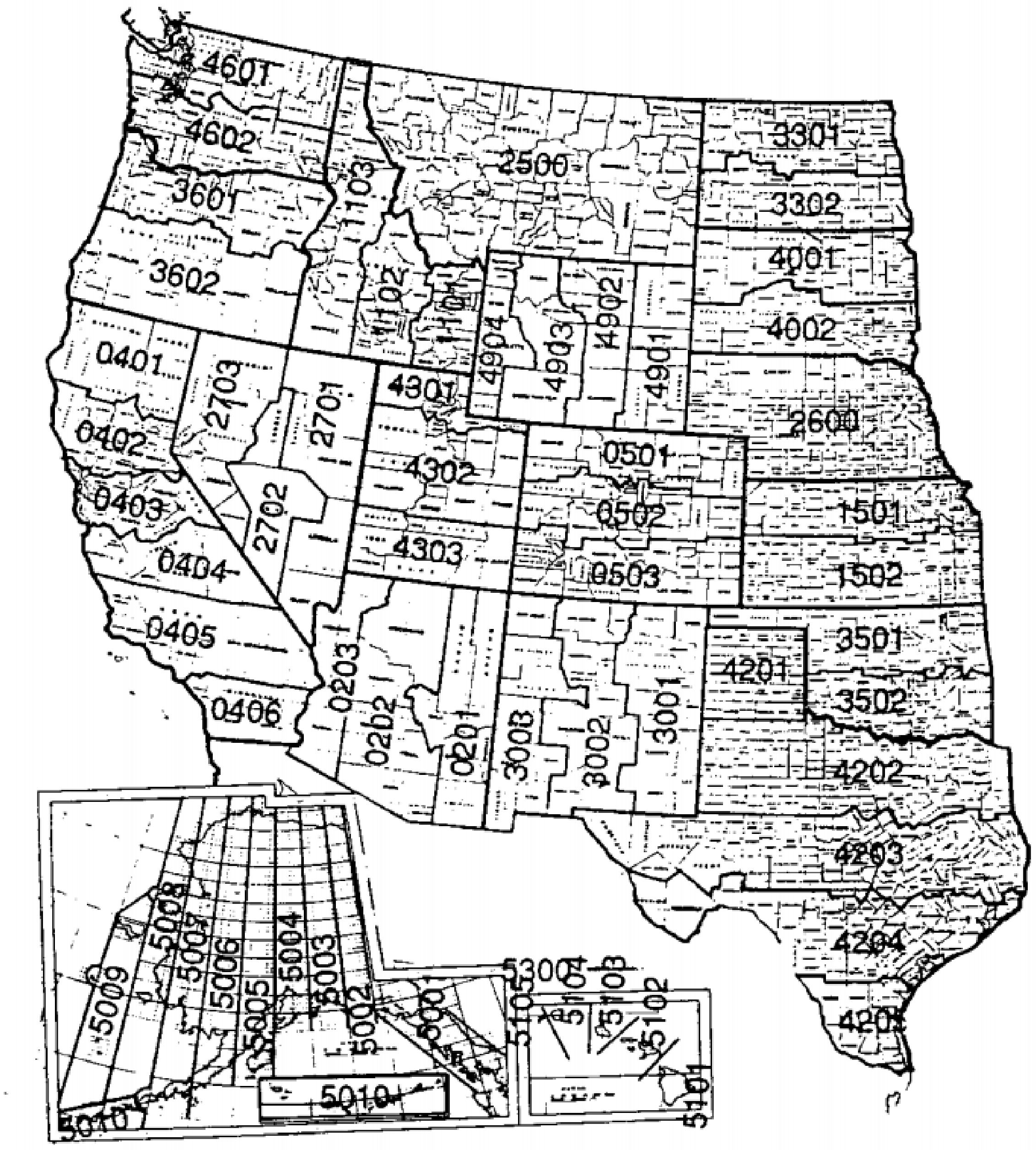


Figure 2.10: Spate Plane Coordinate System Zones for the Western United States [102].

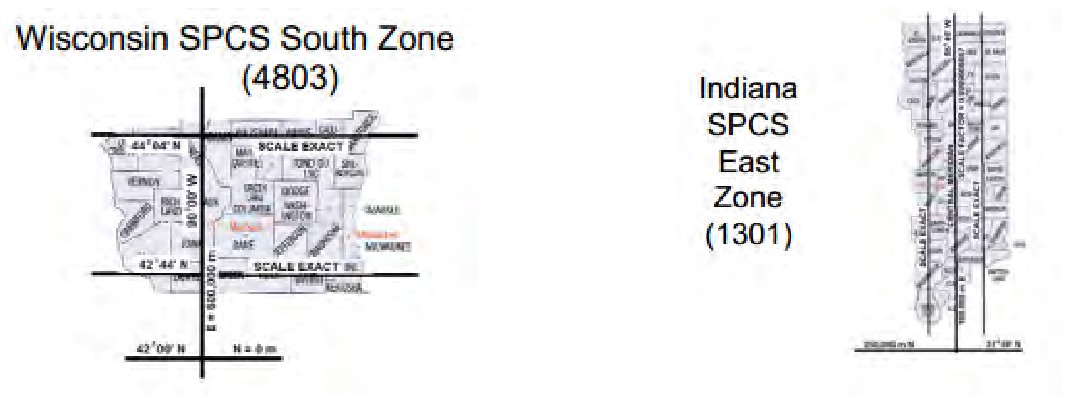


Figure 2.11: Origin and parallels of a example SPCS zone [37], TMP (left) and LCCP (right).

2.3 Computer Vision

Computer vision is strongly associated with image creation and processing. Computer vision based methods employ a wide variety of filters, operators, and morphology functions [23, 70]. Features represent information within an image that are of interest to an application, e.g. detection of continuous road, intersections, and stop lights autonomous car driving using camera images. The interest points (features) for an application come in many forms and combinations, but generally can be lumped into basic low-level features that require no spatial data from the image, and high-level features that focus on shapes and objects within a image [70]. The usefulness of a particular feature is application dependant, but common properties are uniqueness or saliency, and invariance to certain conditions (e.g. illumination levels, distortion and noise). Features extraction can be used to extrapolate information from image data, identifying specific objects or structures that allow classification of data into selected types. Region texture can be defined and segmentation employed to further specify traits of a similar classification. Features and classification data can but used distinctly or added to the original data set, creating higher level data products like reconstructed $3D$ models, or extract an known object from a cluttered background. Feature with sufficient saliency and detectably can be used to characterize objects or locations and used to compute if and by how much two images overlap. SIFT features used in this research and are detailed more in the next sections.

2.3.1 SIFT Features.

A widely used object detection feature in recent years is SIFT. SIFT is of particular interest to this research for its invariance to translation, rotation, and scale changes between images, and further for its more limited robustness to affine, noise, and illumination conditions [55]. The feature algorithm was inspired by human eye behaviour and can be broken up into two primary steps, keypoint detection and descriptor creation. The SIFT algorithm is an image processing technique, computing features from an intensity image (0 to 1 scale pixels, usually shown in black and white), and is frequently paired with a matching algorithm to establish a correspondence between images. There are several direct variants of SIFT available in the literature [8, 46, 52, 54, 64], each demonstrating an improvement of one or more facets of the original algorithm. For simplicity and proof of concept, this research utilizes the original SIFT algorithm as described in the next sections [55].

2.3.1.1 Keypoint Detection.

The first portion of the SIFT algorithm detects the location of the keypoints in the input image. The keypoint is a scale-space extremum (minima and maxima) that are detected, localized, and filtered. To detect the extremum, the input image $\mathbf{I}(x, y)$, where x and y are the image pixel coordinates, is first convolved with a variable scale Gaussian $\mathbf{G}(x, y, \sigma)$, producing the scale space image $\mathbf{L}(x, y, \sigma)$ defined as

$$\mathbf{L}(x, y, \sigma) = \mathbf{G}(x, y, \sigma) * \mathbf{I}(x, y) \tag{2.36}$$

$$\mathbf{G}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{2.37}$$

The doubling of σ notes the change in scale of one octave. To efficiently detect stable keypoints, the Difference of Gaussian (DoG) $\mathbf{D}(x, y, \sigma)$ is calculated from the difference of nearby scales within an octave, designated by a constant k

$$\mathbf{D}(x, y, \sigma) = (\mathbf{G}(x, y, k\sigma) - \mathbf{G}(x, y, \sigma)) * \mathbf{I}(x, y) \tag{2.38}$$

$$= \mathbf{L}(x, y, k\sigma) - \mathbf{L}(x, y, \sigma) \tag{2.39}$$

and shown in Figure 2.12.

The keypoints candidates are finally detected by comparing each pixel to its 26 DoG neighbors, 8 within its own scale and the 9 in the scale above and below it. The pixel is only chosen as a keypoint if it is greatest or least valued of all its neighbors in the DoG stack.

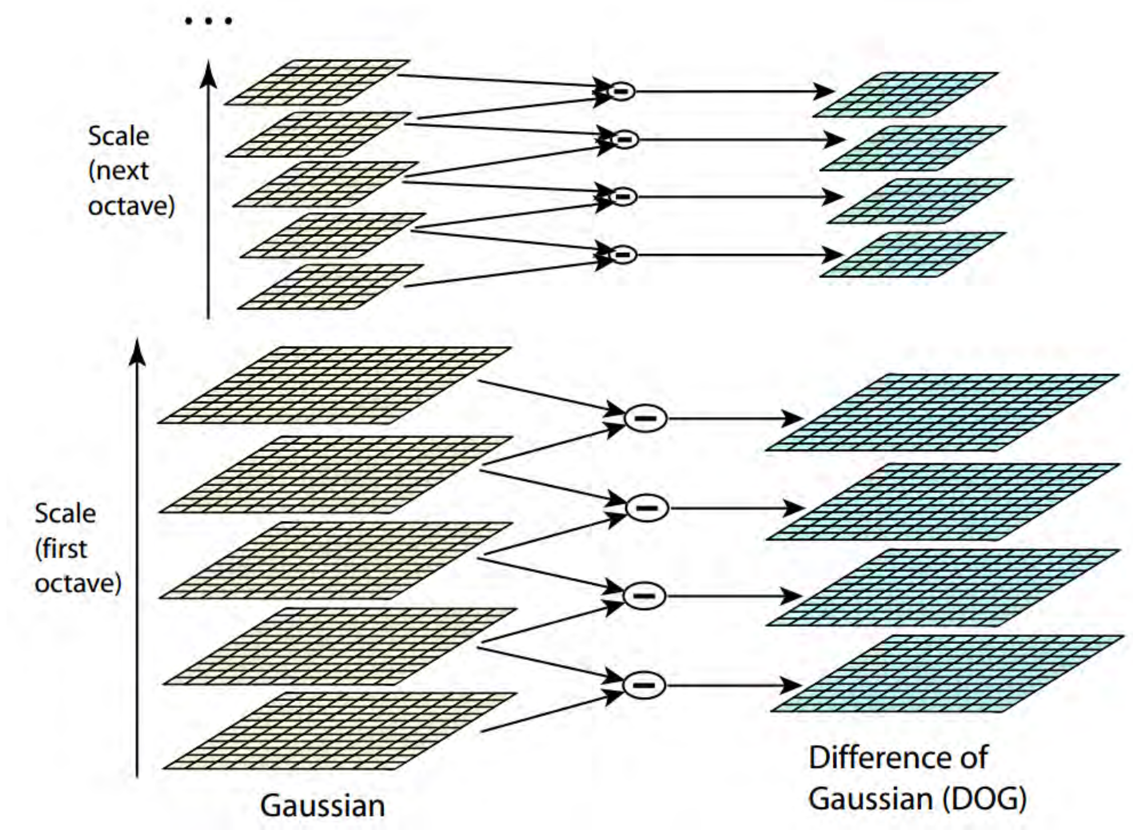


Figure 2.12: Gaussian smoothed images **L** are created for each octave and scales k within each octave, then finally subtracted to create DoG images **D** [55].

The keypoints at this phase are only candidates and are first localized using a $3D$ quadratic interpolation and then filtered. The localization produces a sub-pixel location for the extremum in the image at its scale, and a Taylor series expansion of $\mathbf{D}(x, y, \sigma)$ is used to filter out unstable, low contrast candidates by setting a threshold. In addition, DoG extremum have a strong erroneous response near edges. Edge response extremum are detected with principle curvature computed from a $2x2$ Hessian matrix. The ratio of the largest eigenvalue to the smallest, the principle curves in the Hessian, are then compared against a threshold and the corresponding extremum filtered. The remaining keypoints after contrast and edge response filtering are kept and then assigned a descriptor [55].

2.3.1.2 Descriptor Assignment.

Each keypoint is next given a descriptor representing as relevant to the keypoint local neighborhood orientation. This is done by selecting the smoothed image $\mathbf{L}(x, y, \sigma)$ at the scale the keypoint is in and calculating the gradient magnitude $m(x, y)$ and orientation $\vartheta(x, y)$ using pixel differences:

$$m(x,y) = \sqrt{(\mathbf{L}(x+1,y) - \mathbf{L}(x-1,y))^2 + (\mathbf{L}(x,y+1) - \mathbf{L}(x,y-1))^2} \tag{2.40}$$

$$\vartheta(x,y) = \arctan\left(\frac{\mathbf{L}(x,y+1) - \mathbf{L}(x,y-1)}{\mathbf{L}(x+1,y) - \mathbf{L}(x-1,y)}\right). \tag{2.41}$$

A 36 bin orientation histogram is constructed of the region around the keypoint, each is weighted by its gradient magnitude and a Gaussian weighted circular window with 1.5σ of the keypoint scale. The highest peak in the histogram is interpolated with the 3 closest peaks to establish a keypoint orientation assignment. The descriptor is finally formed from the orientation histogram entries. Then, to establish rotational invariance, the coordinates of the descriptor and orientations are rotated relative to the keypoint orientation. This process is illustrated in Figure 2.13. Best results were achieved with $4x4$ arrays of histograms with 8 orientation bins each, generating 128 element feature vector descriptors for each keypoint [55].

A descriptor assignment is paired with its keypoint and completes the formation of the SIFT features. Each feature in an image is described by its location, scale, and descriptor.

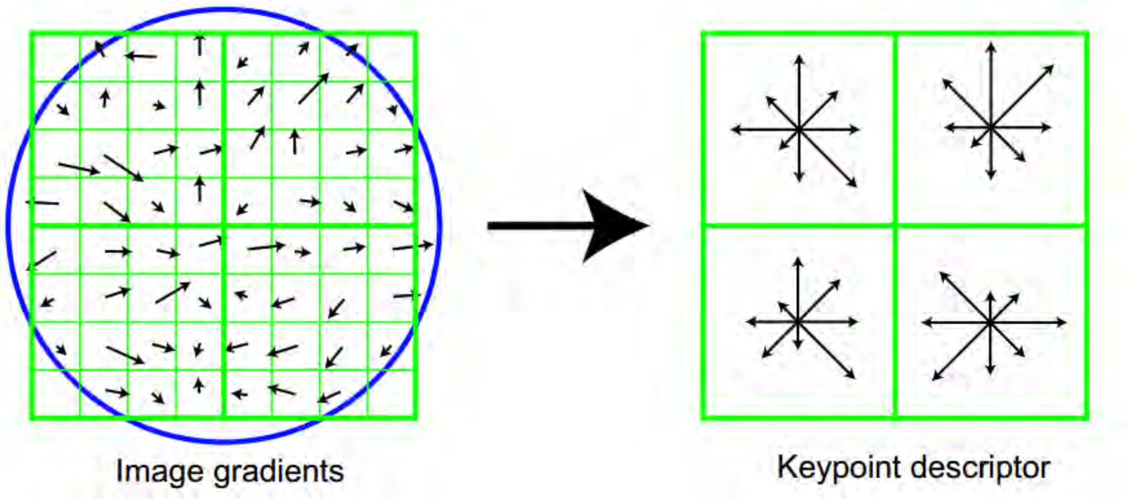


Figure 2.13: In this example a $2x2$ descriptor is calculated from an an $8x8$ sample set. Gradient magnitude and orientations around a keypoint are calculated and weighted by a Gaussian window (circle) on the left. They are then accumulated into histograms (right) summarizing the contents over $4x4$ subregions with the length of the arrow representing the sum of the gradient magnitudes [55].

2.3.2 Matching.

Many applications involving features are paired with a matching algorithm for image registration and object recognition. Image matching algorithms derive a correspondence between two images that contain some portion of the same information. The correspondence is represented by an affine transformation that registers one image or 3D data set to another, performed as a translation, rotation, and sometimes scale, making them approximately spatially coincident [57, 70]. Equation (2.42) shows that two images ($\mathbf{I}_1(a,b)$, $\mathbf{I}_2(x,y)$), located in different but overlapping locations of the same frame, are used to find the translation $\mathbf{t}$ vector and the $3x3$ rotation matrix $\mathbf{R}$ that registers image $\mathbf{I}_2(x,y)$ into the space of the $\mathbf{I}_1(a, b)$ using correspondence algorithm *Match*

$$(\mathbf{R}, \mathbf{t}) = Match\{\mathbf{I}_1(a, b), \mathbf{I}_2(x, y)\} \tag{2.42}$$

$$\mathbf{I}_2(a, b) = \mathbf{R}\mathbf{I}_2(x, y) + \mathbf{t}. \tag{2.43}$$

There are many approaches to solving for the correspondence in computer vision and other fields. The remaining sections describe a few common methods that are utilized specifically with features. For more detail than presented here, a review of several prominent correspondence algorithms was performed in [89].

2.3.2.1 Correlation Matching.

Image correlation or template matching [70] is a high accuracy and computation approach where the target image $\mathbf{I}_{x,y}$ is pixel matched systematically to all locations $\mathbf{W}$ in the template image $\mathbf{T}_{x,y}$. The pixel differences when the template is at a given location (i, j) are summed, and the minimum error e marks that location as the maximum likelihood of the image match

$$\min e = \sum_{(x,y)\in\mathbf{W}} (\mathbf{I}_{x+i,y+j} - \mathbf{T}_{x+i,y+j})^2 \tag{2.44}$$

Template matching can find very accurate translation differences [13], but accuracy suffers if faced with rotational, scale, and illumination changes between images. These can be mitigated by adding search cases for rotations an scales at a higher computation cost. More recently, first extracting features and then using normalized cross correlation can significantly improve both robustness to image differences, and faster computation times [47, 134].

2.3.2.2 ICP Matching.

If the data that needs matching can be represented as a point cloud, Iterative Closest Point (ICP) can be used to find the correspondence [9, 14]. ICP finds the correspondence translation and rotation by calculating the minimum distance between the two clouds' nearest neighbor points and calculating the resulting Mean Least Squares (MLS) error.

If point cloud points **p** are to be matched to point cloud **I** with points **i**, then the minimum distance d between points is

$$d(\mathbf{p}, \mathbf{I}) = \min_{\mathbf{i} \in \mathbf{I}} \|\mathbf{i} - \mathbf{p}\|. \tag{2.45}$$

Once the closest points are found for all points **p**, ICP uses a correspondence algorithm (originally [35]) to solve for the translation and rotation, and then applies it to register the point clouds. Calculating the minimum distance and correspondence transform is performed iteratively until some threshold MLS error is reached. The algorithm assumes all points have some match between the two clouds, and a good estimate of starting location is required because the final correspondence will converge to a local minimum solution. Use of ICP is popular in computer vision and has generated several variants [63, 87, 133] improving robustness and speed of the algorithm. ICP can also benefit from using features instead of full surface representations [94].

2.3.2.3 Outlier Rejection.

The previous matching algorithms all suffer from the presence of outliers in the data. Large or numerous outlier points will influence the generated correspondence, resulting in erroneous registration. This is especially true if the matching is performed using features, where features represent the only data of importance, giving outlier features additional weight. Constraints can be applied to the data sets to detect the presence of outliers.

An algorithm that both detect outliers and solves for correspondence is Random Sample Consensus (RANSAC) [21]. RANSAC assumes a data set is contaminated with outliers, selects a small sample set randomly from the data, and tests it against the consensus set. The consensus set provides an estimation of the correspondence based off of the previous sample sets (using [35] or something similar). The consensus is then updated, a new sample set selected, and the process is repeated iteratively until the probability of the consensus is below a chosen threshold. This process is robust when less than 50% of data are outliers as implemented in [136] and has been demonstrated effective when applied to SIFT features [1, 10, 20, 60, 92, 111, 115].

2.3.2.4 Horn's Correspondence Algorithm.

An algorithm developed by Horn [35] is a popular choice for calculation the correspondence transform required to align one set of points with another. While there have been some further work at advancing the algorithm (e.g;[36, 118]), its basic process remain largely unchanged. A subroutine using Horn's algorithm can often be found in more advanced algorithms or algorithms that implement iterative solutions, and was found present in the versions of ICP and RANSAC utilized for this research. In general, Horn's algorithm calculates the rotation, translation and scale that best aligns two sets of points using least squares. The algorithm requires a minimum of three 3D points known in both data sets, giving nine degrees of freedom, to solve for the seven unknowns of the correspondence. In this research, the option for solving scale was disabled (always equal to 1) to avoid distortion of the LiDAR range-based data and it's unstudied impact on SIFT features.

2.4 Light Detection And Ranging

The LiDAR sensor has been around since the 1960's, but its more modern form used in remote sensing (data collection via energy reflected from Earth) and Simultaneous Localization and Mapping (SLAM) has been steadily advancing in capability since the mid 1990's [93]. With the development of scanning devices, airborne LiDAR scans of the ground have become a major mapping tool and public LiDAR databases are currently available for some regions (e.g; [74]). LiDAR is the primary data source utilized in this research and details of this sensor and its data products are covered in this section.

2.4.1 The LiDAR Sensor.

LiDAR is basically an application of the widely known Radio Detection And Ranging (RADAR). Energy is emitted from a source, reflected off a target, detected, and processed to derive the distance to the target. In the case of LiDAR this energy is light measured in photons and in the more specific case of Laser Detection And Ranging (LaDAR), laser light in particular [83]. In the literature, the terms LiDAR and LaDAR are often used interchangeably, but using LiDAR is both more general and often used in the remote sensing and geodesy disciplines. Therefore LiDAR will be the term used in this research as the exact LiDAR used is application specific and not necessarily restricted to only laser light sources, even if the sources used are lasers.

2.4.2 The Laser Range Equation.

The basic function of a LiDAR sensor is to perform a direct detection of the distance from the LiDAR source to a target. This process can be modeled with the LiDAR range equation [83]

$$P_{Det} = \frac{\tau_o \tau_a D_R \rho_t (dA) P_t}{R^2 \theta_R (\theta_t R)^2} \tag{2.46}$$

assuming that the target and receiver are roughly parallel. Equation (2.46) models the energy in joules collected from the LiDAR detector P_{Det} for a single pulse. The detected energy is a function of several factors relating to the channel the LiDAR passes through (e.g. air or vacuum) on its way to the target, the distance and area of the target, and properties of the physical device itself. τ_o is the transmission properties of the receiver optics as a percentage. The losses due to transmission through an atmosphere are accounted for with a percentage τ_a. The diameter of the receiver aperture is D_R in meters. ρ_t is the general reflectiveness of the target as a percentage. The power of the transmitter is P_t in joules. The area of the target itself dA in meters is dependant on distance and the size of the target compared to the beam and optics size. In this research the ground itself is the target as in remote sensing applications, so the beam itself is always smaller than the target and can be modeled as

$$dA = \frac{\pi R^2 \theta_t^2}{4}. \tag{2.47}$$

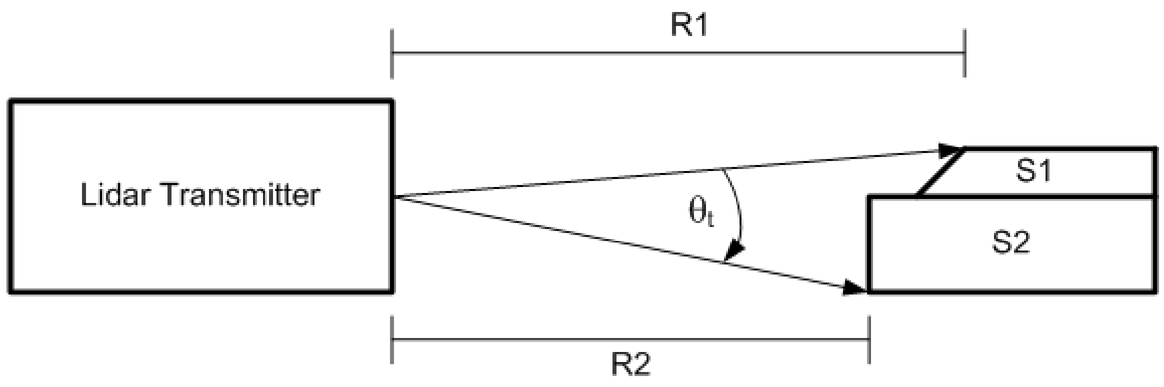


Figure 2.14: A signal pulse sent from a LiDAR spreads out and interacts with a target with two distinct surfaces [83].

The true range to the target is noted as R. The spreading of the beam, beam divergence, is given as θ_t in radians. The reflection angle of the target is θ_R, which is dependant of what the target exactly is, but for general calculations the targets surface can be assumed Lambertian (a diffuse target), with a value of π [83].

The beams interaction with the target is a convolution of the beam pulse wave form and the profile of the target. Then using the amount of power of the returned light and the known speed of light, a signal profile can be determined. If the pulse is assumed to be an impulse, and then for a simple two surface target shown in Figure 2.14, the return signal profile may resemble Figure 2.15.

Figure 2.14 shows two surfaces under the beam spot as it spread out from the beam divergence θ_t. One surface $S2$ is closer at distance $R2$ and parallel to the LiDAR pulse, while the other surface $S1$ is angled and slightly farther away $R1$. Both surfaces are detected from a single LiDAR pulse. The pulse is only recorded within an reasonable length of time called the gating window. Inside this window the LiDAR system will record this specific pulse return, layering of widows allows multiple pulses to be sent and recorded. A peak detection algorithm is used to create a threshold for peaks in a noisy background.

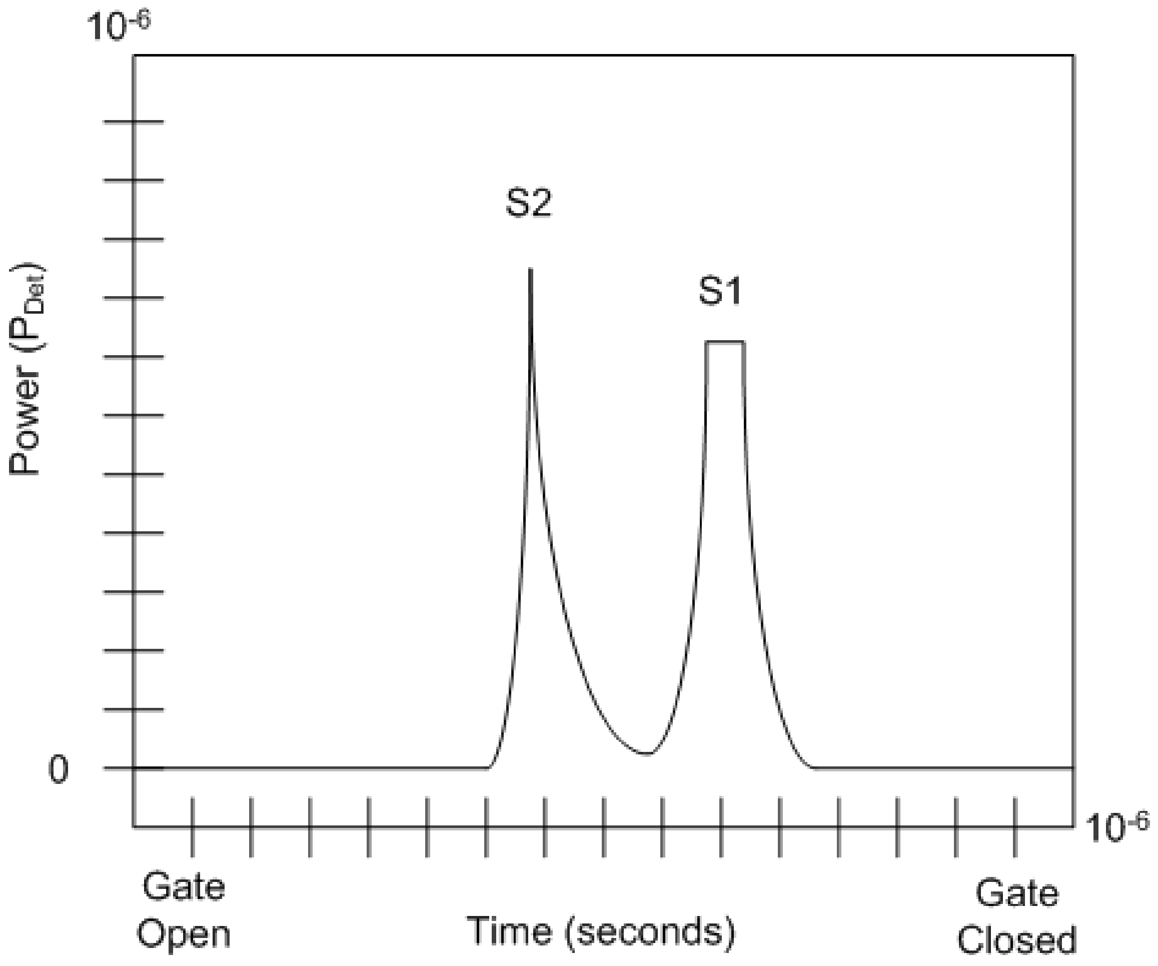


Figure 2.15: The return signal profile from the example in Figure 2.14 [83].

The returned pulse recorded in the gating window is shown in Figure 2.15. The closer surface $S2$ is detected first in time, and its flat profile produces close to a impulse return peak. The second $S1$ peak is more complex; its surface is physically smaller resulting in a smaller peak than $S2$ and it is at an angle flattening the return peak energy. By using the speed of light constant c in meters per second, the range R in meters can be derived from the time t in seconds it took the beam to travel to the target and back. This follows

$$R = \frac{tc}{2} \tag{2.48}$$

$$c = 299792458 \tag{2.49}$$

thereby giving the detected surface ranges, located at their respective power peaks. The time between peaks can be used to measure the height difference of the target surfaces $R2 - R1$. When targeting the ground, pulse returns can be significantly more complex with many peaks from a variety of surfaces.

2.4.3 Sensor Types.

LiDAR sensors come in two types of devices, scanning and flash LiDAR [83]. Scanning LiDAR fire single beams at a typically high rate, using mirrors to move successive beams around in a scanning pattern. The type of pattern is dependent on the particular scanning device used, and has impacts on the resulting sample data. The three primary types of scans are oscillating mirrors that produce a zig-zag scan pattern, two sided cam driven or spinning polygonal mirrors that create a left-to-right sweeping pattern, and the Palmer scanner that rotates an unlevel mirror creating a circular sweep pattern [93]. Other forms of scanners are variations of static or sweeping LiDAR mounted on a spinning axis, creating 360 degree sweeps around a vehicle.

The second type of device is the flash LiDAR. Flash or $3D$ imaging LiDAR are very similar to the device used in a scanning LiDAR but a single larger area pulse is fired and the signal return is processed on a detector array instead of a single detector. The results are area range profiles similar to stereo photogrammetry, and the timing complexities of collecting and registering scanning device data are significantly reduced compared to scanning LiDAR. Providing uniform illumination through the single pulse on a complex surface can be challenging [83].

2.4.4 Common Data Products.

LiDAR signal returns are typically not useful in their raw form. The raw signal returns are normally collected as a coherent group and processed into one of several more universal data products. These products are then available for further application specific use. The most common two are referred to as point clouds and Digital Elevation Model (DEM). Point clouds are a formalized data type, where raw data samples are collected and rendered into a coordinate space. They are used to represent surfaces and, depending on density, can show very high detail. Raw LiDAR range data collected over time is a point cloud registered in the LiDAR sensor frame. Further processing can register the point cloud into a navigation frame and be viewed like a map. There are a great many algorithms available to work with point cloud data and some are available for use online [88]. In addition, point clouds and images can be processed into significant points, points of interest, or features, creating a feature point cloud. Such a reduced data set contains only the information of interest, and can speed up further processing.

The second common data product is a DEM. DEM are generated from point cloud data, projecting the elevation value of the $3D$ points into a $2D$ plane. Often referred to $2.5D$ images or range images, Digital Surface Model (DSM), and Digital Terrain Model (DTM), a DEM allows a compression of what can be millions of individual data points into a single image file, and also opens up further processing using well known $2D$ image processing methods [23, 70]. The raw LiDAR data is scattered and somewhat random in organization by its nature, and as a result the projection is performed by creating a raster image. Raster images are formed when irregularly distributed data is projected into a grid, a process called rasterization [93]. Multiple data points in a grid cell or an empty cell are averaged, and the grid is typically sized to avoid leaving blank areas and minimize interpolation. An example of rasterization is shown in Figure 2.16. Raster images are

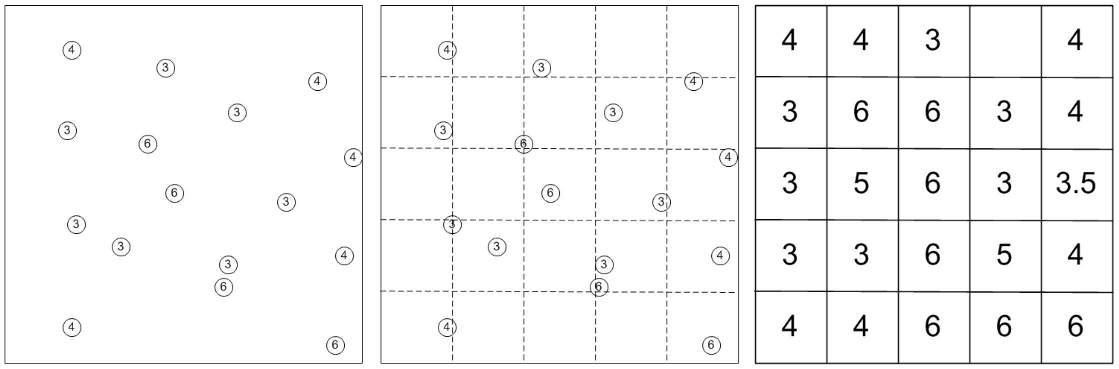


Figure 2.16: A top down view of a point cloud, the individual points show elevation values (left). The raster image is formed (right) by populating a grid (center) with the average elevation values. The resulting matrix of elevations forms the pixel values of the DEM using a direct projection, other interpolation methods may also be used.

popular in that the end product is an image with each pixel representing a cell sized area of the original point cloud sampled region.

Another popular surfacing method with Geographic Information Systems (GIS) software users are Triangular Irregular Network (TIN). TIN are formed by making an interpolation between the most likely nearest neighbor points using Delaunay triangulation, over the entire point cloud [4]. Filling in the TIN generated surface becomes a full $3D$ interpolation of the scanned surface as seen in Figure 2.17. Finally, the most recognisable use of point cloud data is simple contour line generation. A $2D$ image or $3D$ surface is generated by linking points together that share equal or near equal elevation data, creating concentric rings and loops as elevation changes. This has some advantage in that less interpolation is used and it is easily understood by users of topography and cartography shown in Figure 2.18.

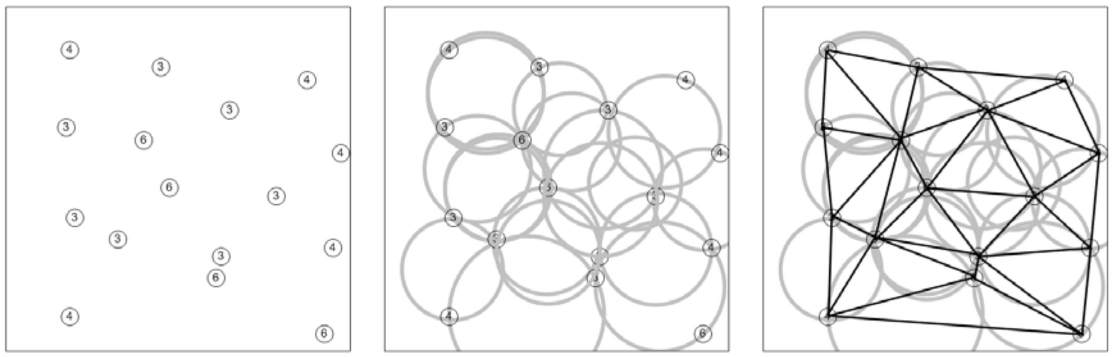


Figure 2.17: A top down view of a point cloud, the individual points show elevation values (left). The TIN surface is formed (right) by applying Delaunay triangulation (center) to the sample points. The resulting surface is and linear interpolation of the linking heights.

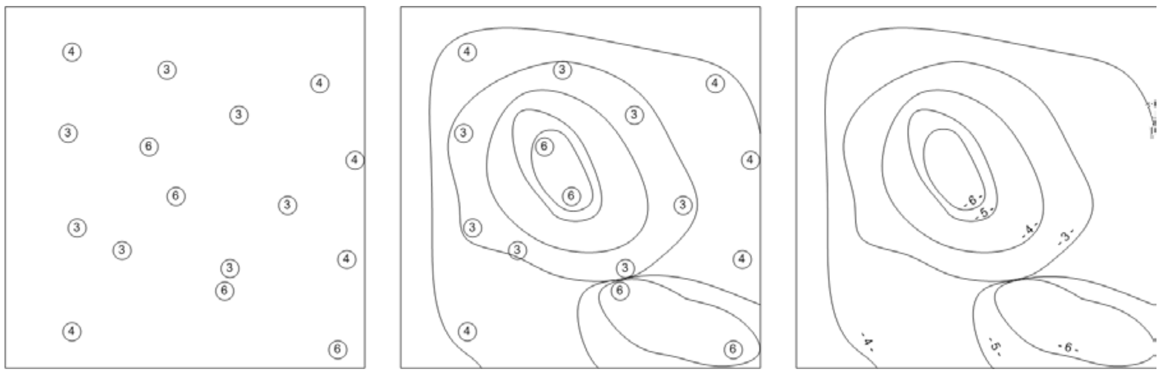


Figure 2.18: A top down view of a point cloud, the individual points show elevation values (left). The contour lines are formed (right) by linking groups of similar value (center). The resulting rings interpolate the flow from peaks to valleys.

Point cloud points can also be encoded with data other than coordinates during the sampling. The first detected LiDAR return only records the highest point sampled from a pulse. In practice, LiDAR frequently penetrates vegetation so that imaging the ground often produces multiple returns that can provide a great deal of data about ground cover, as seen in Figure 2.19. The last return can be filtered for a more sparse but accurate bare earth model [93, 95]. Recording all of the returns and their number significantly enhances the data available for analysis of the sample region, but at the expense of more data storage. In addition to range data, recording the power level of a signal return can be recorded and, once rasterized, creates an intensity image with similar properties to photography [2, 113, 115]. Another much more storage intensive procedure is to simply record the entire waveform of the returned pulse. An active area of research, full waveform LiDAR is showing promise for advanced classification methods on vegetation, soil, and water content. In addition to containing the return and intensity data, the full waveform data makes more rigorous noise analysis possible [49, 66, 93, 119].

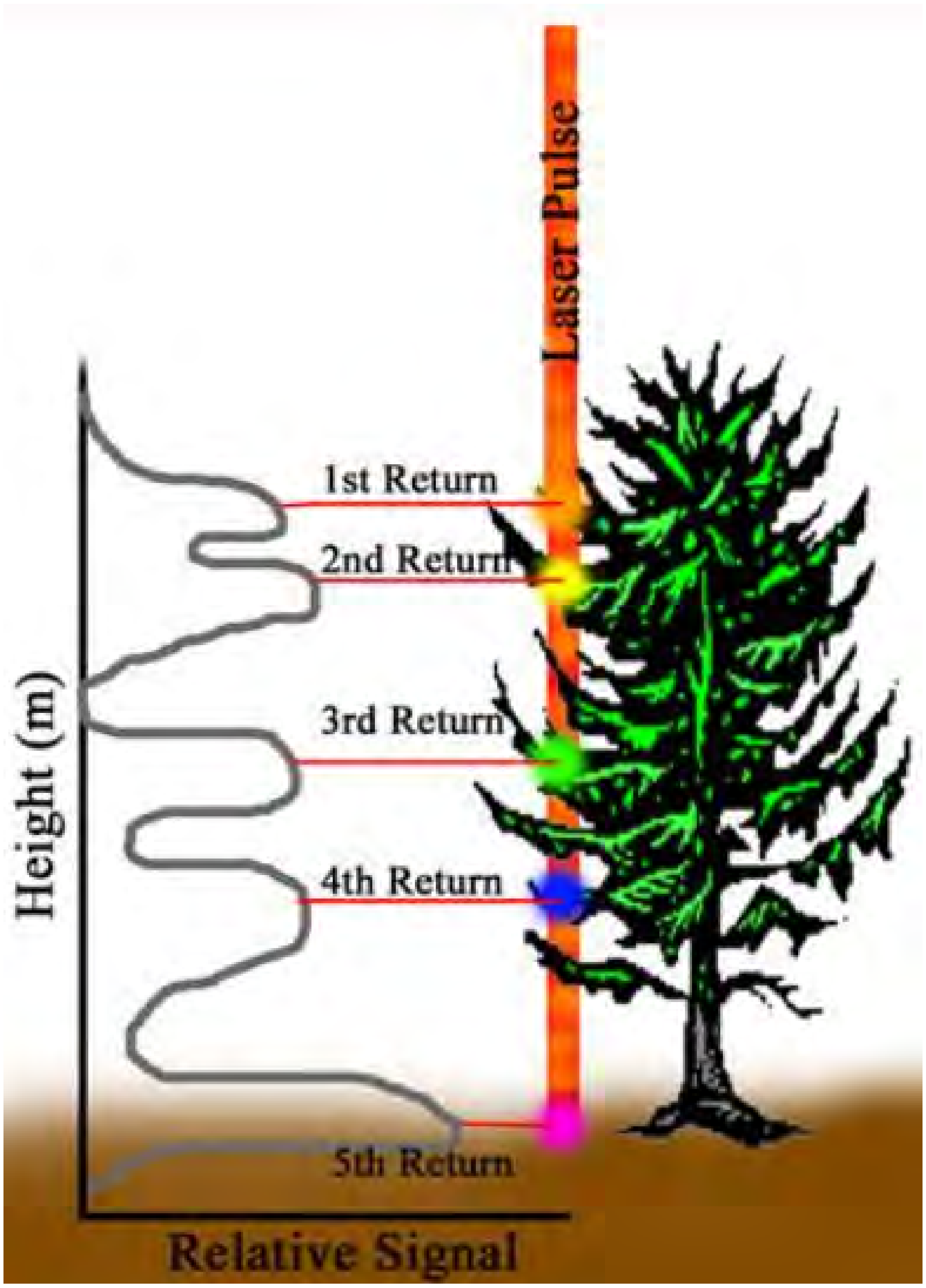


Figure 2.19: An example of a multi-return LiDAR signal interacting with ground vegetation [67].

2.4.5 Error Sources.

There are several systematic sources of error inherent to LiDAR systems [29, 93, 127]. Some are derived from the physics of the sensor, while others are inherited from supporting systems like GNSS or the scanning device. All of these errors are represented in the registered point cloud.

- **Bore-sight Offset Bias:** Bore-sight offset bias results from a small error in measuring the true offset distance between the LiDAR sensor frame axis to the body frame axis which is coincident with an on board Inertial Measurement Unit (IMU). This creates small registration errors that are independent of range distance and scan angle (the fire angle of the LiDAR), but dependent on heading direction in the navigation frame. Bore-sight errors can be corrected with calibration methods [93].

- **Bore-sight Angular Bias:** Bore-sight angular bias results from a small misalignment of the LiDAR sensor frame axis to the body frame axis which is coincident with an on board IMU. This creates registration errors that increase with range distance and change with heading direction in the navigation frame. It also changes with scan angle in other than cross heading direction. Bore-sight errors can be corrected with calibration methods [93].

- **GNSS Noise:** Noise in the GNSS information used in registration will impart similar noise in the derived point cloud and will be independent of the range distance, heading and scan angle [93].

- **Angular Noise:** Noise from IMU or scanning mirror angles will affect the horizontal coordinate components more than vertical and is dependant on range distance and scan angle[93].

- **Range Noise:** Range noise is primarily seen in the vertical coordinate component and is dependant on scan angle. This noise in the detected range is derived from several aspects of the beam or laser used and are outlined below. Beam integral noise sources all involve detection of unwanted photons, and therefor impact the power distribution of the return signal [83].

- **Detector Thermal Noise:** The electronics in the device create extra photons by virtue of being above 0° K in temperature. Measured as dark current in the detector when no light is present, this noise is modeled as a Poisson random variable and can be mitigated by operating the system in Geiger-mode at the cost of slower fire rates.
- **Laser Speckle Noise:** Speckle noise are extra photons resulting from the beam interaction with the reflected surface. Speckle noise is proportional to beam power, and inversely proportional to beam coherency.
- **Photon Counting Noise:** Photons returning to the detector are proportional to what is sent out and arrive at random times creating an uncertainty in the measurement. The number of photons measured in the gating window are modeled as a Poisson random variable.
- **Detector Background Noise:** Background noise are detected photons that are not part of the beam signal. In practice this is reflected sunlight within the detectors sensitive frequencies. Background photons are modeled as a Poisson random variable.

Noise and bias sources are usually hidden from the user, the whole senor system essentially a black box. Effects of a un-calibrated LiDAR are then only visible to the end user in the resulting point cloud, and calibration methods to detect and correct for these errors must be through various quality control methods [93]. If the raw LiDAR measurements are available (INS data, range and scan angle) bore-sight correction is possible [29, 97] to improve accuracy.

2.5 Topographic LiDAR Registration

Remote sensing of the Earths surface topography using a airborne LiDAR system creates a point cloud representing an irregular sampling of the ground. Registration of the point cloud points to a coordinate frame utilizes the coordinate transform equations discussed in Section 2.1.3 and Section 2.2 and is further detailed in [93]. To start with a simple example, assume that a calibrated scanning LiDAR system is attached to a aircraft and pointed at the ground. Also assume that the INS, GNSS and LiDAR are all aligned and in the same reference frame. The aircraft flies over the area to be scanned, creating a swath of sampled ground points. This simple case using an ECEF scenario is illustrated in Figure 2.20.

For a single sample point, the LiDAR system fires a beam and records a range to a ground point. The ECEF position vector $\mathbf{r}^e$ represents the slant range from the aircraft to the ground point. The GNSS system records a position estimate at the time the LiDAR beam fires, this position is the vector $\mathbf{a}^e$. With the position of the aircraft and distance from aircraft to the ground point in the same frame, the resulting position $\mathbf{g}^e$ can be found with

$$\mathbf{g}^e = \mathbf{a}^e + \mathbf{r}^e. \tag{2.50}$$

In a more realistic case, the Equation (2.50) is significantly expanded upon. First, the range information produced from the LiDAR originates in the sensor frame of the LiDAR, and must be transformed into a common frame before it can be combined with the INS and GNSS sensor data. Second, the sensors generally can't occupy the same space on the aircraft and therefore their axes are not collocated as assumed. Removing the assumptions that lead to Equation (2.50) results in the component vectors residing in different coordinate frames. An usable equation can be made using coordinate transforms to move $\mathbf{a}^{WGS}$ and $\mathbf{r}^s$ into the same frame. First the starting states of the three sensors are described, illustrated in Figure 2.21.

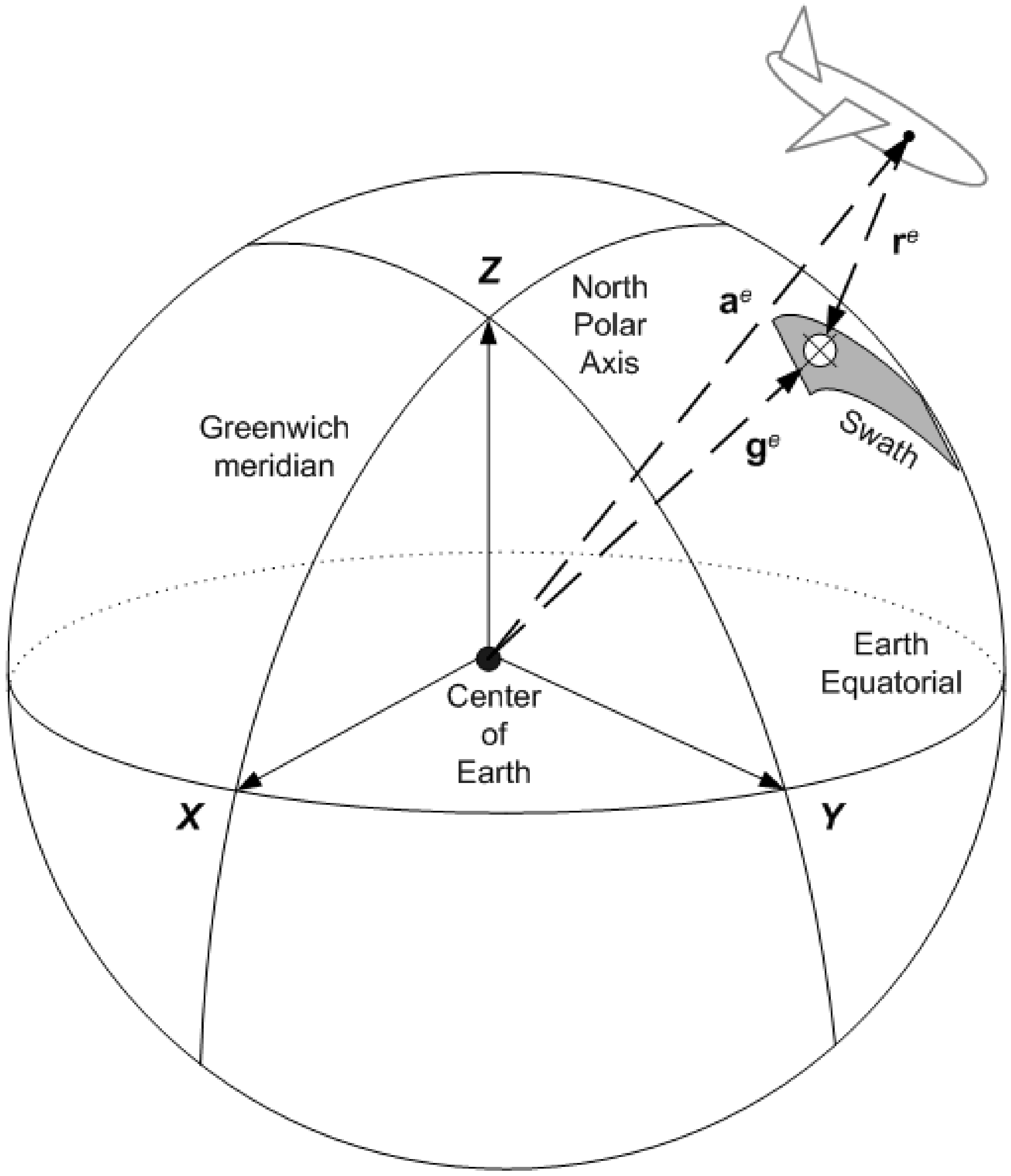


Figure 2.20: The basic case for registration of a LiDAR sample [93].

The LiDAR sensor records each range R sample with its fire angle θ at GNSS time in the LiDAR sensor frame s. To be consistent with the other vectors used in this method, the

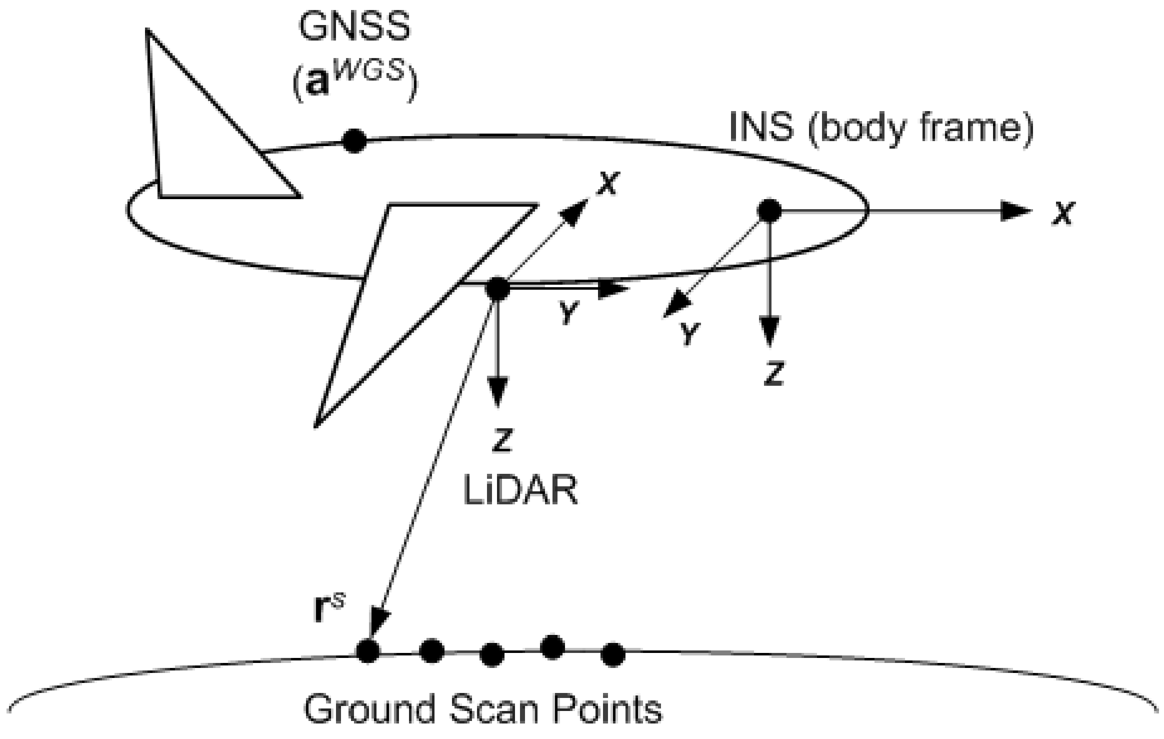


Figure 2.21: The raw states of the three sensors used in topographic LiDAR scanning.

LiDAR data is converted to cartesian coordinates shown as

$$\mathbf{r}^s = [R, \theta] \text{ where,} \tag{2.51}$$

$$[R\cos(\theta), 0, R\sin(\theta)] = [X_r, Y_r, Z_r]\,. \tag{2.52}$$

The axis of the LiDAR sensor frame are as described in Section 2.1.3.4 with X_r representing cross track scanning. The aircraft origin is represented in the body frame as described in Section 2.1.3.3 at the location of the INS. The GNSS receiver is also located in a separate place on the aircraft, generating timing and location in WGS84 coordinates. The axis of all three sensors are neither collocated nor aligned. First the LiDAR sensor frame is rotated into the body frame using a DCM. The rotation needed to align the senor with the body frame is only a 90 degree rotation about the Z_r axis, as seen in Figure 2.21, creating the simple DCM

$$\mathbf{C}_s^b = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.53}$$

and applying the it to the LiDAR vector gives

$$\mathbf{r}^{b'} = \mathbf{C}_s^b \mathbf{r}^s. \tag{2.54}$$

The origin of the LiDAR is usually centered on the reflection point of the scanning mirror [93] and doesn't coincide with the INS origin in the body frame. The origin can be shifted to align with the INS origin through the use of a lever arm. A lever arm is a distance vector, typically hand measured, from one sensor to another. Once in the body frame, the addition of the lever arm $\mathbf{l}_{LiDAR}^b$ moves the origin of the LiDAR to the origin of the INS. Including the lever arm in Equation (2.54) gives

$$\mathbf{r}^b = \mathbf{C}_s^b \mathbf{r}^s + \mathbf{l}_{LiDAR}^b. \tag{2.55}$$

Similarity the GNSS records the position of the aircraft in, typically, WGS84 LLh coordinates. More specifically, the GNSS records the position of the phase center of the receiver antenna, and will need to be shifted to collocate with the INS origin in the body frame using another lever arm $\mathbf{l}_{GNSS}^b$. But the GNSS position data is not in the body frame and will need to be converted further before it can be applied. The first step of this process is to use Equation (2.25) to convert $\mathbf{a}^{WGS}$ geodetic coordinates into $\mathbf{a}^e$ ECEF coordinates. Applying this and Equation (2.54) to Figure 2.21 gives Figure 2.22.

The second step is to convert the LiDAR data to a navigation frame. Since the end result is registering the ground points, an ENU frame is used as described in Section 2.2.2.2. Creating the DCM requires the RPY rotation angle information provided by the INS at the same time as the LiDAR pulse. Following Equation (2.34) makes a body to NED frame,

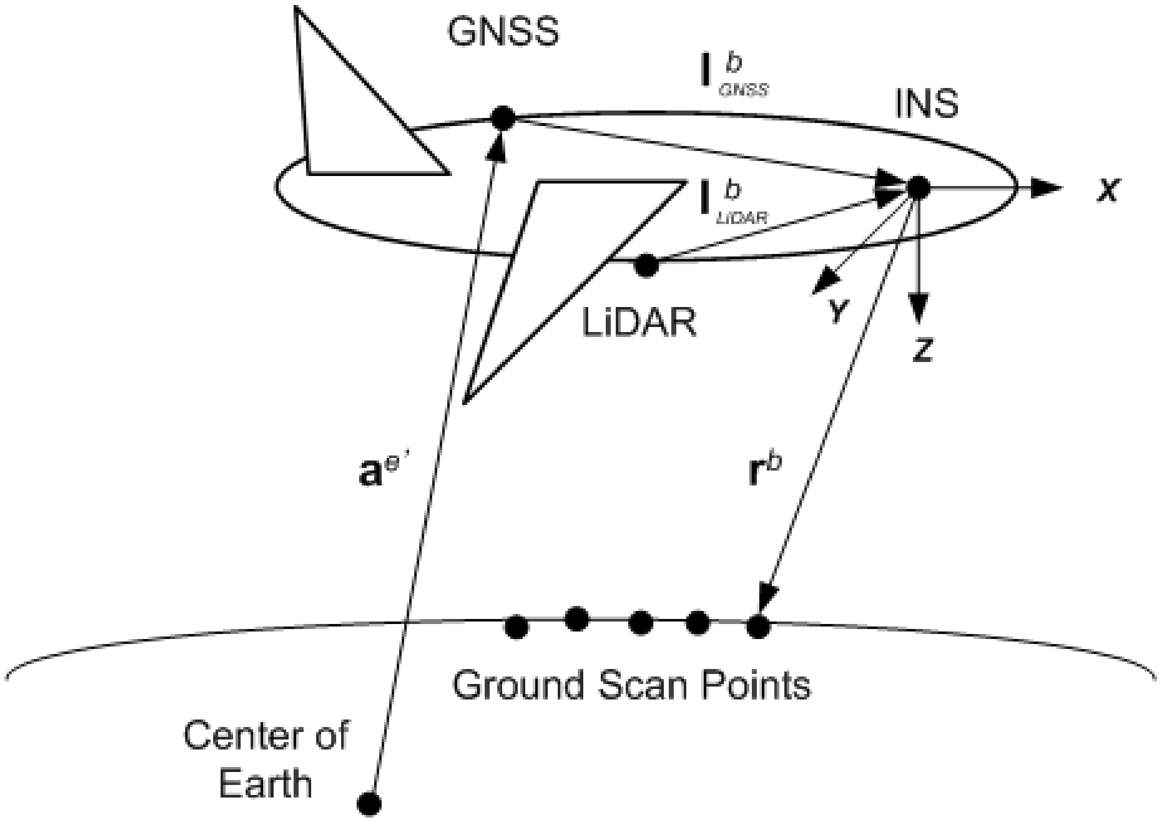


Figure 2.22: Transforming the LiDAR sensor frame data to align with the INS body frame, the lever arm to align the GNSS data is still not in the correct frame to apply.

but combining it with Equation (2.18) provides the the desired ENU frame DCM applied as

$$\mathbf{C}_b^{ENU} = \mathbf{C}_{NED}^{ENU}\mathbf{C}_b^{NED} \tag{2.56}$$

$$\mathbf{r}^{ENU} = \mathbf{C}_b^{ENU}\mathbf{r}^b. \tag{2.57}$$

This DCM must be uniquely calculated for each LiDAR sample time to account for the continued movements of the vehicle. Similarity, the GNSS lever arm can be rotated into navigation frame coordinates

$$\mathbf{l}_{GNSS}^{ENU} = \mathbf{C}_b^{ENU}\mathbf{l}_{GNSS}^b. \tag{2.58}$$

This new arrangement is illustrated in Figure 2.23.

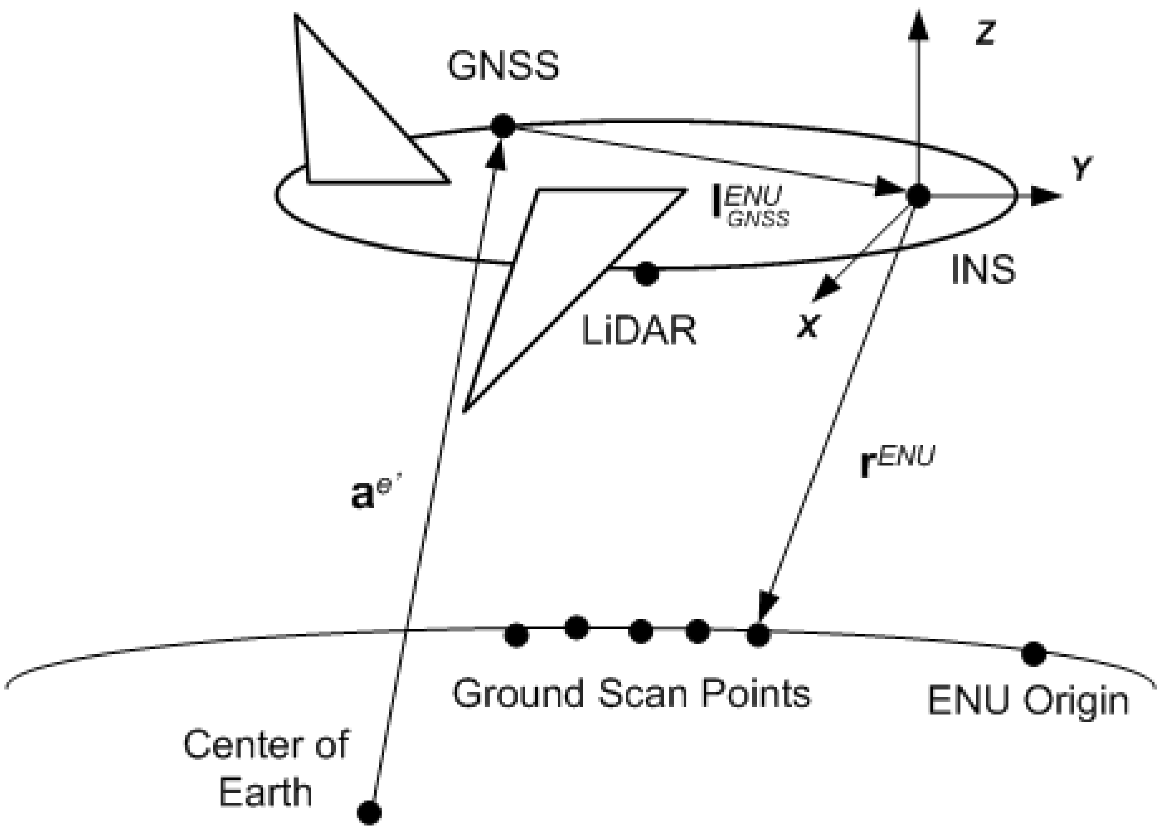


Figure 2.23: Applying the INS RPY angles to apply a body to ENU frame rotation to both the LiDAR data and the remaining GNSS lever arm.

The third step in this process is to rotate the ENU frame coordinates into the ECEF. This is accomplished by using a transpose of the DCM provided by Equation (2.29). An origin point for the ENU frame must be chosen, its coordinates in LLh are utilized for the angles required in Equation (2.29). Again, this transform is also applied to the GNSS lever arm and the process is applied as

$$\mathbf{C}^{e}_{ENU} = \left(\mathbf{C}^{ENU}_{e}\right)^{T} \tag{2.59}$$

$$\mathbf{r}^{e} = \mathbf{C}^{e}_{ENU}\mathbf{r}^{ENU} \tag{2.60}$$

$$\mathbf{a}^{e} = \mathbf{a}^{e'} + \mathbf{C}^{e}_{ENU}\mathbf{l}^{ENU}_{GNSS}. \tag{2.61}$$

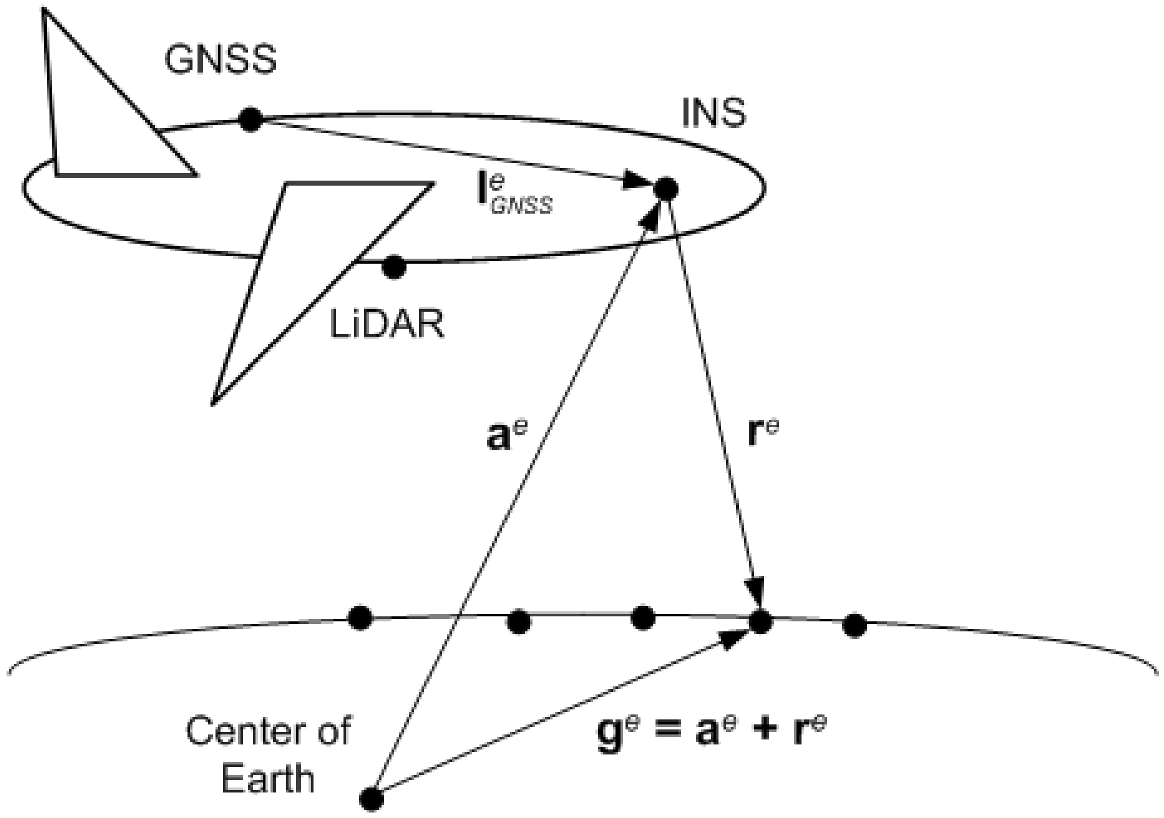


Figure 2.24: Conversion to ECEF frame and applying the GNSS lever arm allows the vector addition of the LiDAR and GNSS data creating the final registration of the LiDAR ground points $\mathbf{g}^e$ in the ECEF frame.

Finally the process of transforming sensor data into the form of Equation (2.50) is complete, with all vectors in the same frame. Figure 2.24 shows this final step relating the process to what is shown in Figure 2.20.

The fully expanded form of Equation (2.50) is

$$\mathbf{g}^e = \mathbf{a}^{e'} + \mathbf{C}^e_{ENU}\mathbf{C}^{ENU}_b\mathbf{l}^b_{GNSS} + \mathbf{C}^e_{ENU}\mathbf{C}^{ENU}_b\left(\mathbf{C}^b_s\mathbf{r}^s + \mathbf{l}^b_{LiDAR}\right) \tag{2.62}$$

or simplified as

$$\mathbf{g}^e = \mathbf{a}^{e'} + \mathbf{C}^e_{ENU}\mathbf{C}^{ENU}_b\left(\mathbf{C}^b_s\mathbf{r}^s + \mathbf{l}^b_{LiDAR} + \mathbf{l}^b_{GNSS}\right). \tag{2.63}$$

Repeating this process on each LiDAR point in a scanned swath creates a point cloud registered into the ECEF frame.

2.6 Terrain Referenced Navigation

Modern navigation in general is possible through use of an INS. The INS is a dead reckoning device that can be used to estimate current position based on past trajectory and its internal sensors. The INS sensor suite chiefly consists accelerometers that measure the specific force as the vehicle moves, and gyroscopes that measure angular rate as the vehicle changes orientation. Both sensor data types are further processed into orientation and trajectory estimates over time [11, 110]. The sensors are self contained and accept no outside influence or data to create this estimate, measuring only the ongoing motions of the aircraft. While position estimates from the INS are frequently sufficient, the sensor themselves are prone to several well studied internal cumulative error sources [11, 110], with the primary effect causing the calculated trajectory to drift over time. This drift error can be mitigated by using higher quality INS at a cost and rigorous calibration techniques, but can never be fully eliminated.

A secondary source of trajectory estimates may be provided to allow an estimation of the drift errors accumulating in the INS, and regular corrections can be applied to maintain its accuracy. There are a wide variety of senors available to generate additional trajectory estimates, but by far the most prevalent modern sensor used is GNSS [106]. Through the use of satellite networks with very high accuracy timing, a GNSS receiver provides a position estimate with very high accuracies. Because of its accuracy, GNSS has become the default external sensor for providing updates to correct drift error, although GNSS is not as robust to interference as the INS [125]. To do this, the combination of the INS, GNSS, or other trajectory estimation source are frequently combined by a Kalman filter. Kalman filters and their variants [44, 59, 90, 126] efficiently combined information sources in a recursive, statistically optimal way, improving the overall estimate. These elements together create the an integrated navigation system, providing accurate and reliable long distance navigation solutions over a variety of conditions.

2.6.1 Historical Terrain Referenced Navigation.

The use of external sensors for IMU updates extends before the application of GNSS, and even with the advantages of using GNSS data, other position estimation methods continue to be developed. These methods and sensors can be used to contribute to the overall position estimate within the INS or allow more robust operation conditions for when other sensors are unavailable or corrupted [45, 125]. This section will give an introduction to a class of navigation solution methods referred to as terrain aided navigation or frequently TRN. In general TRN approaches use an onboard sensor or sensor suite to detect attributes of the ground topology and process them into a position estimate for use in the INS. This family of methods all have the potential to be used to supplement or substitute GNSS data when GNSS is unavailable [125].

One early TRN system was called Terrain Contour Matching (TERCOM) [22] for use on cruise missiles, which utilized a radar altimeter and a barometric altimeter to create a terrain contour line in the direction of flight. The radar altimeter would sample the ground giving a measurement of the vehicle height above the ground, the barometric altimeter would provide a height estimate of the vehicle above sea level. The difference between the two sets of sensor data created a contour line that was an average height topographic map of the terrain, and could be matched using mean absolute difference from an onboard DEM of the flight path. Matching the measured contour and the onboard map successfully generates a position estimate correction that was used to update the IMU drift errors.

Since TERCOM, several related systems have been developed. Sandia National Labs developed Sandia Inertial Terrain Aided Navigation (SITAN) [33] used the same sensors as TERCOM but utilized it on cruise missiles, aircraft (AFTI/SITAN), and helicopters (HELI/SITAN), but instead of collecting a line of ground samples and batch processing them, the SITAN approach utilized banks of Kalman filters to improve the accuracy of the position estimates. Terrain Profile Matching (TERPROM) [85] offers a combination of TERCOM and SITAN and included terrain collision warning and avoidance features. There are additional works that offer algorithmic enhancements to the original TERCOM system (e.g. [104]) but the general approach of utilising a radar altimeter in this way has largely remained the same and some of these systems are still used today.

2.6.2 Image-Based TRN.

A subset of TRN that expands the original concepts of the TERCOM system is called image or vision based navigation [77]. These more modern approaches expand on the concept of recording ground data and matching, then extend the concept to two dimensions using optical cameras, various radars, sonar, scanning lasers, and combinations of them all in hyper-spectral imagery. Matching takes place either with a collected image and a stored map image, or by comparing the collected image to the previous collected image, both use difference information to estimate a change in trajectory. Once a map is matched, the trajectory information is estimated and error correction can be applied to the INS estimate [124, 129, 132].

Images typically have good spatial representation of a location, but suffer from some distortions and ambiguities native to the specific imaging sensor [18, 124]. Image data is generally planar so multiple images, constraints, and additional sensors (e.g; stereo cameras) are used to provide enhanced or third dimension information. Images can also be formed from other data sources than traditional optical cameras to include radar [128] and sonar [58, 68]. Any planar data source is easily described numerically in terms of an image space, even point sources like LiDAR range or intensity data can be collected over an area and converted into spatial images [28].

Image based navigation takes advantage of more possible information available to $2D$ or $3D$ data structures as opposed to the single sample batches utilized in TERCOM-like systems. Howerver this increase in information comes at a cost in data management and processing requirements. Fortunately, the use of imagery allows utilization of a wealth of methods from the field of image processing and computer vision which will be detailed more in Section 2.3. Problems with compression, matching, filtering, and processing have a variety of possible approaches for numerous sensor image and image data types, although their effectiveness on a particular application is an ongoing research area.

A related navigation method employed in robotics and autonomous vehicles is called SLAM. Using SLAM, as a vehicle moves around in its environment it creates or updates a local map and then uses that map to track its own movements. SLAM focus strongly on map creation, autonomy, path planning, and optimization of movement through an environment. SLAM methods include ways to both generate a trajectory estimate and register environment data [19, 51, 53, 130]. In terms of sensing the nearby environment, the mapping portion of SLAM becomes similar to a image navigation problem where sensors are used to detect the nearby area, generate a map, and compare the map to what is already known or mapped [91]. The trajectory of the vehicle can then be estimated based on change of the surroundings relative to itself. Mapping sensors come in all varieties, but notably various LiDAR have been used in robotics for some time. In indoor scenarios LiDAR provides range data of the vehicle to the surrounding environment, creating a relatively high accuracy picture of the vehicles immediate area, and can be collected in high detail quickly. In airborne scenarios, automated Unmanned Ariel Vehicle (UAV) path planning and mapping functions are predominant [12]. In all cases the general SLAM problem is part of the growing field of computer vision, and is taking advantage of both image processing and estimation theory methods [117] and is a good source of techniques related to image based navigation.

2.7 LiDAR Range-Based TRN

Of interest to this research is performing TRN using specifically the range data component of LiDAR sensors. As described in Section 2.5, LiDAR sensors have been popular for some time with remote sensing and mapping activities, but its use as a navigation tool is more recent. A modern version of a TRN process can be generalized into 6 phases as shown in Figure 2.25.

In this process, first raw sensor measurements of the environment are recorded, usually with at least a good estimate of the current vehicle trajectory (position, attitude, velocity, etc...). Next the raw sensor data is refined, this can be a fairly minimal process such as coordinate transformations, or much more involved including filtering and feature extraction but the single goal is to determine what makes this terrain unique for matching. The third step is general matching of the new collected data, and reference data. The matching step involves a process that makes the association that terrain in the collected data is also in the reference. the reference is either previously collected data of the environment (reference matching), the data from the previous successful matching attempt (local matching), or an error corrected product from the applied correspondence transform (reference/map update). Once the terrain data is matched, a correspondence transform can be calculated that would align two data sets (e.g; an affine rotation and translation). The final step is error estimation based on the amount of change required by the correspondence transformation. The estimated error can be used to update the current trajectory information and the process begins again.

This section continues with a brief selection of modern approaches to TRN that are similar to the proposed research in terms of the general approach outlined in Figure 2.25. Then this section reviews the most current efforts with respect to TRN using similar initial criteria as the proposed research. This final section highlights the uses of range data

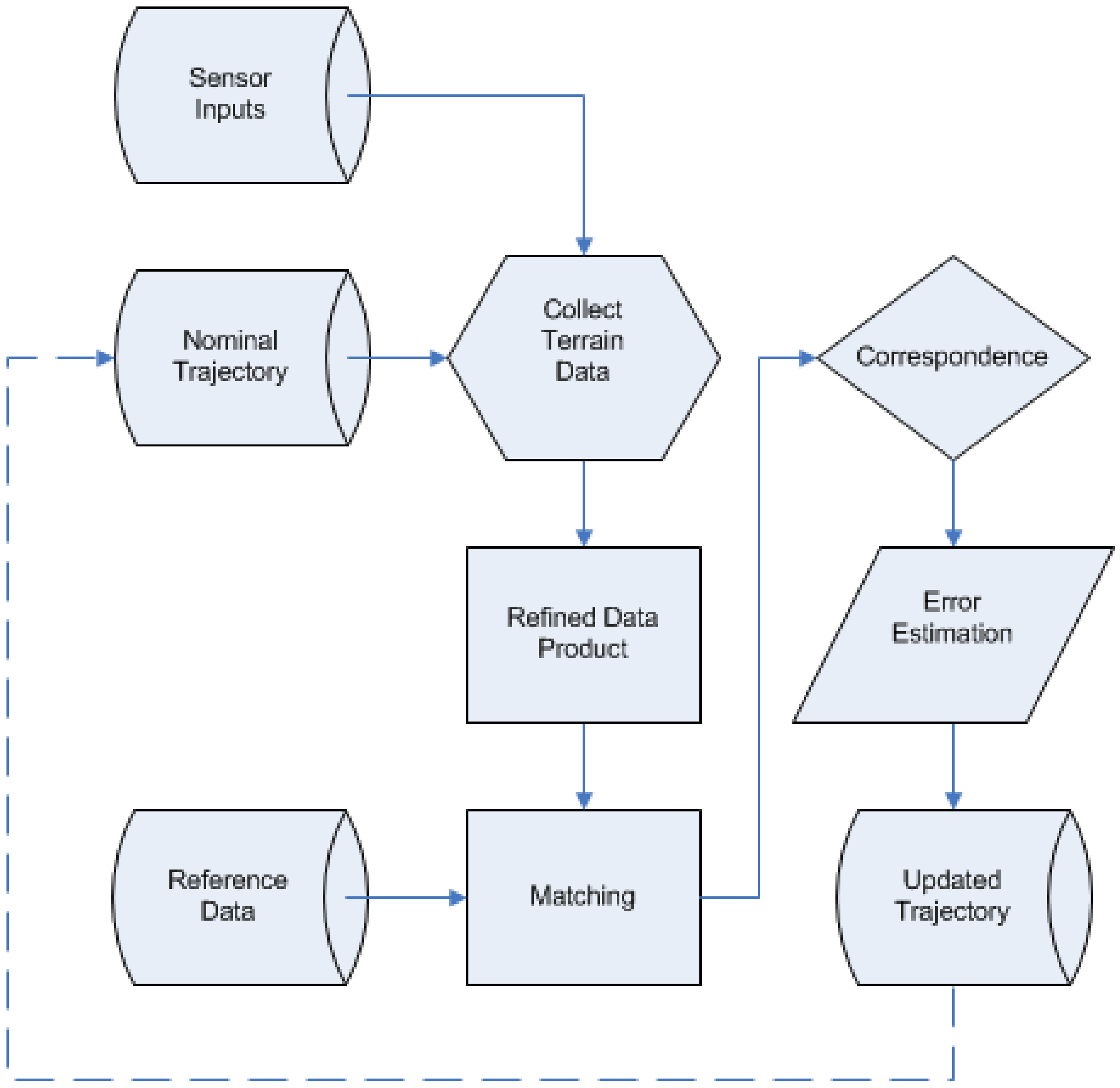


Figure 2.25: A high level flow for modern TRN processes.

derived from scanning LiDAR, the extraction of features for use in matching and solving for correspondence, in the direct application of estimating errors in airborne vehicle TRN.

2.7.1 Current Terrain Referenced Navigation Approaches.

Each step in the process outlined in Figure 2.25 has a large number of approaches to chose from, lending to a great deal of variety in the literature. Just focusing on TRN applications, choice of sensor include but are not limited to cameras [5, 56, 72, 81, 124, 129, 132], sonar [58, 68], radar [128], Flash LiDAR [28, 120], and scanning LiDAR (proceeding sections). Restricting the application space to scanning LiDAR offers two common TRN scenarios, terrestrial and airborne navigation. Terrestrial navigation is dominated by mobile robotics and SLAM approaches, many of which are restricted to indoor environments. As such, these approaches tend to have distinct assumptions and requirements when compared to similar airborne navigation approaches, notably in required data densities, effective sensor ranges, and traveled distances. Terrestrial approaches frequently use sensors and algorithms similar to airborne TRN, and a selected few approaches are mentioned here that also use features and a matching process: indoor navigation [103, 135], outdoor navigation at night [60], additional sensor data [25], and both terrestrial and airborne LiDAR data [122, 123]. The ALS TRN approaches are covered in more detail in the next section.

2.7.2 Methods of Interest Using ALS Range Data for TRN.

The approaches to TRN in this section focus on scenarios involving aircraft, helicopters, and planetary landers. Each of these scenarios utilize feature based or correlation matching, and solve for correspondence utilizing range data from ALS. The exact goals of the navigation scenario and the particular processes they use to derive the correspondence vary significantly. The papers listed here are the closest found in the literature to this researches proposed methodology for ALS TRN. The papers are grouped into 5 efforts with common authors.

2.7.2.1 Hebel and Stilla.

The method proposed by Hebel and Stilla in [32], and summarized in Figure 2.26, utilizes ALS from a helicopter platform in urban areas to reduce INS drift errors in a GPS dropout scenario. The helicopter builds a 3D point cloud representation of the topographic surface, specifically targeting buildings and other man made structures,preferably with multiple ALS passes. The method requires a functioning INS, a initial position estimate, and an off-line computed ALS generated point cloud of the area under ideal GPS conditions, including generated features. To generate features in newly collected point cloud data, each scan line is processed through a 2D RANSAC process coupled with a region growing segmentation process. This both removes outliers from the scan line data and groups the remaining points into line segments. Line segments are then orthorectified into the map coordinate frame, the registered segments are further grouped into planar segments using Euclidean distances, normal direction, and group coplanarity. This process converts the entire point cloud into a collection of segmented 3D planar features. These features ar further refined to remove irregular features. The refinement process calculates the centroid and normal of each plane segment and classifies them. Then ground surface and vegetation associated plane features are removed, leaving only planar features classified as buildings. These features are matched using RANSAC to stored, similarly generated features, from the off-line computed point cloud to generate the translation and rotation components of a correspondence solution. Applying the solution to the previous INS position and attitude estimate, provides the INS error estimation, to be used to reduce the drift errors. The effectiveness of the algorithm was tested off-line on field test data taken at approximately 300 m above the surface elevation, and achieved sub-meter accuracies if at least 10 corresponding planar features are used. The authors express concern over errors increasing significantly with helicopter flying altitude.

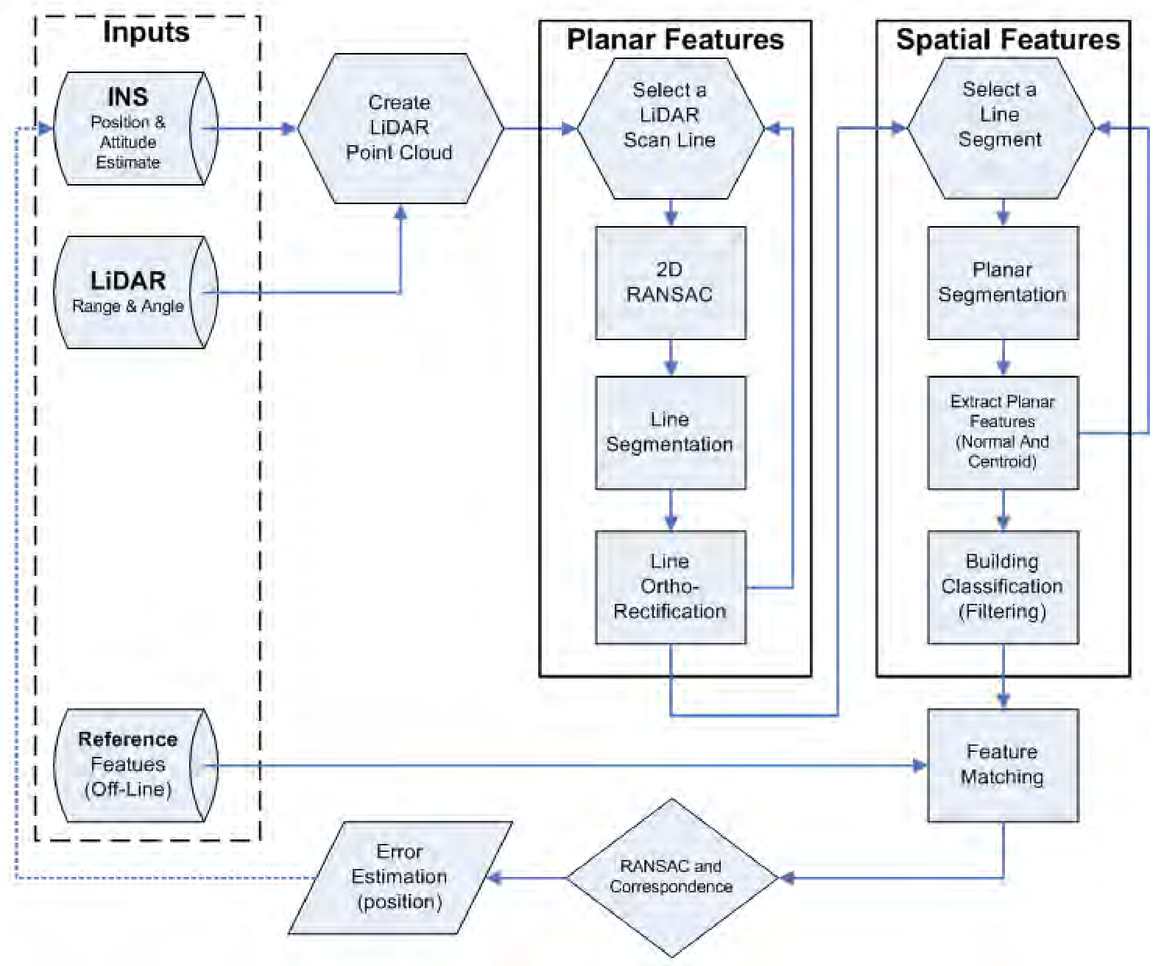


Figure 2.26: Flow of Hebel and Stilla urban multi-pass ALS TRN algorithm using segmentation and classification [32].

2.7.2.2 *Uijt de Haag et al..*

Uijt de Haag et al. proposed two different approaches to ALS TRN. Although the approaches in this section do not utilize features as described earlier, the algorithms are deemed as particularly noteworthy approaches and are thus detailed here. The first approach by Campbell et al. [13] proposes using ALS range data create a point cloud of the topography and matches it to a stored reference point cloud to solve or position and attitude errors. The method is summarized in Figure 2.27. The process assumes a functional GNSS and INS with a ALS system on an aircraft. Each range measurement in a scan line (in this case it is a single back and forth sweep) from the LiDAR is orthorectified into the coordinate frame of the stored reference point cloud, while a second range is calculated using ray-tracing from position and attitude of the aircraft, the scan angle of the LiDAR, and the stored point cloud ground points. The two range measurements are compared using the mean squared difference, and residual error was found by using a correlation search with respect to each translation and rotation variable (6 total). The minimum mean squared difference of the search space produces an offset value for each variable, in total giving the position and attitude errors estimates of the GNSS and INS systems. This system found position errors under 2 m and attitude errors varied between 0.2 and 0.8 deg, utilizing flight test data and a reference point cloud with 2 points per m^2 point density. The authors noted that the found accuracies were highly dependant on reference point cloud density.

In a later paper by Uijt de Haag et al. [27] the proposed ALS system is compared to TERCOM and SITAN based systems for a position estimate update during GNSS dropouts. This process, summarized in Figure 2.28, assumes a position and attitude estimate, an INS, and a reference point cloud are available. The reference point cloud is preprocessed into a gridded 2.5D height DEM at a density of 1 point per m^2, using bilinear interpolation. The ALS range samples are collected and once a large enough set is collected they are orthorectified into the coordinate frame of the reference DEM and similarly processed into a gridded height DEM using bilinear interpolation. An Sum of Squared Error (SSE) surface is calculated between the reference and ALS DEMs. The position error is found by performing a gradient search to find the minimum SSE. The method was tested during a live flight at an approximate altitude of 300 m above the topography. The algorithm was able to correct the INS drift during GNSS dropout for 3D position at better than 1 m accuracy with 2 m standard deviation, an order of magnitude improvement over the TERCOM system, and

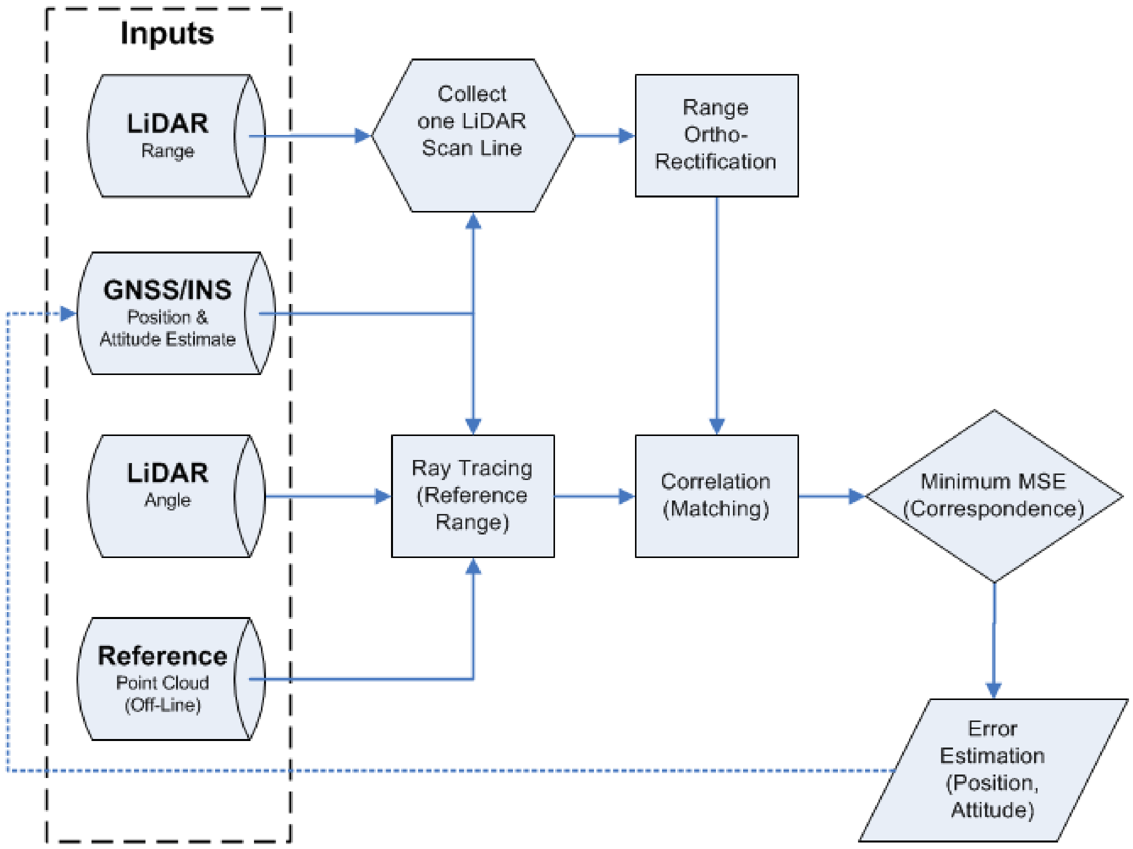

Inputs
LiDAR
Range
GNSS/INS
Position &
Attitude Estimate
LiDAR
Angle
Reference
Point Cloud
(Off-Line)
Collect one LiDAR Scan Line
Range Ortho-Rectification
Ray Tracing (Reference Range)
Correlation (Matching)
Minimum MSE (Correspondence)
Error Estimation (Position, Attitude)

Figure 2.27: Flow of Campbell et al. ALS TRN algorithm using Ray Tracing, Correlation, and Minimum MSE [13].

two orders improvement over the SITAN system. The authors noted the systems sensitivity to potential noise in the INS provided attitude angles.

A third paper by Vadlamani et al. [121], demonstrates a approach that does not need reference terrain data to calculate position estimation errors without GNSS inputs. The system assumes an available GNSS for an initial position and long term position updates, and an INS. Between updates, the accumulating INS drift errors can be estimated by using 2 ALS sensors, one pointing angled fore and one aft. An accumulated point cloud for each LiDAR is collected over 1 second, orthorectified into a common coordinate frame,

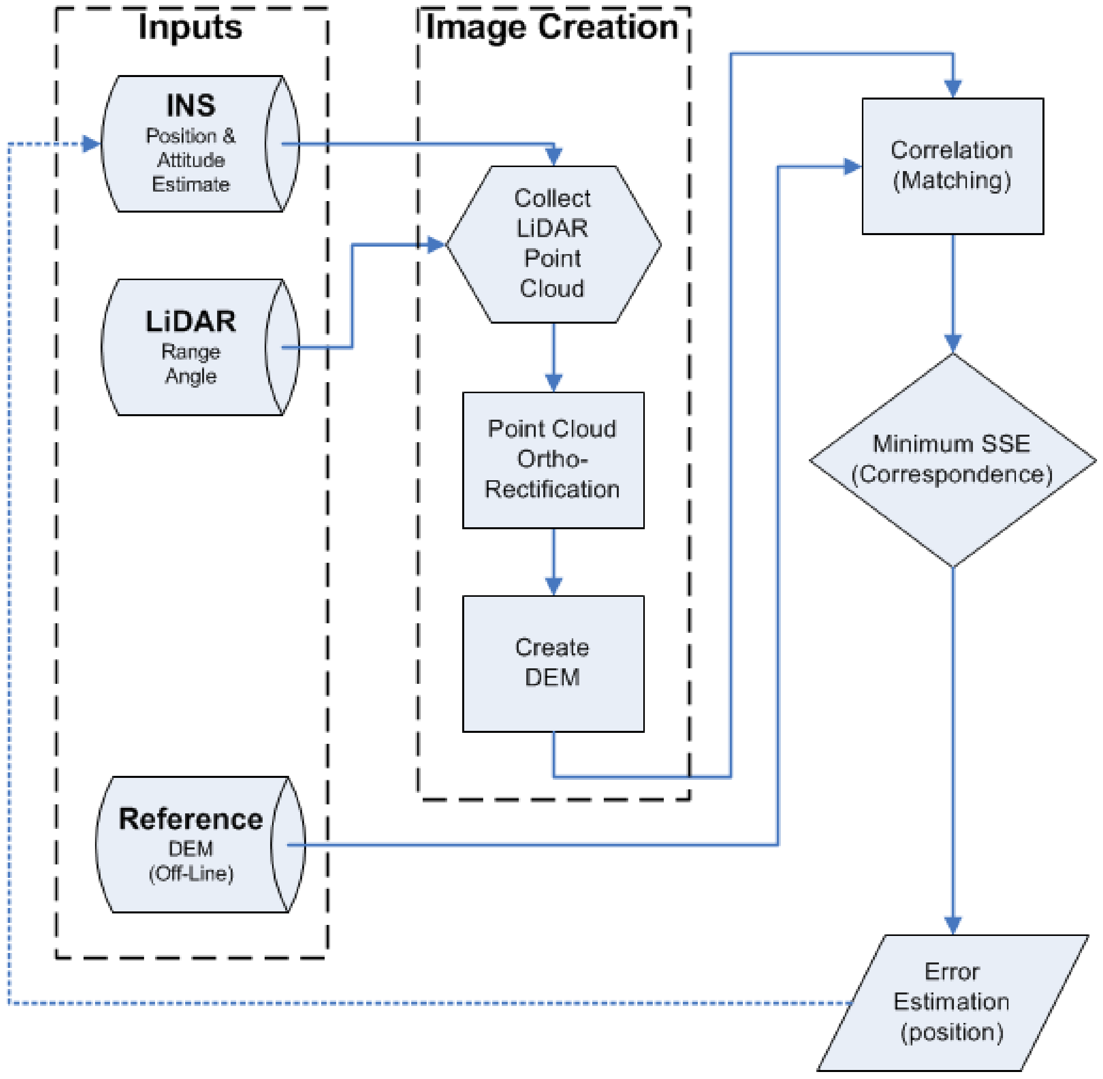

Inputs
INS
Position & Attitude Estimate
LiDAR
Range Angle
Reference
DEM (Off-Line)
Image Creation
Collect LiDAR Point Cloud
Point Cloud Ortho-Rectification
Create DEM
Correlation (Matching)
Minimum SSE (Correspondence)
Error Estimation (position)

Figure 2.28: Flow of Uijt de Haag ALS et al. ALS TRN algorithm using range images, correlation, and minimum SSE [27].

converted into 2.5D height DEMs and compared with a reference DEM. The differences in the two DEMs is created by the INS drift errors that accumulate during the time difference for the aft LiDAR to scan an region previously sampled by the fore LiDAR. A correlation of the two DEMs is performed using a 4 point adaptive grid search, and the algorithm is summarized in Figure 2.29. The minimum error variance offset of the grid search gives the position errors to update the current position estimate. Local matching in this way, with no reference, can reduce but not eliminate accumulating drift errors, hence for long periods of time, or as available, a GNSS or other reference update is needed to reset the position estimate. This algorithm was tested on real flight data, and GNSS-less timing was satisfactorily provided by a regular on-board clock. Position errors were kept below 5 m for up to 3 minutes using a navigation grade INS. Further analysis added nominal attitude errors into recorded flight data, contributing less than an additional 0.5 m over the same time frame. The authors noted that flat or periodic terrain didn't provide enough observable differences to generate unique position fixes.

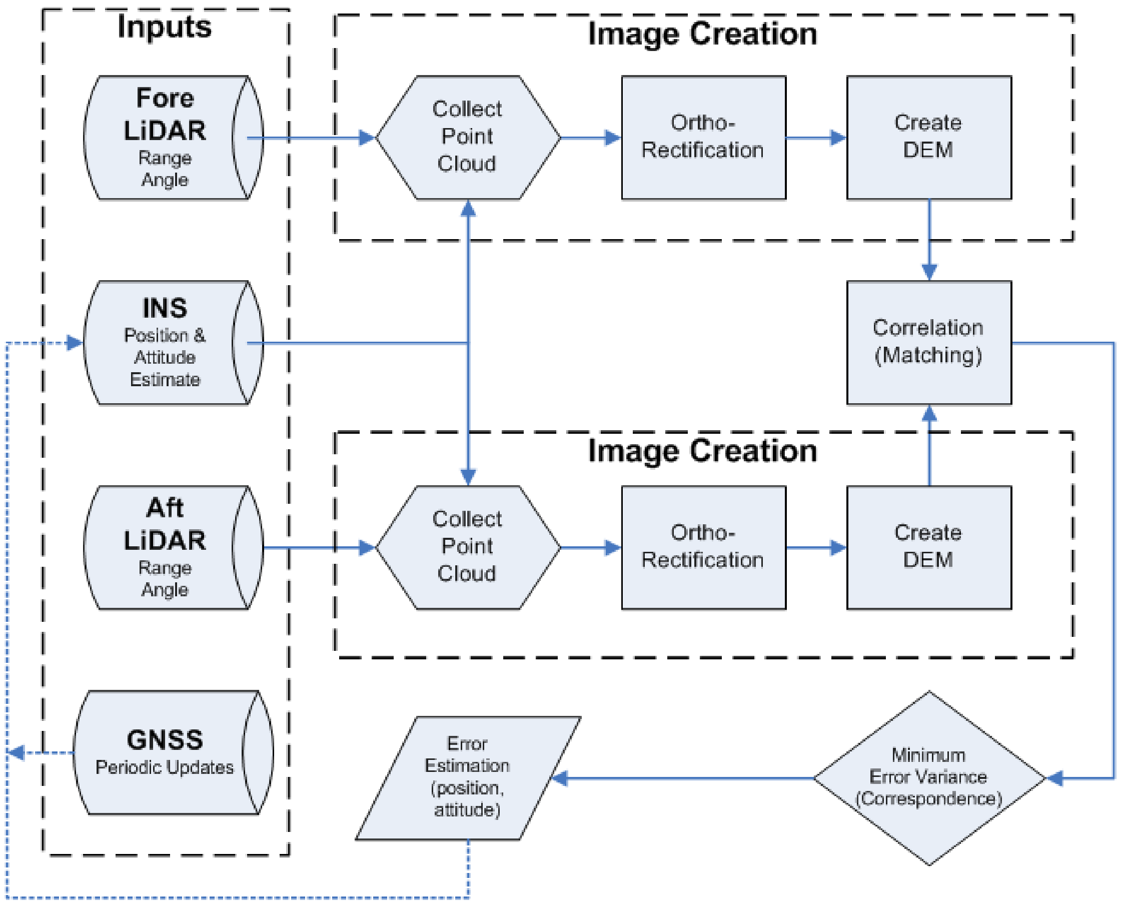


Figure 2.29: Flow of Vadlamani et al. ALS TRN algorithm using two LiDARs, range images, correlation, minimum error variance [121].

2.7.2.3 Johnson et al..

Two TRN approaches are described in works by Andrew Johnson et al. in efforts to detect motion and safe locations for comet or lunar landers. The first paper [40], describes a method using previous scans of the surface as a reference to the current scan, using a ALS system, to determine position. As described in Figure 2.30, a LiDAR scanned swath is used to create a gridded DEM of the topology using bilinear interpolation and a averaging method to fill in holes. The current image is rotationally and scale corrected to match with the previous image using the on board INS sensor data. Features are generated using a variation of Principal Component Analysis (PCA) to determine the "textureness" of a 3D region, and outliers are filtered out using a dissimilarity measure on the same PCA data. Matching is conducted on the features with a variation of a correlation search and the correspondence is drawn from minimizing a of Mean Sum of Errors (MSE). This process generates a translation and uncertainty that aligns the two DEMs and is used to update the current position estimate. Because this method matches between the current and previous scan, errors will still accumulate overtime and the use of a ”key” scan DEM surface is used until the search window exceeds an overlap threshold or the scanned terrain becomes too feature poor, where then a new "key" scan is chosen. The algorithm was tested using terrestrial LiDAR scans of selected comet-like terrain and utilized ground markers for calibration and true position measurements. The authors noted that the absolute errors are dependant on the ground points' sample density, and these errors are approximately half the spacing between range samples.

The second paper [43], focus on a similar lunar landing methodology. This paper assumes a reference map of lunar topology is available and proceeds to explore both LiDAR and camera sensors for best utility for position and velocity estimation. Matching were also explored to include pattern recognition (one of which included SIFT features), correlation, and structure from motion algorithms. Correlation methods were chosen for computation efficiencies, and LiDAR (both Flash and scanning types) preferred over cameras due to the capability to work without sunlight. A computer simulator was developed to simulate scanning LiDAR data and lunar terrain, although the particulars of the simulation better support the application behavior of a Flash LiDAR, and the same simulator is further refined in a later paper [42] using just flash LiDAR. The algorithm varies somewhat from [40], using Harris corner features derived from planar patches within the prepared, gridded DEMs. The feature detection method [41] utilizes some additional photogrammetry methods to give the correlation a better chance at success. A validation

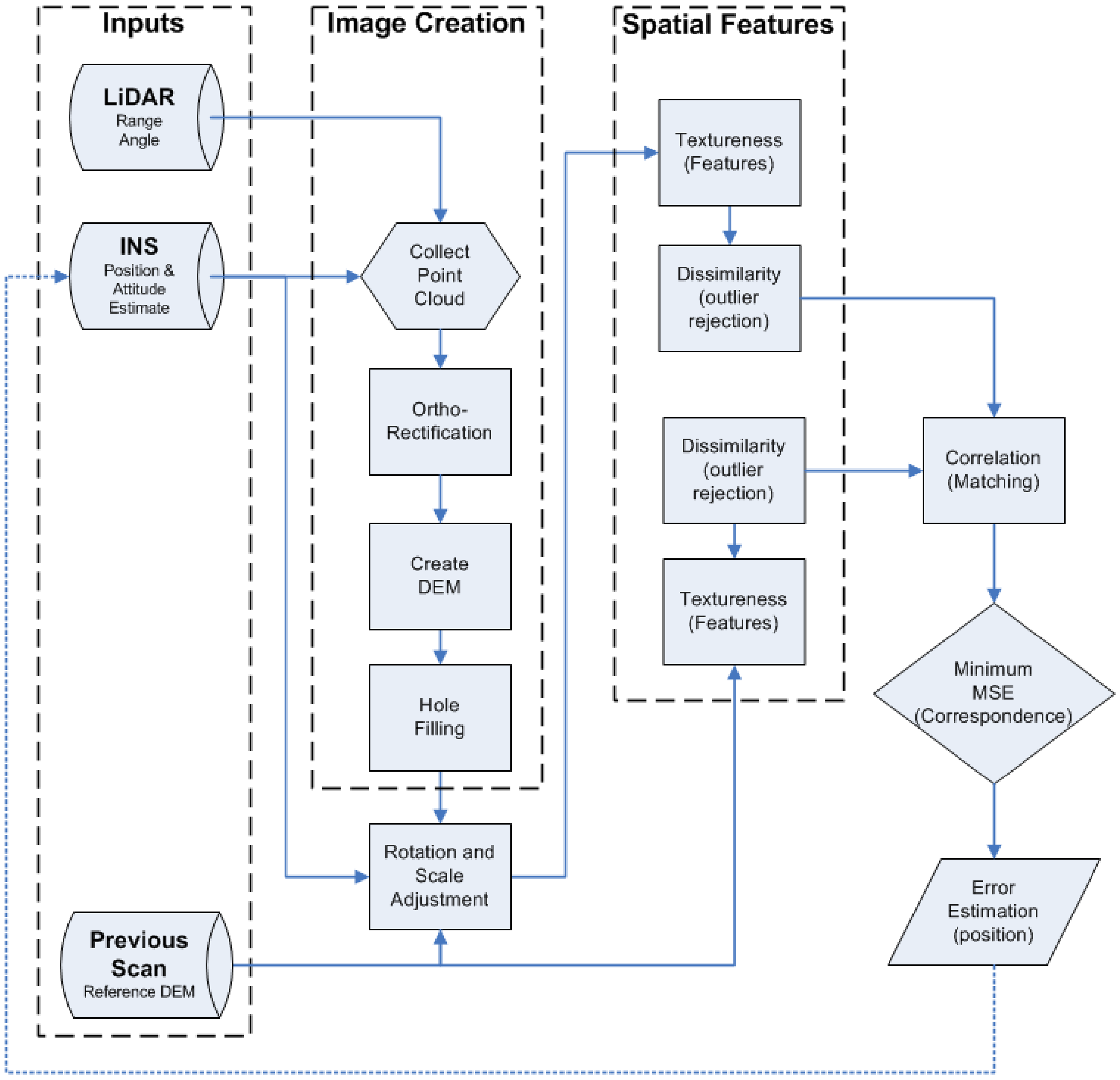


Figure 2.30: Flow of Johnsons et al. year 2000 ALS TRN algorithm using range images, "Textureness" features, correlation, and minimum MSE [40].

check on the matches generated by the correlation to remove weak or erroneous resulting correspondences is used instead of dissimilarity measure outlier removal. These changes are reflected in Figure 2.31. Monte Carlo tests using the simulator found that with scale matched data, ground point density and scanned area size had a strong impact on accuracy of the resulting position estimates.

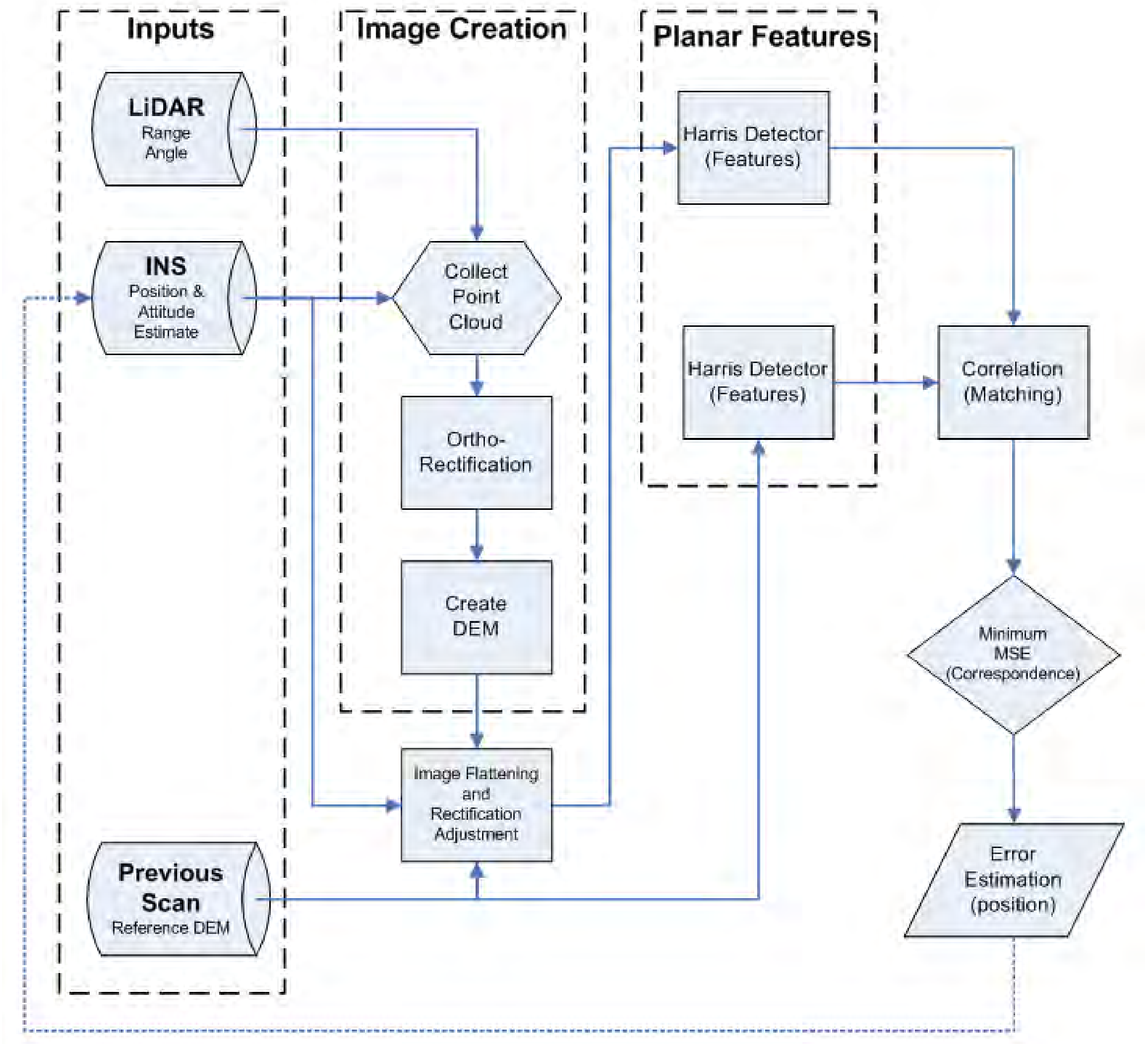


Figure 2.31: Flow of Johnson et al. year 2008 ALS TRN algorithm using range images, Harris features, correlation, and minimum MSE [43].

2.7.2.4 Lafontaine et al..

Partially inspired by Johnson et al. papers, Hamel et al. [31] proposes a planetary lander (Mars) utilizing some similar methods. This approach creates a DEM from LiDAR data, extracts features, matches them and computes a correspondence solution for position estimation. In this work, features are a combination of altitude extremum and slope.

Multiple matching algorithms were tested, including a 2D cross-correlation of features or phase, a 1D feature profile correlation of the feature surface, and a star tracker inspired method, then comparing them to a similar reference feature map. The star tracker method involves refining the features into the most unique of common feature groupings, creating "stars" out of the pool of extracted features (these are point features, not to be confused with actual stars). A stored matrix of "stars", their locations and distances from each other is used as the reference and matching is performed using a constellation search within a tolerance (another type of correlation) to establish correspondence. Using this distance matrix, the minimum mean sum of errors is computed for all detected stars compared to the reference stars finding the position . The weighted mean sum of errors is used as an outlier removal step. The star tracker method performed better than the correlation methods in both speed and accuracy, and is shown in Figure 2.32. Of note is that the star tracker method by using features instead of the total DEM, is able to avoid matching scales in the DEM creation process as done in the Section 2.7.2.3 methods, instead scale is calculated during the correspondence process. Lafontaine is involved with two other papers that detail methods for safe planetary landings using scanning LiDAR. Both contain some crossover with methods shown in [31], but [48] is much more focused on mapping the topography and detecting safe landing sites, and [6] essentially an in-flight LiDAR calibration method.

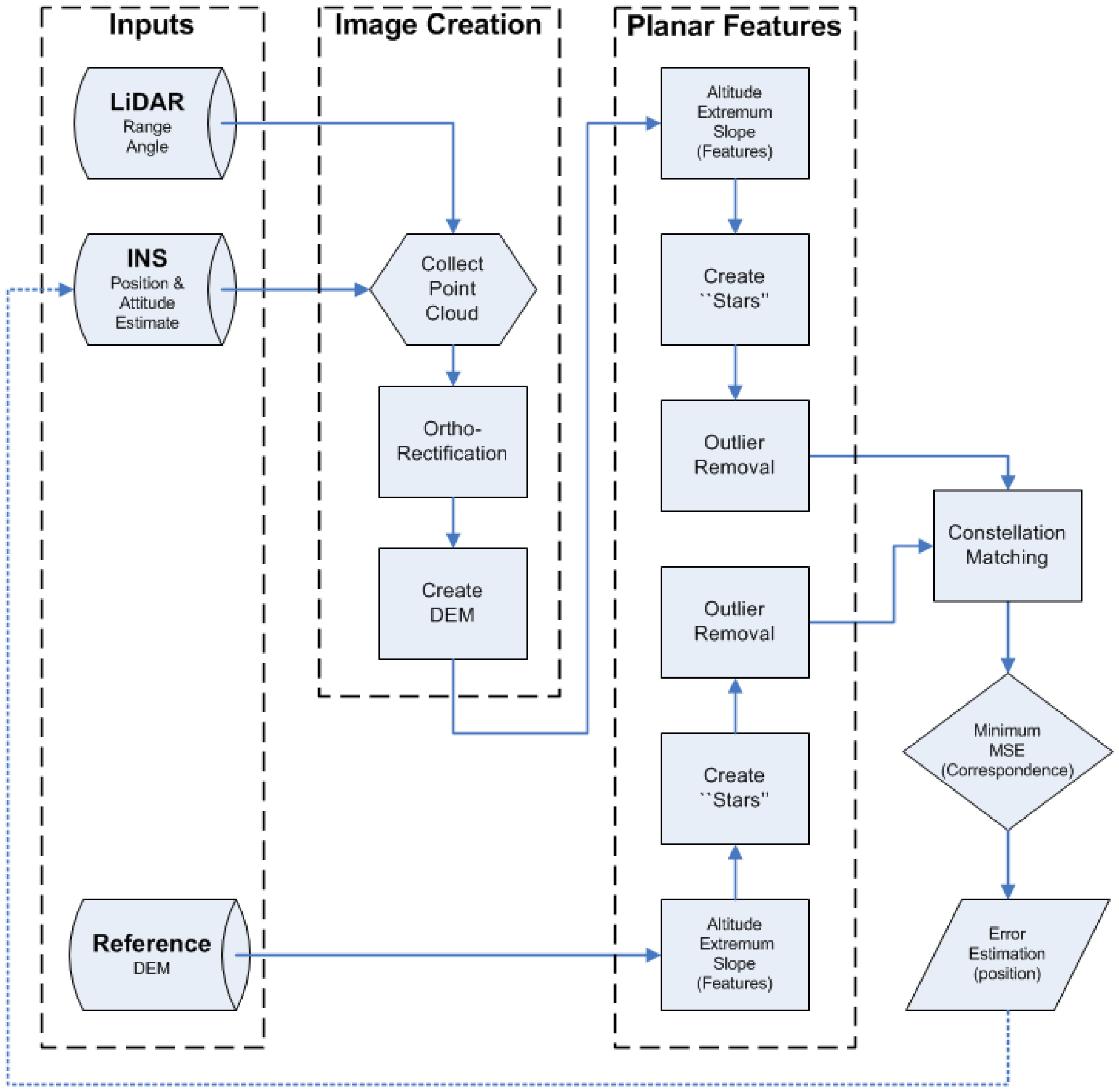


Figure 2.32: Flow of Hamel et al. ALS TRN algorithm using "Star" features, constellation matching, and minimum MSE [31].

2.7.2.5 *Toth, Grejner-Brzezinska, et al..*

Charles Toth, Dorota Grejner-Brzezinska, et al. have a collection of works studying the characteristics and applications of LiDAR senors and their data products, including LiDAR based TRN. The first paper of note by Toth et al. [114] uses narrow sub-strips of LiDAR collected point clouds to match with ICP, estimating the position error correction to the INS. The premise is that over short enough time intervals, deformations in the point cloud samples resulting from INS drift errors are minimal, and can be corrected with frequent, small ICP matchings. The algorithm was tested with a simulated urban environment, with a few samples per m^2 density, and many building-like surfaces achieving sub-meter 3D position accuracies.

An additional paper [112], highlighted that the ICP approach has some difficulty reliably producing a good match under these conditions, and to address the issue a segmentation and filtering method for "breaklines" should be applied before matching. Breaklines are caused by groupings of ground samples that mostly vary in only the vertical component. Breaklines frequently appear in LiDAR scans of buildings, where the rooftops and one or two walls are typically clearly defined and the remaining walls occluded. If the reference data set is scanned from a different direction as the collected set, previously occluded walls could be sampled at different sample densities and distributions depending on the particulars of the flight, creating a different set of breaklines. These differences caused ICP to produce erroneous matches on the small sub-strips, and detecting and filtering out breaklines until just roof tops remain considerably improved performance. Both methods are similar and the breaklines method is summarized in Figure 2.33. It is noted that larger sub-strips improve accuracy at the cost of additional data collection and computation time.

A third paper by Grejner-Brzezinska et al. [24] simulates a comparison of a method very similar to [112] with hyper-spectral optical imagery and ALS. The optical imagery was matched using SIFT and was able to gain meter level position accuracies in simulation compared to the LiDAR sub-meter accuracies. It is noted that the LiDAR data was composed of one second point cloud collections, translating into roughly 70 by 200 m sub-strips at a density of 4 points per m^2. The paper [111] by Toth et al. continues this comparison by using the method outlined in [112] and [24] for scanning LiDAR

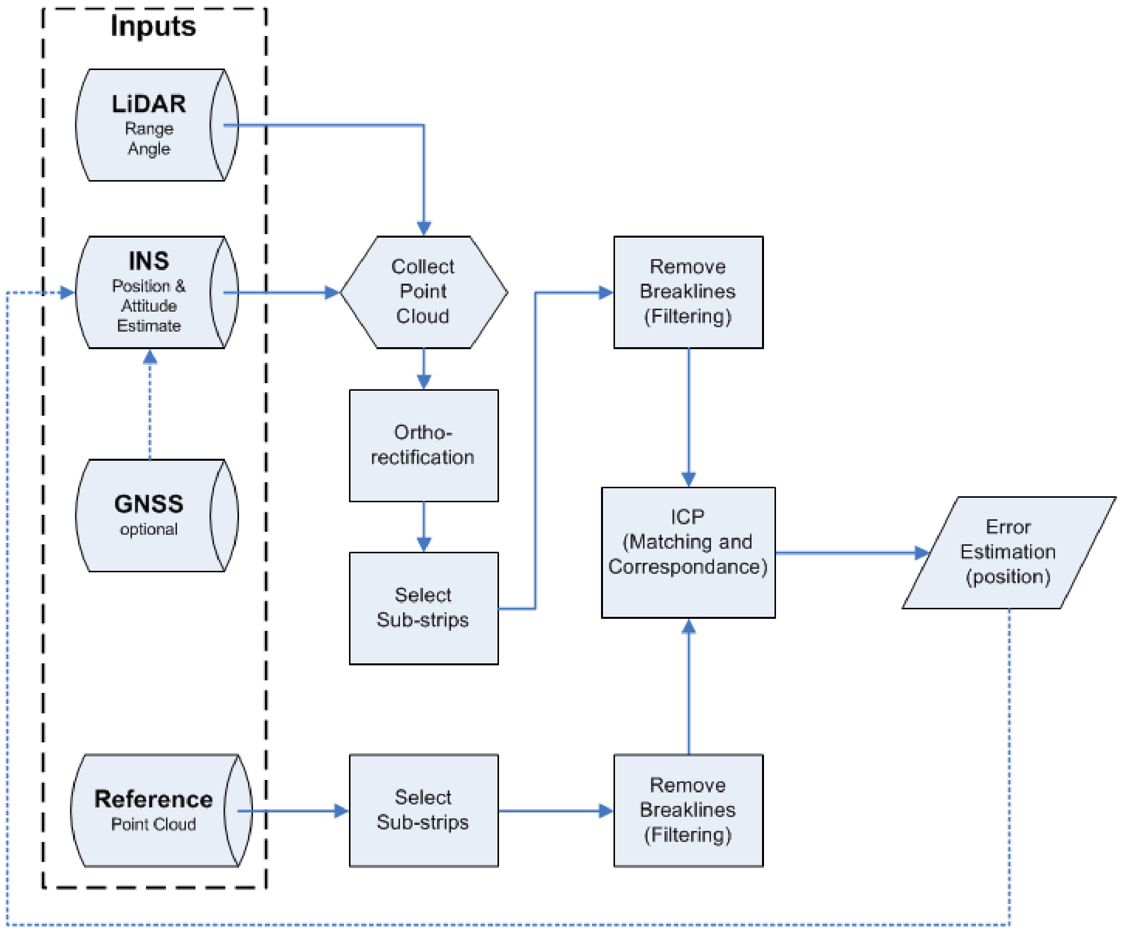


Figure 2.33: Flow of Toth et al. ALS TRN algorithm using Breaklines, and ICP [112].

data matched with ICP. The comparison methods include optical camera images matched using SIFT with an RANSAC outlier removal step, and Flash LiDAR data using a novel eigenvector based segmentation and feature extraction method, RANSAC outlier removal and a correspondence step using [35]. Each data type was matched with a reference of the same data type, and tested in simulation for accuracy. All three methods were found to be viable for TRN, with sub-meter accuracies in the scanning and flash LiDAR based approaches.

A second data type comparison paper by Toth et al. [115] which performed SIFT feature extraction followed by RANSAC outlier removal and matching on different data types, though actual TRN was not performed. The number of correct matches and the 2D position error distribution of the matched features was used as the quality metric on 4 data types under simulated error conditions. The comparisons included matching between optical camera images, camera and satellite images, between LiDAR based DEMs, and between LiDAR based intensity images. Optical camera images performed the best with numerous SIFT features generated and the most matches after RANSAC, although the error distributions were notably skewed in one direction, speculated to be from differences in resolution and flight direction. Camera matched with satellite images also produced many SIFT features, but considerably fewer good matches, though it was enough for accuracies similar to the camera only matches. The LiDAR height DEM matching produced notably fewer features and only a about a dozen matches, resulting in higher uncertainty than the other methods. The LiDAR intensity image matching performed almost as well as the optical camera matching, so in terms of using SIFT features intensity images were preferred.

Another paper by Oh et al. [73] proposes a different approach based on the algorithm outlined in [112]. LiDAR intensity and DEM images matched to reference models could be used to derive 3D position errors. This approach uses breaklines as a feature extraction tool, creating a breakline image similar to image edge detection filters. This feature image is then matched using a novel correlation variant called relative edge cross correlation (RECC), followed by a matching success metric (the concentration value) to filter out high uncertainty matches. The algorithm is outlined in Figure 2.34. This method is focused on providing a TRN update under the normally difficult condition of nearly flat terrain. If the terrain being scanned has a strong features in the intensity image (its reflective properties of the LiDAR beam) or if the terrain has good height contrast, the better position estimate

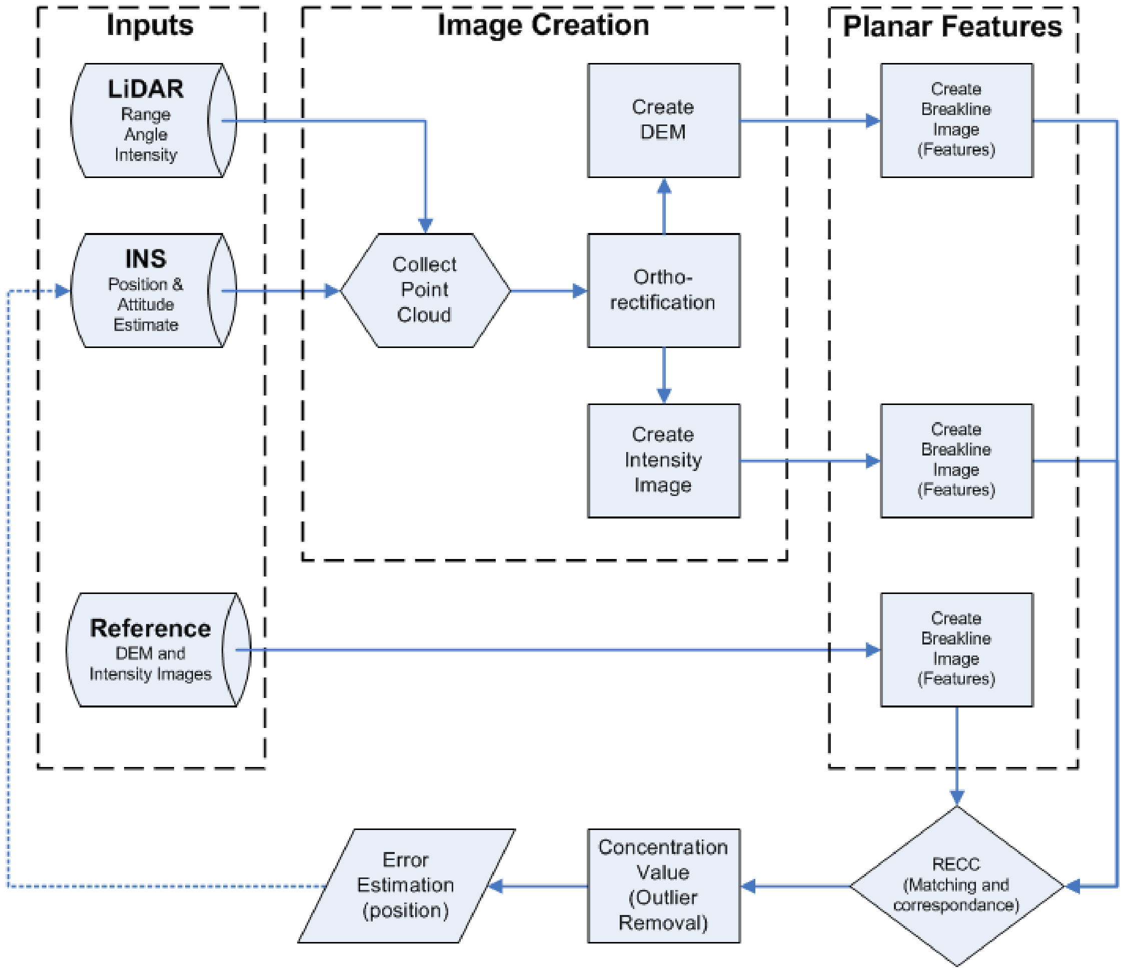


Figure 2.34: Flow of Oh et al. ALS TRN algorithm using Breakline images, RECC, and a concentration value filter[73].

between the two matches is chosen to update the INS. This method is noted as not effective on terrain that cannot meet either of these scenarios like deserts and grasslands. The resulting accuracies were generally under a meter, and the algorithm was tested on real LiDAR strips collected within minutes of each other at an average of 1.8 samples per m^2.

III. TRN Using ALS Range-Based SIFT Features

In this chapter the proposed approach to ALS TRN using range-based SIFT features is described. Each major component of the full algorithm is described in the following sections.

3.1 Proposed Algorithm

The proposed method uses SIFT features to match ALS collected point cloud range data to an off-line prepared reference point cloud. The algorithm finishes by establishing a correspondence transform that aligns the set of ground features extracted from both the collected and reference point clouds. Applying the calculated correspondence transform to the nominal aircraft position and attitude provides an new estimate of the true position and attitude. The algorithm to perform this operation is broken down into several steps:

- Input requirements
- Point cloud collection and ortho-rectification
- DEM creation
- Feature extraction
- Feature matching
- Outlier removal
- Point cloud feature selection
- Iterative correspondence calculation
- Position and attitude estimation

The flow of the proposed algorithm is summarized in Figure 3.1.

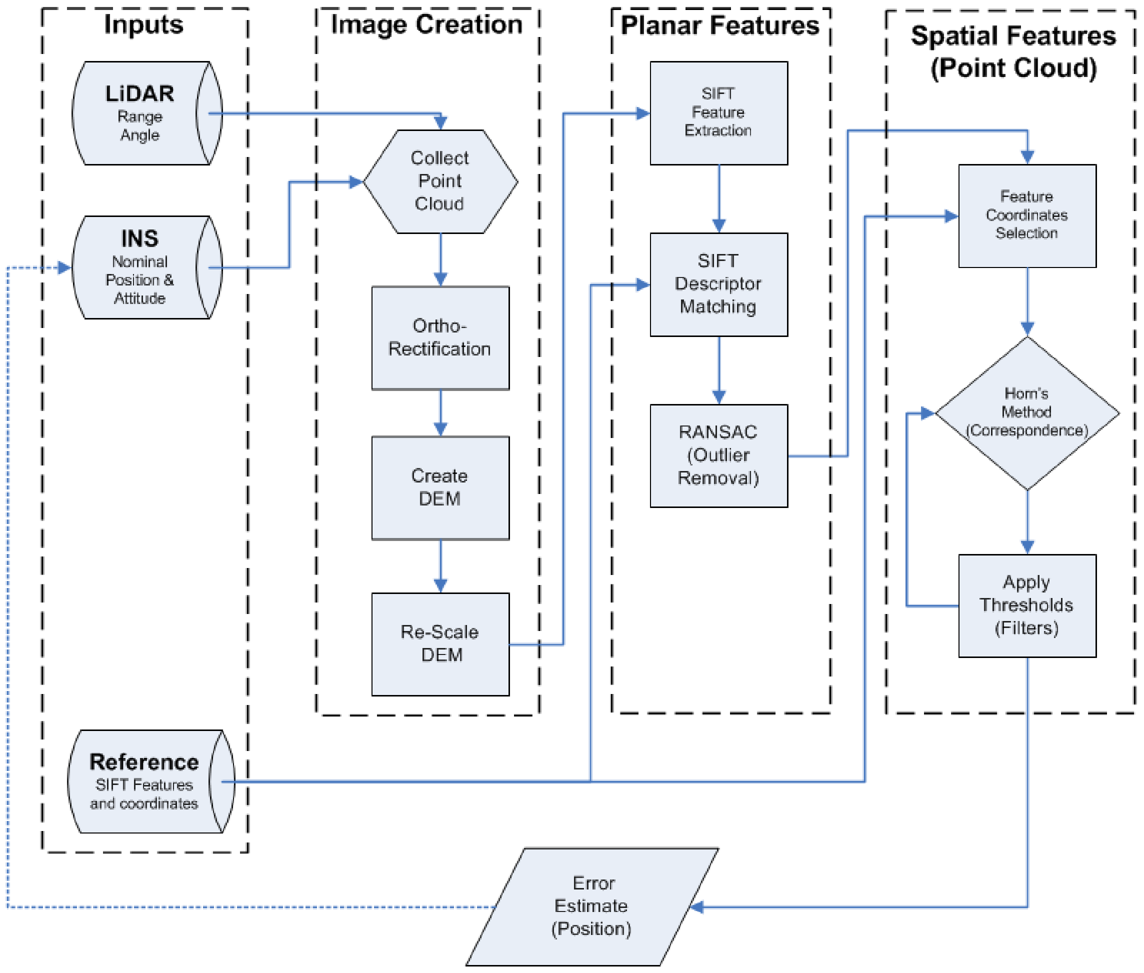


Figure 3.1: Proposed SIFT range-based ALS TRN algorithm flow chart.

3.1.1 Algorithm Inputs.

This method has input requirements similar to the methods described in Section 2.7.2. An ALS is mounted to an aircraft to record range data from the aircraft to the ground. An INS is assumed to be available to provide a nominal position and attitude estimate of the aircraft for each LiDAR range sample. GNSS position information is optional in this setup, assumed to be degraded or unavailable, though the receiver itself may be present. GNSS timing information or at least a reasonably accurate timing system is required to synchronized the INS and the ALS measurements. The reference point cloud information is assumed to be available and prepared for the flyover region ahead of time. These inputs are listed in Table 3.1.

Table 3.1: Algorithm required inputs.

Source:	**Provides:**		**Defined:**
LiDAR	Sample Positions		Equation (2.52)
INS	Position	Attitude	Equation (2.24)
Clock	Timing		
Reference	SIFT Features	Feature Positions	Section 2.3.1

The reference point cloud represents a large region where the aircraft is expected to fly in. The region's area potentially leads to very large amounts of representative LiDAR data. Reducing the reference region into just the elements required to perform the proposed algorithm by performing the necessary processing steps off-line before flight helps alleviate some of the processing needs. In the proposed algorithm, the reference region is further subdivided into small sections, and the most relevant section is chosen to be compared to the collected LiDAR point cloud. The relevant reference section is chosen to contain the nominal INS position estimate and is expected to contain more than 50% of the collected point cloud area.

3.1.2 Collection and Orthorectification.

As the aircraft flies the scanning LiDAR sweeps the ground collecting range information time synchronized to the INS position and attitude information. Once enough ground area has been sampled by the LiDAR to make a useful match, the individual sample points are collected into a point cloud located with respect to the LiDAR sensor. To make the collected point cloud into a regional surface sampling of the local topography, the point cloud is transformed (ortho-rectified) into the same coordinate frame as the reference point cloud. The processes of point cloud ortho-rectification are detailed in Section 2.5. This process results in a collected point cloud with each point coordinate ortho-rectified into the same coordinate frame as the reference point cloud. The collected point cloud can be represented as a set of position vectors

$$\bar{\mathbf{a}}_{pc} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{3.1}$$

were n is the number of coordinate points in the point cloud. Note that the coordinates of the points in the ortho-rectified point cloud will be affected by INS errors.

3.1.3 DEM Creation.

The original SIFT algorithm was designed to extract features from photographic intensity (2D, black and white, binary scale) images but is not inherently able to process 3D LiDAR data. To utilize SIFT, the scattered points that comprise a LiDAR point cloud must be converted into a 2D image representation and converted into an intensity image format that SIFT will understand. This creates several additional steps in the process related to applying SIFT, and this methodology is far from optimized.

The collected point cloud heights are projected into the (x, y) plane forming a 2.5D height image (a DEM). The formation of a DEM is performed using a rasterization process as mentioned in Section 2.4.4 and illustrated in Figure 3.2. To prepare the DEM for use with the version of SIFT used in this research, the height values must be converted into an 8-bit gray scale. The conversion is performed by the re-scaling and quantization of the height values in the DEM to the gray scale shown in Figure 3.3. The gray scale assignment is performed as each new point cloud processed, with the minimum and maximum heights present representing 0 and 1 respectively. The heights between the maximum and minimum values are scaled linearly, although this may cause some interpolation of the original

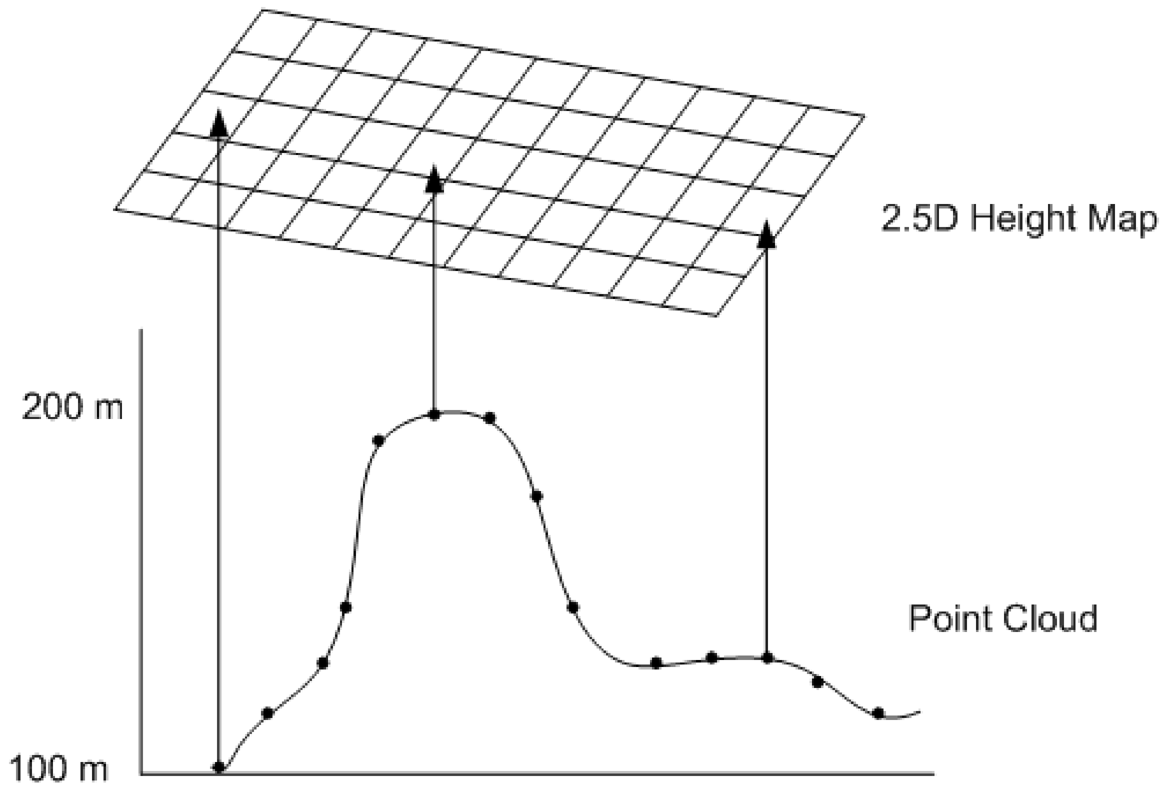


Figure 3.2: Depiction of the rasterization of the point cloud into a 2.5D height image or DEM.

heights. The 8-bit gray scale assignment creates 256 levels of intensity values for the range of heights present in a given DEM.

An example of a scaled range image is shown in Figure 3.5, using this process on a region illustrated in Figure 3.4. Figure 3.6 is shown to illustrate how different a range image of a region is from a similar representation using LiDAR intensity data. At the end of this step, the original LiDAR ortho-rectified point cloud is represented as a scaled range image. The DEM of the collected point cloud is a matrix representation of point cloud elevation

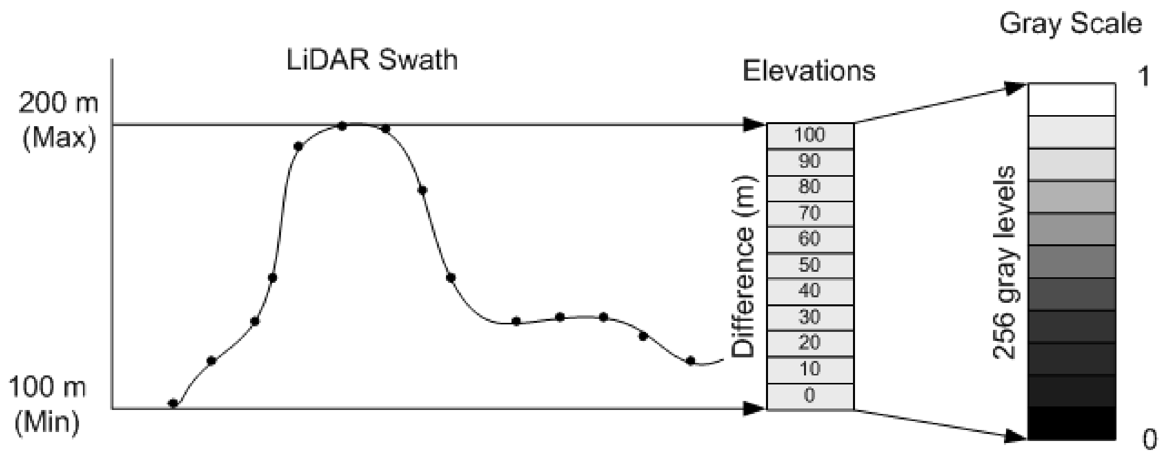


Figure 3.3: The coloring (intensity) of the resulting image is determined by the range of elevations present in the point cloud before converting into gray scale.

values, arranged in the (x, y) plane

$$\mathbf{A}_{DEM}(x_i, y_j) = \begin{bmatrix} z_{1,n} & \cdots & z_{n,n} \\ \vdots & \ddots & \vdots \\ z_{1,1} & \cdots & z_{n,1} \end{bmatrix} \tag{3.2}$$

where each

$$z_{x,y} = [0 \text{ to } 255]$$

where each elevation value z is placed at its (x, y) coordinate for all n coordinates in $\bar{\mathbf{a}}_{pc}$ and empty raster cells are filled by linear interpolation of the neighboring cells. Each z elevation value is then interpolated into a 256 level gray scale. This process is also performed off-line on the reference point cloud sections, creating $\mathbf{B}_{DEM}$.

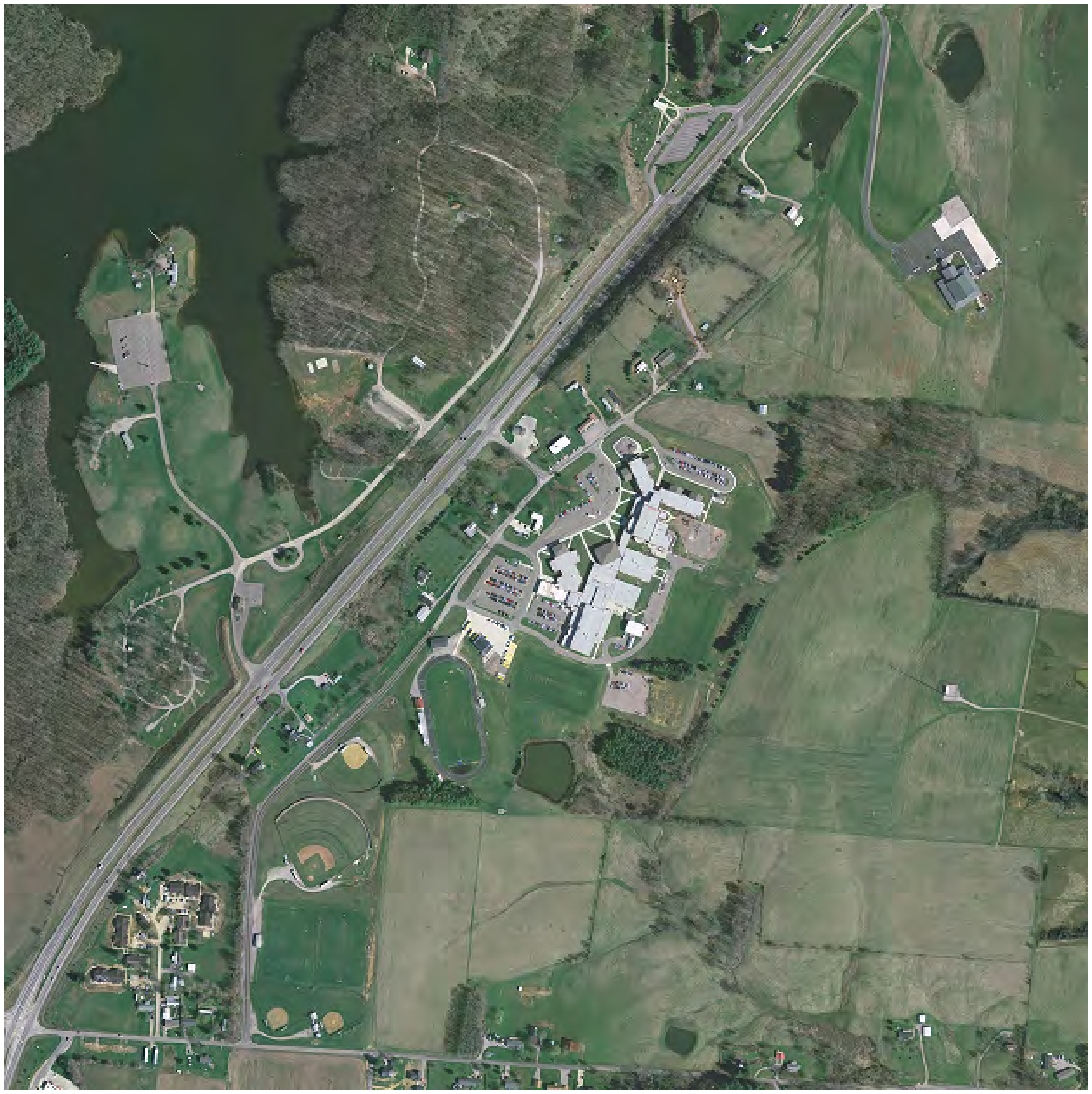


Figure 3.4: Photograph a region to be flown over and scanned with a LiDAR [74].

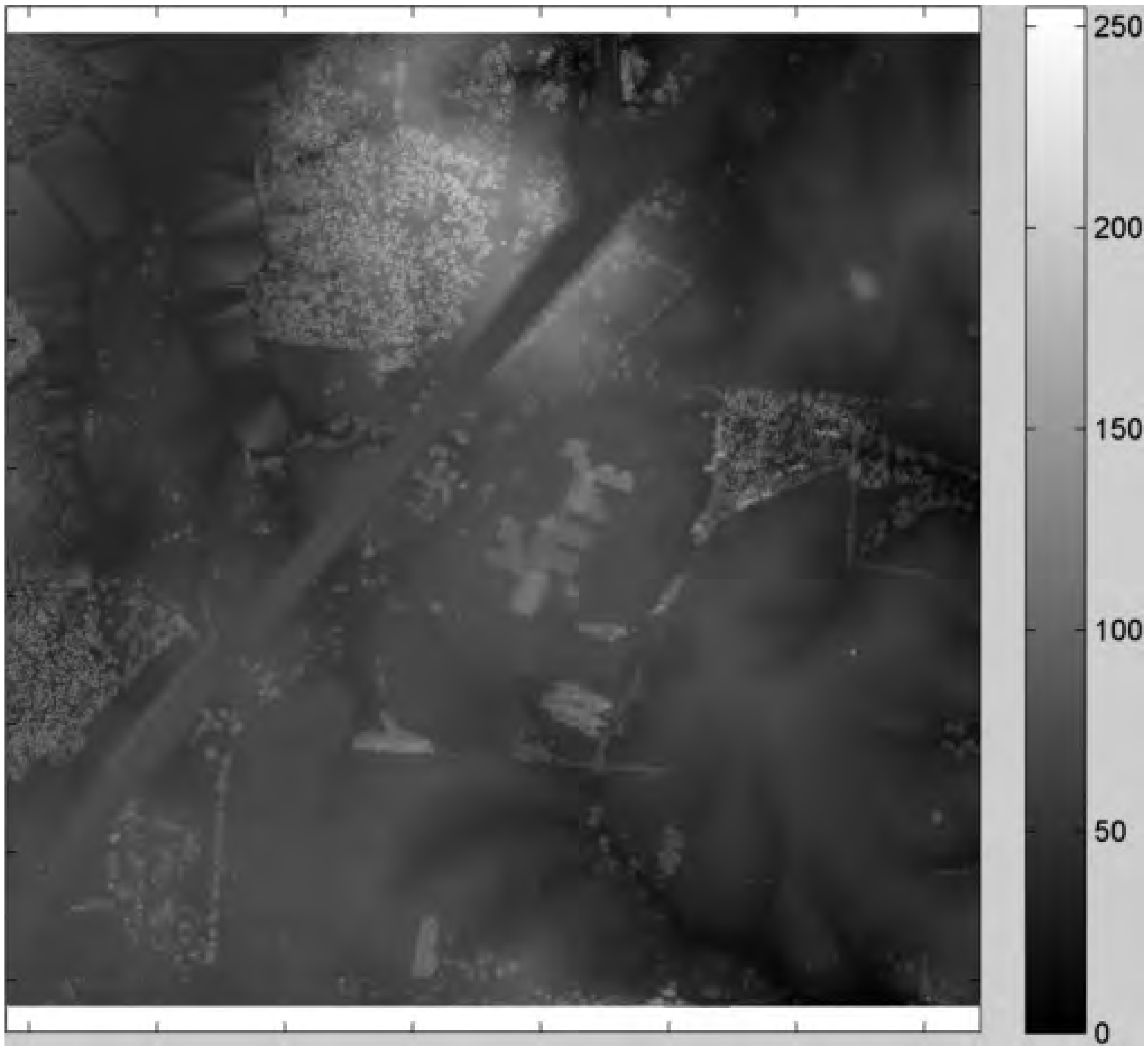


Figure 3.5: A scaled LiDAR range image using the process described in Section 3.1.3 on the region in Figure 3.4. This represents the changes in elevation of the represented terrain.

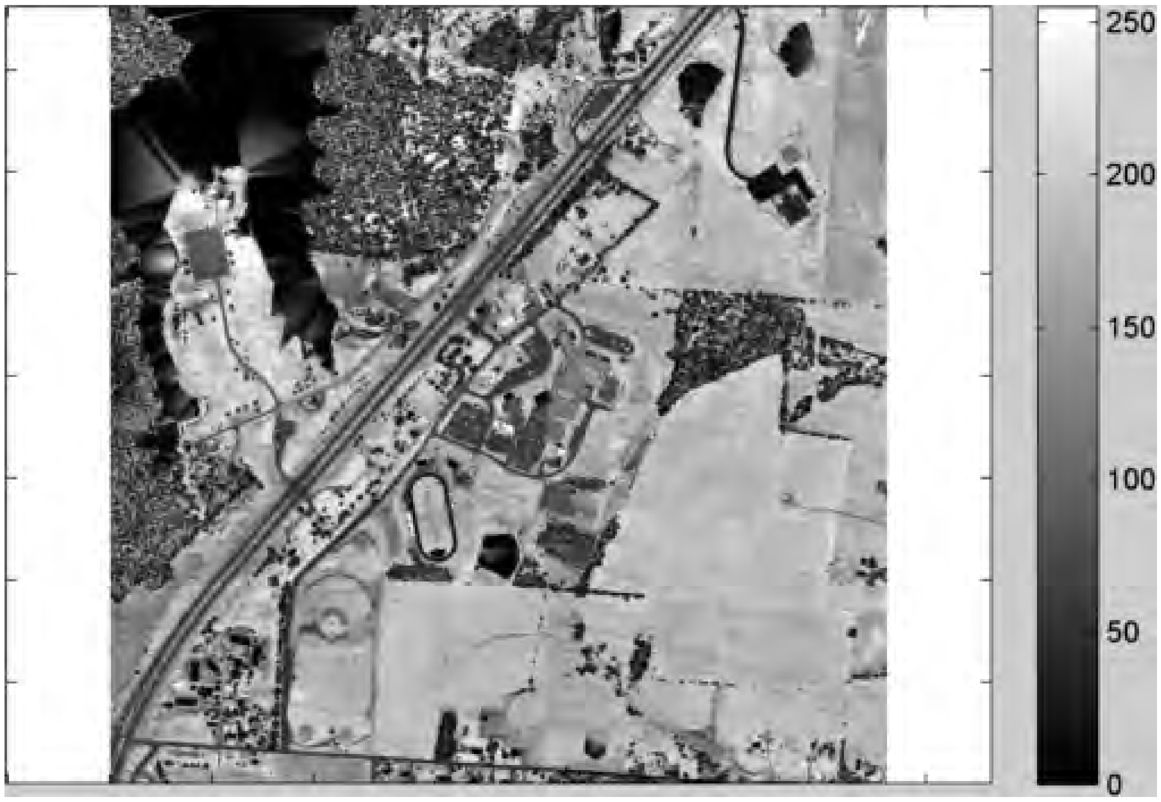


Figure 3.6: A LiDAR intensity image of the region in Figure 3.4. Not to be confused with a range image, intensity images are a 256 grayscale of the LiDAR reflection energy intensity.

3.1.4 SIFT Feature Extraction.

The SIFT algorithm as described in Section 2.3.1 is then performed on the scaled range image. SIFT creates features from the image data. Each SIFT feature is represented by its location in the image, scale, orientation, and 128-byte descriptor. SIFT features have the added benefit of being invariant to changes in scale and orientation, and partially invariant to some other distortions of the image. The invariance is especially useful in the case here where the SIFT features in the collected image will be compared to features created from an image derived from the reference point cloud. The reference point cloud was likely collected using a different LiDAR sensor, under different conditions, altitude, and direction, any of which may impact the resolution (scale) and orientation of the generated range image. SIFT features are locations selected as unique, in a way that should be invariant to these differences, allowing a match to be made. Figure 3.7 shows the SIFT features detected in the range image (Figure 3.5), with each feature shown with a representative scale and orientation. This step concludes with two sets of SIFT features, one set from the collected range image $\mathbf{A}_{DEM}$, and a set created off-line from the reference point cloud section derived range image $\mathbf{B}_{DEM}$. The set of SIFT features detected in the collected image is given by

$$\bar{\mathbf{a}}_{SIFT} = \begin{bmatrix} (x,y)_1 & \sigma_1 & \vartheta_1 & Descriptor_1 \\ \vdots & \vdots & \vdots & \vdots \\ (x,y)_n & \sigma_n & \vartheta_n & Descriptor_n \end{bmatrix} \tag{3.3}$$

were (x, y) represents the keypoint location, σ is the scale, ϑ is orientation, *Descriptor* is the 128 element descriptor, and n is the number of SIFT features detected in the scaled DEM. Similarity,

$$\bar{\mathbf{b}}_{SIFT} = \begin{bmatrix} (x,y)_1 & \sigma_1 & \vartheta_1 & Descriptor_1 \\ \vdots & \vdots & \vdots & \vdots \\ (x,y)_n & \sigma_n & \vartheta_n & Descriptor_n \end{bmatrix} \tag{3.4}$$

for the reference range image SIFT features.

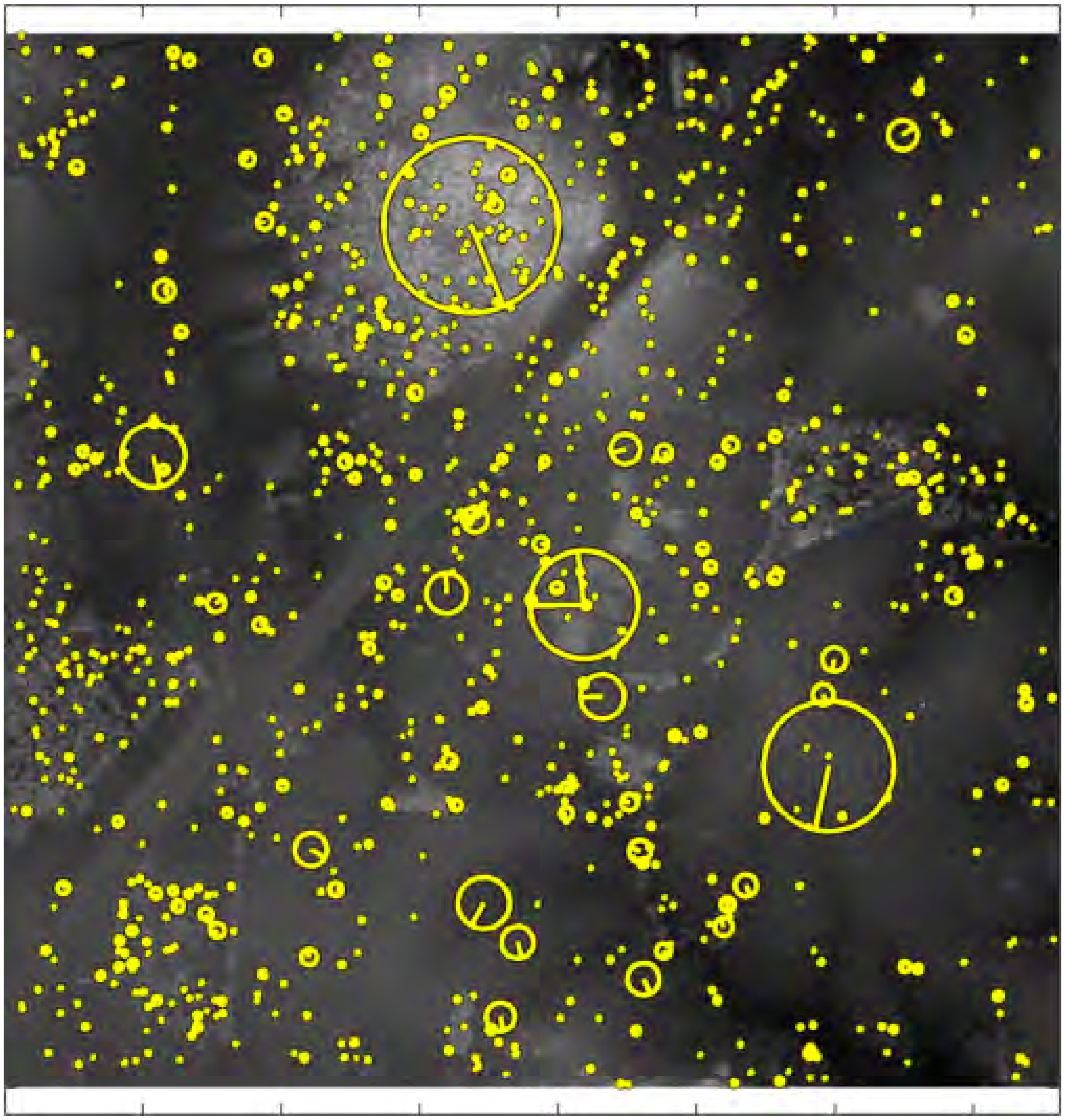


Figure 3.7: The LiDAR range image from Figure 3.5 with a sample of the SIFT features present (1000 shown of the 6500 features detected).

3.1.5 Feature Matching.

The SIFT features from the collected and the reference range images are compared using their SIFT 128 byte descriptors. An efficient nearest neighbor descriptor search algorithm [65] is used find matching descriptors, which would imply that the same keypoint is in both images. An example of the results of this process is illustrated in Figure 3.8 with a collected DEM that is partly overlapped with a reference DEM. Note that there are considerably fewer features found to match between the collected and the reference range images. Only 92 of the 1058 features available in the collected DEM were found matching in the Figure 3.8 example. Also note that some features are matched to wildly incorrect locations. Each SIFT feature is not infinitely unique and therefore SIFT is not immune to creating poor keypoint associations. This step results in a subset of SIFT features that are identified as present in both the collected and reference range images. The set of collected data SIFT features $\overline{\mathbf{a}}^{*}{}_{sift}$ are descriptor matched to the reference data features $\overline{\mathbf{b}}^{*}{}_{sift}$ and are a subset of the total detected SIFT features such that

$$\overline{\mathbf{a}}^{*}{}_{SIFT} \subseteq \overline{\mathbf{a}}_{SIFT} \tag{3.5}$$

$$\overline{\mathbf{b}}^{*}{}_{SIFT} \subseteq \overline{\mathbf{b}}_{SIFT} \tag{3.6}$$

where each feature in one set is matched to a feature in the other such that

$$\overline{\mathbf{a}}^{*}{}_{SIFT(Descriptor)} \approx \overline{\mathbf{b}}^{*}{}_{SIFT(Descriptor)}. \tag{3.7}$$

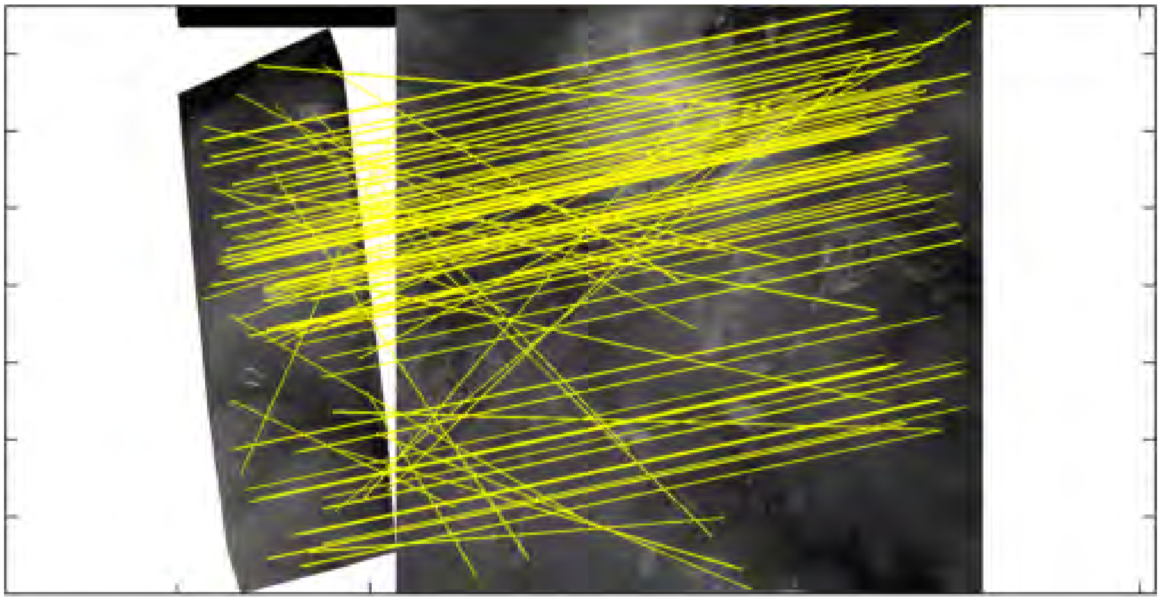


Figure 3.8: The smaller range image on the left was produced from a collected point cloud, while the right image was chosen from the reference point clouds. The lines connecting each image show the association that the descriptor search made between matching SIFT features.

3.1.6 Outlier Removal.

To remove poorly matched SIFT feature pairs, RANSAC as described in Section 2.3.2.3 is performed. RANSAC randomly selects a small group of features and uses a solver like [35] to determine a correspondence transform, creating a consensus set. The algorithm performs this iteratively, identifying features that do not strongly agree with the current consensus transform and then removing them from the consensus set, until a threshold is met. The end result is a subset of feature pairs that consistently share a similar transform. The features not in this final consensus set do not positively contribute to the consensus solution and are ignored as set outliers. An example of this is shown in Figure 3.9, which shows matches after RANSAC has been applied. In this case, when compared to Figure 3.8, all the erroneous matches appear to have been removed. The features that survive

RANSAC are a subset of the matches made by the descriptor search. In this example the 92 matched features have been reduced to 36 consistent matches by RANSAC outlier removal. At the end of this step, the previously matched collected image SIFT features and the reference SIFT features have been reduced to a subset of matched features with very similar correspondence transforms. Similar to the previous step, applying RANSAC to the SIFT feature pairs gives

$$\overline{\mathbf{a}}_{RANSAC} \subseteq \overline{\mathbf{a}}^{*}_{SIFT} \tag{3.8}$$

$$\overline{\mathbf{b}}_{RANSAC} \subseteq \overline{\mathbf{b}}^{*}_{SIFT} \tag{3.9}$$

and each matched feature pair has a similar corresopondence transform such that

$$\overline{\mathbf{a}}_{RANSAC} \cong \overline{\mathbf{b}}_{RANSAC} \tag{3.10}$$

to within the RANSAC threshold.

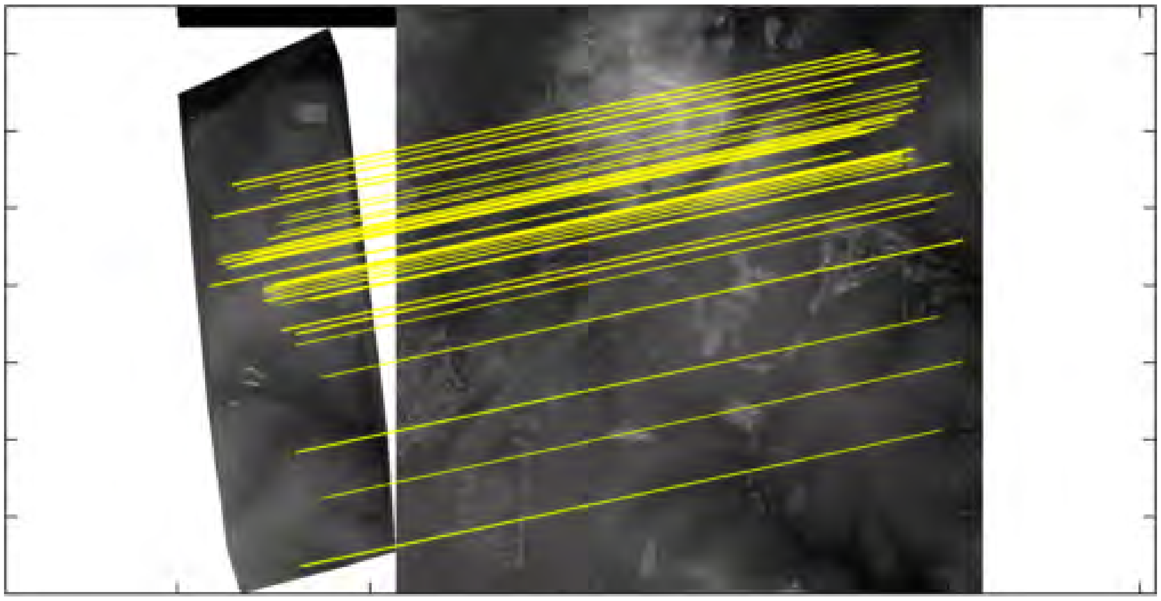


Figure 3.9: Shown is the result of applying RANSAC to the matches shown in Figure 3.8.

3.1.7 Point Cloud Feature Selection.

If this was a purely image based approach (such as with a camera) the algorithm would conclude here for the error estimation process. But in the LiDAR-based approach, the images used in SIFT matching were derived directly from collected 3D data, and in this proposed approach the SIFT features can be used to select directly detected 3D features from the original point cloud. From the SIFT keypoint location information of the remaining descriptor matched and RANSAC constrained features, the coordinates of the original point cloud samples that the features were derived from can be recovered. This process is illustrated in Figure 3.10.

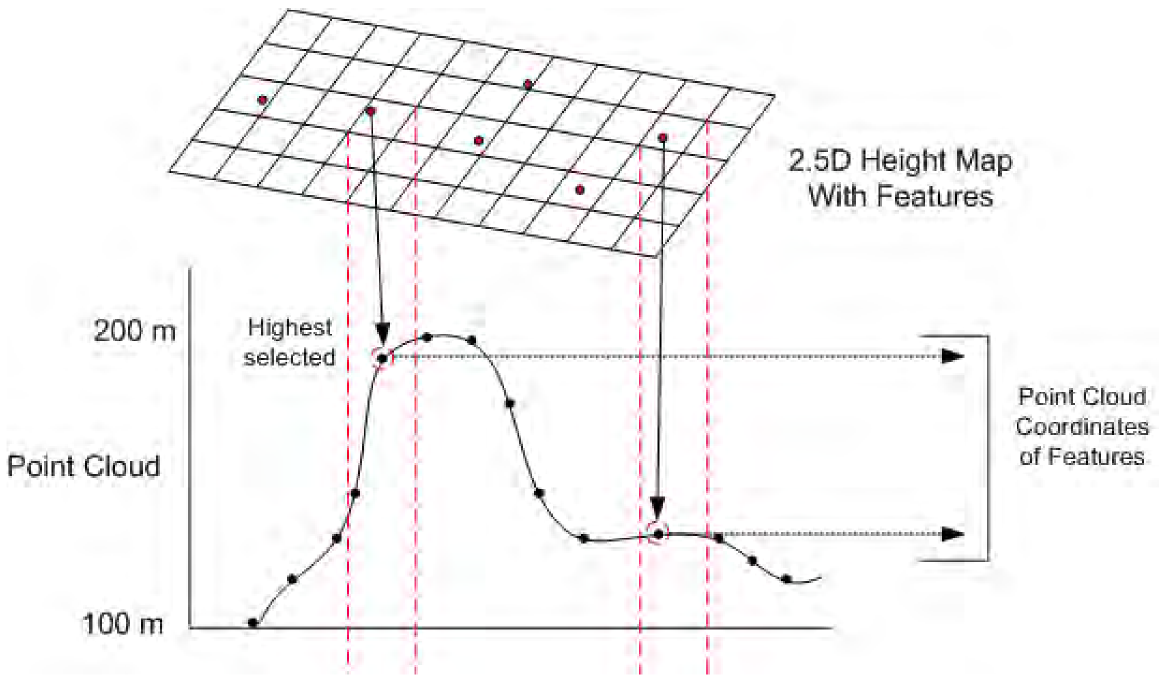


Figure 3.10: The projection of range features back to the original LiDAR point coordinates. If multiple points are within a raster cell containing the feature, the highest point is chosen.

Due to the rasterization process used to create the range images, some interpolation exists between the point cloud and the DEM. Another source of interpolation error is that

SIFT can place features between pixel centers. Both of these interpolations imply that the accuracy of the true feature locations is dependent on the density of the point cloud. The interpolations also create cases where the direct projection of a point cloud feature from its raster cell can result in multiple LiDAR points contributing to the same SIFT feature. In these cases, the LiDAR point coordinate with the largest height is selected, as first return LiDAR was used in the collected data set and to alleviate difficulties with breaklines similar to [112]. This step results in a set of 3D ortho-rectified LiDAR points from both the collected $\bar{\mathbf{a}}^*_{pc}$ and reference LiDAR $\bar{\mathbf{b}}^*_{pc}$ point clouds. Each LiDAR point pair has been associated with each other by the SIFT algorithm, and have similar correspondence transforms determined by the RANSAC algorithm. This gives a set of point cloud features

$$\bar{\mathbf{a}}^*_{pc} = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix} \tag{3.11}$$

where there is one point cloud coordinate for each of the n features, and similarly for $\bar{\mathbf{b}}^*_{pc}$. Furthermore,

$$\bar{\mathbf{a}}^*_{pc} \subseteq \bar{\mathbf{a}}_{pc} \tag{3.12}$$

$$\bar{\mathbf{b}}^*_{pc} \subseteq \bar{\mathbf{b}}_{pc} \tag{3.13}$$

and each point cloud feature in $\bar{\mathbf{a}}^*_{pc}$ has a similar correspondence transform to its match in $\bar{\mathbf{b}}^*_{pc}$ such that

$$\bar{\mathbf{a}}^*_{pc} \cong \bar{\mathbf{b}}^*_{pc} \tag{3.14}$$

within the RANSAC threshold.

3.1.8 Iterative Correspondence Calculation.

With these two feature point clouds, the correspondence transform that aligns the collected point cloud to the the reference point cloud can be calculated. Horn's algorithm

[35] is used to calculate the final correspondence translation and rotation that aligns the the collected point cloud to the reference. The inputs to the correspondence algorithm are the point cloud features associated with the SIFT features remaining after RANSAC, given as set $\overline{\mathbf{a}}^*{}_{pc}$, and their matching feature points in the reference point cloud, $\overline{\mathbf{b}}^*{}_{pc}$. The correspondence rotation is a DCM $\mathbf{C}_A^B$, and the translation an offset vector $\mathbf{t}$. Applying the transform gives

$$\overline{\mathbf{a}}_{fit} = \mathbf{C}_A^B \overline{\mathbf{a}}^*{}_{pc} + \mathbf{t} \tag{3.15}$$

where,

$$\overline{\mathbf{a}}_{fit} \approx \overline{\mathbf{b}}^*{}_{pc} \tag{3.16}$$

where $\overline{\mathbf{a}}_{fit}$ is the least squares fit of features $\overline{\mathbf{a}}^*{}_{pc}$ to features $\overline{\mathbf{b}}^*{}_{pc}$.

While the application of RANSAC does a good job of removing poor matches before this step, the projection of the features back into the original point cloud brings an element of variation back into the matched pairs, and even a single outlier can be disruptive to the final correspondence. In the cases where some outlier points were passed through RANSAC, there may be some match pairs that are hundreds of meters off. To identify and remove these poor quality points from the final set, the matching algorithm is applied iteratively with three threshold conditions.

First the 3D error magnitude is calculated for each feature input into the correspondence algorithm. The error of each feature is ϵ is determined by

$$\epsilon = \overline{\mathbf{a}}_{fit} - \overline{\mathbf{b}}^*{}_{pc} \tag{3.17}$$

and each feature 3D error magnitude is

$$\varepsilon = \sqrt{(\epsilon_x^2 + \epsilon_y^2 + \epsilon_z^2)}. \tag{3.18}$$

Next the magnitude errors ε of each feature pairing are sorted largest to smallest. This allows use of filter conditions on the largest error features. If the largest feature error is above a threshold set at α, then the presence of an outlier is assumed to be distorting the correspondence, the largest magnitude error feature is removed and the correspondence is recalculated. If the ratio of the largest magnitude error divided by the second largest is above a threshold set at β, then the largest error is assumed to have a significantly different correspondence then the other matched features, the largest error feature is removed and the correspondence is recalculated. Finally if after any iteration of the matching algorithm, the total number of features falls below a threshold set at γ, then this entire set of features is rejected completely. Once the correspondence has been calculated with the final set of features that meet all the condition thresholds, the final rotation and translation are output that best maps features $\overline{\mathbf{a}}^*{}_{pc}$ onto $\overline{\mathbf{b}}^*{}_{pc}$. This step concludes with a correspondence transform rotation DCM, translation, the set of collected point cloud features with the transform applied $\overline{\mathbf{a}}_{fit}$, and the set of remaining reference features $\overline{\mathbf{b}}^*{}_{pc}$.

3.1.9 Position and Estimation Calculation.

The previous steps determined feature points common to both the collected LiDAR and the reference LiDAR point clouds and solved for a transform to align the collected points with the reference. The final goal is to estimate the position of the aircraft using this information. Given that the features are derived from the LiDAR sensor and nominal aircraft position, it is assumed that applying the correspondence transform to the aircraft position will also align it with the true position. If the ground features are accurate enough, determined by the thresholds in Section 3.1.8, an accurate aircraft position estimation is possible, and the uncertainty can be further studied under these conditions. The algorithm concludes with the ground correspondence applied to the nominal aircraft position, giving an updated aircraft position estimation.

IV. Results

This chapter describes the experiments performed on collected flight data to test the performance of the algorithm proposed in Chapter 3. The first section covers the experiments data sets in two parts; the collected point cloud, and the reference point cloud. Particulars of the region covered by the data sets are also mentioned. The first section also mentions how collected data was used to calibrate the LiDAR sensor parameters. The last section details the overall results of applying the proposed algorithm to the collected data. The results are broken up into two parts; the main cases with different initial conditions, and a sensitivity analysis of the various thresholds used to ensure good results.

4.1 Experiment Details

This section details aspects of the experimental data. The data is covered in two sections, the All-Source Positioning and Navigation (ASPN) data set, and the Ohio Statewide Imagery Program (OSIP) data set. Both data sets are located within the Athens county region of southeast Ohio, USA. Athens county is generally 220 meters above sealevel and with the exception of two small cities, the region consists mostly of forested and agriculture use terrain.

4.1.1 ASPN Details.

The collected point cloud used in this research is from an experiment flyover preformed by Ohio University and part of the ASPN data collection was lead by the Autonomy and Navigation Technology (ANT) center of Air Force Institute of Technology (AFIT). The flyover recorded data on a DC-3 [7] equipped with a variety of sensors, including INS [34], GNSS, and scanning single return LiDAR [80]. The experiment used a differential GNSS\INS solution to produce truth position and orientation data over the course of the flyover, and the specific intrinsic and extrinsic calibration parameters were determined as part of the ASPN program. The experiment is available as a collection of raw sensor measurements and must be processed to place them into a point cloud format. The raw data consists of each distinct LiDAR range and fire angle measurement, the true position and orientation of the aircraft, all coordinated with a GNSS time stamp. Some linear interpolation is required to ensure each LiDAR sample is matched to INS and time measurement. The ASPN point cloud is divided into smaller swaths for use in creating the individual range images. The flyover starts near takeoff and levels out at an altitude of approximately 720 meters. The LiDAR scans the ground with a 45 degrees view angle, producing average swaths approximately 400 meters wide on the ground with an average point density just under 1 point per meter square, although the exact size and density of a swath changes depending on the altitude and velocity of the aircraft.

To test the proposed algorithm, an ASPN swaths was created every 10 seconds over the course of the 11 minute flyover. In the tests, each swath was processed at a nominal aircraft position provided by the GNSS truth with an added position offset. The LiDAR points included in the swath are the 7.5 seconds of scanning time before and after the nominal aircraft position, making each swath 15 seconds long, overlapping each other, and containing approximately 180000 LiDAR samples. Over the course of the flyover, this generates 66 swaths that are later processed into range images by the proposed algorithm. While other sized swaths were explored, larger swaths up to double the size of the OSIP regions didn't significantly impact the overall results. Smaller swaths reduced the number of completed position estimations at the conclusion of the algorithm.

4.1.2 OSIP Details.

Use of LiDAR data for the creation and improvement of mapping is becoming more common. Online repositories such as the OSIP [74] contain LiDAR and corresponding image data of the ground from aircraft flyovers. This data can be of great use to municipalities for infrastructure modeling, zoning, and land use [39, 62, 93, 119]. OSIP is commissioned to collect imagery data of all of Ohio and includes LiDAR scans as precision references to the more widespread photographic imagery. These LiDAR point clouds are organized by county and are further broken down into roughly 1.5 square kilometer tiles with a sample spacing of one LiDAR ground point per 2 meters square. Possibly due to this being multiple return LiDAR, the average density appears to be higher than advertised. The point cloud of each tile is available in a .LAS file type [15]. The LiDAR data has gone through significant postprocessing including filtering, calibration, and strip alignment in conjunction with the other collected imagery [82, 97]. Because of this postprocessing, the OSIP data is considered an excellent reference set. The navigation frame that the OSIP data is registered in is NAD83, specifically the south Ohio state plane zone 3402. This places the OSIP point cloud coordinates in the SPCS, which is a ENU system with special origin requirements.

It is important to note that the tile system provided by OSIP was used directly for the reference regions matched with the ASPN range images. Each nominal aircraft position was compared to a list of tile SPCS coordinates, and the selected OSIP region was used for the rest of the algorithm. This places two constraints on the algorithm: the area involved in matching the SIFT features is limited to one OSIP tile, and the possibility of the ASPN swath landing on tile boundaries. In this algorithm, combining tiles to make larger search areas is not further pursued. A swath crossing a tile boundary or corner happens frequently in the ASPN flyover data, and it is flagged when less then 50% of a swath is included in the selected tile. Of the 66 swaths collected in this experiment one third crossed tile boundaries and were flagged in this way. Of those, one third (one ninth of the total swaths) were still able to solve for the aircraft position. Flagged swaths or swaths that contain a water feature are somewhat at a disadvantage, as they generally contain significantly less LiDAR data useful in matching to the chosen reference tile.

4.1.3 Terrain Observations.

Observation of the OSIP areas relevant to the ASPN flyover gives the following generalization of the terrain:

Table 4.1: Terrain types present in the flyover area.

	Agriculture	Forest	Road	Water	City
Number of Mixes When Observed	43	45	15	10	10
Percent of Total	65%	68%	23%	15%	15%

In Table 4.1, the frequency of occurrence is a simple count of the terrain types present in a given flyover region and the percentage is that count out of the 66 ASPN swaths processed over the course of the flyover. Multiple terrain types are present in most of the regions, hence the percentages are well over 100%. Agriculture use terrain is counted as the presence of fields, sparse and scattered structures, and manmade tree line borders. Forested terrain is counted as areas with few if any structures and a dominance of tree canopy. City terrain was counted primarily for the areas that flew directly over the city of Athens itself, and are noted as a collection of densely packed structures with many small roads. Roads were counted if the flyover region included a large, obvious road, such as a US highway.

Water in some form was present in many of the flyover regions. Water is a special terrain type in this experiment as both data sets did not utilize a LiDAR capable of calculating a range measurement over water (the LiDAR beam is absorbed instead of reflected). As there are no range measurements on water regions, it is the absence of range measurements in the shape of rivers or lakes in low areas that define the presence of water. Water is therefore counted as a terrain type if enough was present to cause interpolation errors over the water area in the resulting range images, representing an area with no actual

LiDAR points. There are certainly more areas in the flyover that contain small rivers or shallow livestock ponds then are counted as containing water. These uncounted areas do not produce the interpolation errors normally observed and are assumed sufficiently shallow water to permit a detectable LiDAR range measurement, to have vegetation on the surface, or vegetation cover over them.

Time of collection is also a factor between the two data sets. The OSIP LiDAR data in Aspen county was collected approximately 4.5 years before the ASPN data collection effort. This difference in time leads to some small portions of the resulting point clouds that have significantly changed between collections. Some observed differences include new building construction, water level and shoreline changes, and vegetation changes. The observed differences in vegetation, specifically trees, includes some minor new growth and cutting changes, but very a prominent change is the season. The OSIP data was recorded in the mid to late spring months, while the ASPN data was recorded in late fall. What is specifically being observed is the OSIP data shows full, leafy tree crowns, while the ASPN data shows considerably fewer tree crowns, possibly because the trees are not full of leaves.

Overall, the algorithm generated the most number of features in the city type terrain. The numerous square corners and rooftops over roads appear to generate SIFT features easily. Large roads and agriculture use land were also consistently good at generating numerous SIFT features, particularly the scattered buildings, but also on the forested borders of fields, and local elevation peaks or valley bottoms. Heavily forested terrain generated the smallest number of features, and were the most frequently unable to produce an aircraft position solution. This was at least in part due to the significant seasonal vegetation density differences observed. There were some forest exceptions that performed very well. Forested images also containing valleys or peaks that were not as densely covered in vegetation, and large clusters of trees that were significantly taller than the surrounding canopy both generated many SIFT features. Some additional detail about each algorithm match in the data set are given in the appendix, Chapter A.

4.1.4 Calibration.

While comparing early results, it was observed that the collected and reference point clouds were misaligned by up to 20 meters, even when using the truth solution for the ortho-rectification. Some simple tests showed that this alignment error was dependant on the aircraft heading, indicating that the LiDAR was not aligned with the body frame of the aircraft exactly as assumed in Figure 2.21. This is consistent with a bore-site misalignment of the LiDAR with respect to the aircraft body as defined in Section 2.4.5. There are several methods to perform a bore-site correction given in [93], but most require multiple overlapping swaths to utilize. In this case, the raw LiDAR measurements can be used directly in a fashion similar to the [30] quasi-rigorous method. Roof planes on buildings are commonly used to perform bore-site correction methods, so a correction set of 16 small swaths was hand selected over the course of the flyover, each containing a single building with little or no vegetation around it. This generally gives a clear roof plane well above a flat noise free ground plane common in both point clouds. A brute force planar correlation search was used to align the two point clouds and determine the offset found in each swath in SPCS coordinates. Ortho-rectification from SPCS back into the body frame, allows for a roll and pitch correction DCM to be calculated based on the average offset found in the correction swaths. Applying the correction DCM to the LiDAR sensor measurements in the sensor frame, effectively corrects for the bore-site misalignment. After calibration the result is the ASPN and OSIP point clouds are on average aligned to within approximately 2 m^2 (this is also the approximate density of the reference LiDAR ground points).

4.2 Experiment Results

The application of the proposed algorithm to the collected ASPN point cloud swaths and the OSIP reference point cloud tiles functions well utilizing the GNSS to update the INS nominal aircraft position. To test the effectiveness of the algorithm in a more uncertain scenario, as if the INS had no GNSS updates and has some accumulated drift error, additional error was injected into the ASPN raw sensor measurements. This section discusses the results of the experiments with errors added to the INS position and explores the impact of that error on the final aircraft position errors and sensitivity to the threshold parameters.

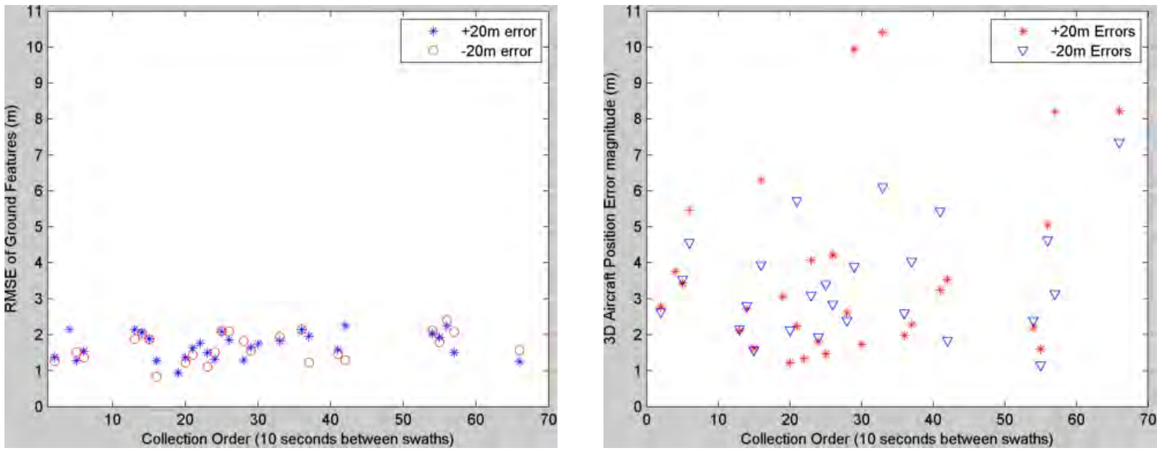
4.2.1 Impact of Initial Conditions.

The impact of error in the INS nominal aircraft position was tested in two initial condition cases, one with a positive 20 meter error in LLh, and the other with a negative 20 meter error. The original truth data was then used for the calculation of the resulting error metrics. The added errors are not cumulative over the course of the flyover, they are added as a given swath is ortho-rectified into the SPCS so that the results of an earlier swath match do not impact the subsequent swaths. The thresholds selected for use in this experiment were

$$\begin{gathered}\text{Maximum ground feature error magnitude, } \alpha = 4 \text{ meters}\\ \text{Maximum ground feature error magnitude ratio, } \beta = 1.15\\ \text{Minimum number of features required, } \gamma = 8 \text{ features.}\end{gathered} \tag{4.1}$$

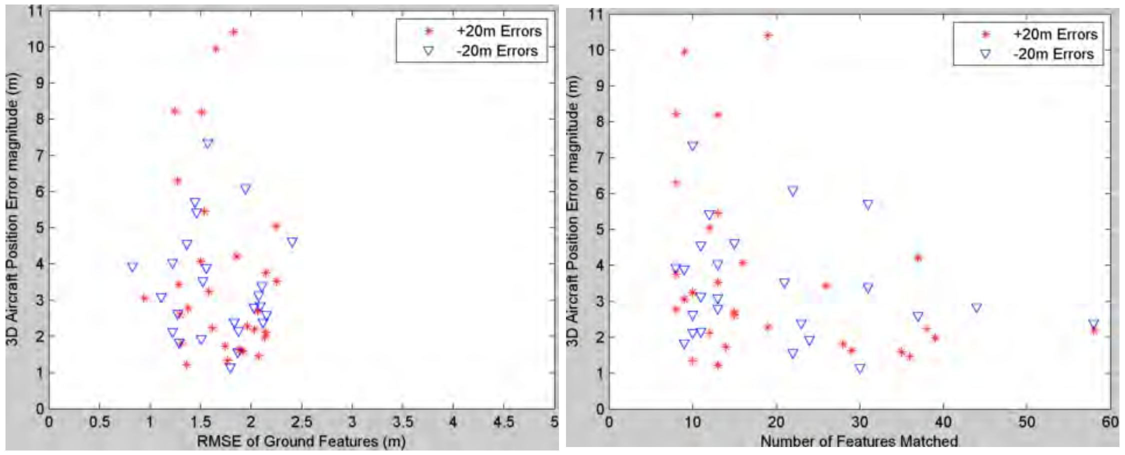
These thresholds were chosen specifically to limit the resulting aircraft position errors from being greater than approximately 10 meters, and are explained more in Section 4.2.2.

The Root Mean Squared Error (RMSE) of the ground features for each swath is shown over the course of the flyover in Figure 4.1 (a). Figure 4.1 shows that under both initial conditions the total RMSE error in matching the collected swath ground features to the reference tile ground features are about the same, around or below 2 meters. This is noted as being similar to the ground density of the reference point cloud. In Figure 4.1 (b) is the 3D aircraft position error magnitude for each swath over the course of the flight.



(a) (b)

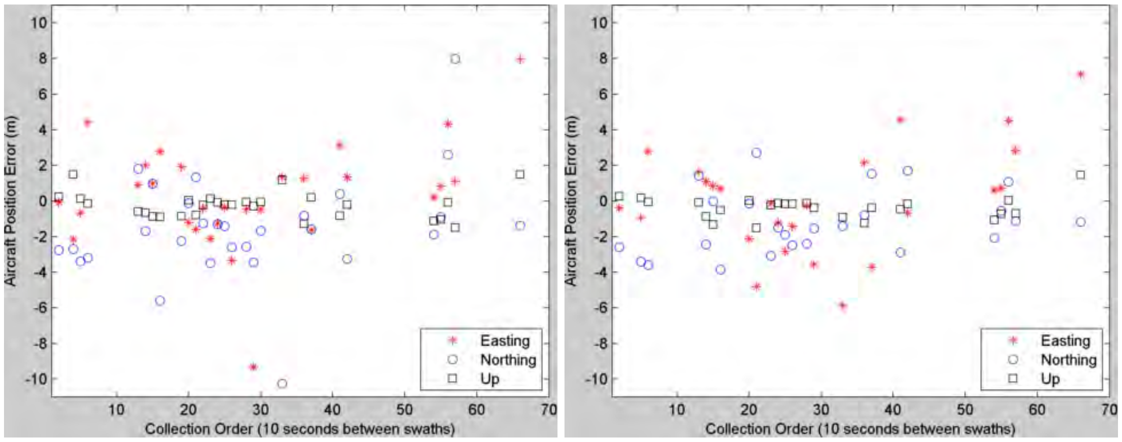
Figure 4.1: (a) RMSE of the ground features and (b) the 3D aircraft position error magnitude of each swath over the course of the flight.



(a) (b)

Figure 4.2: The calculated 3D aircraft position error magnitude in relation to: (a) the matched ground feature RMSE; (b) the number of features used in the calculation.

In Figure 4.2(a) the relationship between the ground feature RMSE and the resulting aircraft position is easier to see. The ground feature RMSE needed to keep 3D aircraft

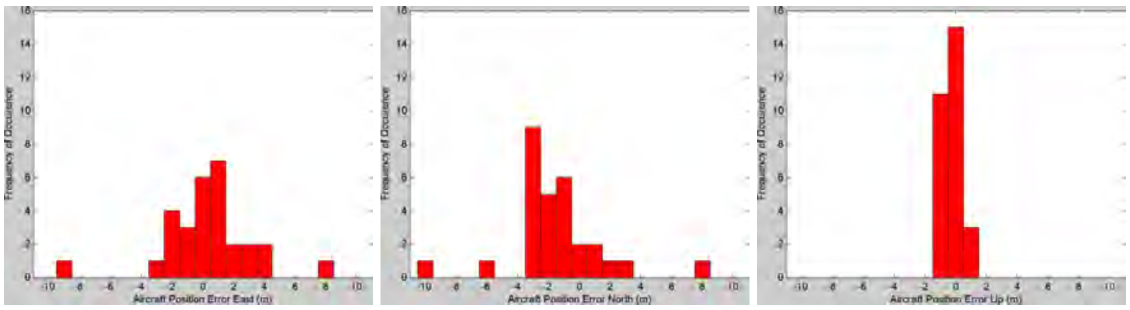


(a) (b)

Figure 4.3: Aircraft absolute errors in ENU for: (a) the +20 meter and (b) the -20 meter error cases

position error magnitudes below 10 meters appears to be approximately 2.5 meters or less. Another prevalent factor on the 3D aircraft position error magnitude is the number of ground features used to solve for the correspondence transform, shown in Figure 4.2(b). The γ threshold used ensures that only matches that have 8 or more features available are used to calculate the 3D aircraft position error magnitude. It is clear to see a trend that as more features are available, the resulting 3D aircraft position error magnitude approaches the ground feature RMSE. Unfortunately in these tests, two thirds of swaths created have less than 20 features after the RANSAC and threshold portions of the algorithm, and so increasing the γ threshold to be more exclusive will quickly reduce the frequency that an aircraft position can be calculated.

The aircraft absolute errors in each ENU axis are shown in Figure 4.3. It is easy to see that the absolute errors in the aircraft position estimation are approximately zero mean in all three coordinate axis. There also appears to be very little difference between both initial condition cases. The up axis in particular has noticeably less variance than the east or north



(a) (b) (c)

Figure 4.4: Histograms of the aircraft absolute errors in the (a) easting, (b) northing, and (c) up axis.

axis. An estimation of the aircraft position error probability distribution function can be seen in Figure 4.4. Because the results of both the initial condition cases are quite similar, only the +20 meter error case will be further discussed.

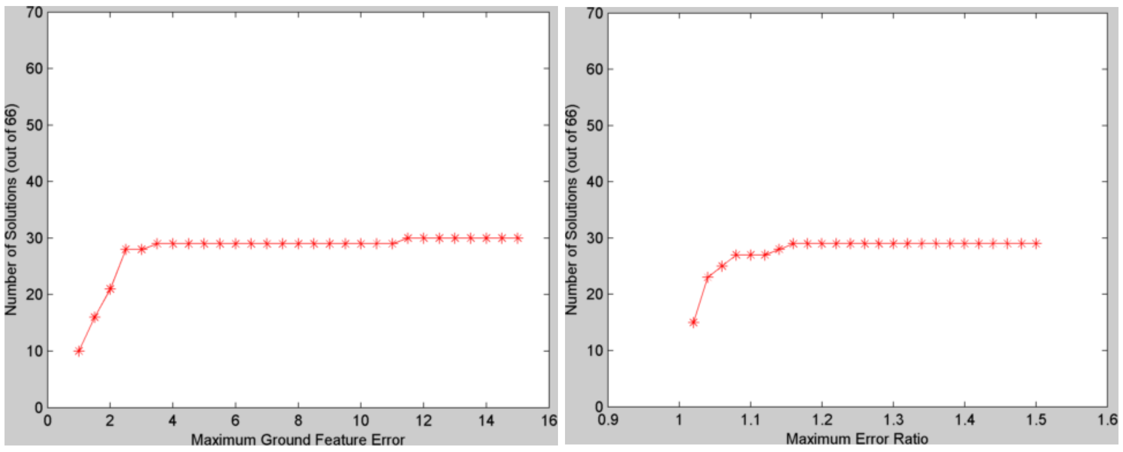
Figure 4.4 shows the histogram of the aircraft errors in each axis using one meter bins. The grouping of all three histograms suggests a gaussian or similar distribution of errors with the up axis being considerably less dispersed than the east or north axis. With this limited data set, histogram (b) appears to suggest a slight southward bias in the north axis. This could possibly be attributed to a slight bore-sight offset still existing after the calibration effort, but the exact cause has not been explored.

Overall, the impact of changing the initial conditions appear minor. The RMSE of the ground features are approximately the same for each swath. The error magnitudes and the absolute errors of the aircraft position are similar in both cases with some small variations. The larger variations result from how the OSIP tile is chosen for matching with a given swath. In this data set about one third of the swaths were generated on a OSIP tile boarder or corner, and several in the -20 meter additional error case caused a different tile to be selected then in the +20 meter case. These instances create much different matching conditions between the cases, and is the reason there are 4 less successful aircraft position estimates in the -20 meter error case. Most of the remaining swaths have small, one or two meter, absolute error differences in one or more axis between cases. A close look at each match shows that these swaths were matched to the same OSIP tile in both cases, but have small differences in the number of features, and more interestingly the exact features themselves. It appears that most of the features that survive RANSAC are common to both cases, but will include a few new or exclude a few features between the two cases. This appears to be related to the random selection process that RANSAC uses to form consensus sets, and the iterations do not exhaustively use all available features, it stops when the threshold is reached. It appears that these traits of the RANSAC algorithm are responsible for the observed variability in the aircraft absolute errors.

4.2.2 Sensitivity to Thresholds.

The three threshold parameters in Equation (4.1) used in these tests were selected to constrain the resulting aircraft position errors to approximately 10 meters or less. The particular parameter values chosen for the results previously shown were selected as the best tradeoff possible in terms of low aircraft position errors, high number of aircraft position estimates, and a minimum of time between aircraft position estimates. To better understand the impact the threshold parameters were having on these results, each parameter was varied through a range of values while the remaining two were held constant as selected in Equation (4.1).

Figure 4.5 (a) shows the impact varying parameter α. It shows that accepting a maximum error in any individual ground feature larger than 4 meters does not generally improve the total number of swaths that produce a position estimate. Similarly in Figure 4.5 (b), increasing parameter β, the acceptable ratio of the largest ground feature errors, above 1.15 also does not improve the total number of swaths that produce a position estimate. In



(a) (b)

Figure 4.5: The number of swaths that resulted in a position estimation in relation to changes in (a) threshold parameter α and (b) threshold parameter β, while the other thresholds are held constant.

both cases, making the threshold more strict by setting the parameter values to lower than used in the results sharply reduces the the total position estimates.

In Figure 4.6, the minimum number of feature matches required to produce an acceptable position error estimate, γ, is varied. It is shown that as more and more features are needed to make a acceptable match, there are fewer swaths available that can produce enough SIFT features than that meet the RANSAC and threshold requirements. This is similar to the trend found in Figure 4.2 (b), where the 3D aircraft position error magnitude is shown with the number of features used in each swath. It also suggests that simply accepting matches with fewer features will significantly increase the number of aircraft position estimates.

Conversely, accepting matches with very few features should reduce the accuracy of those matches. These trade offs are illustrated by varying the threshold parameters against the aircraft position errors averaged over the entire flight. To show this, the Mean Radial

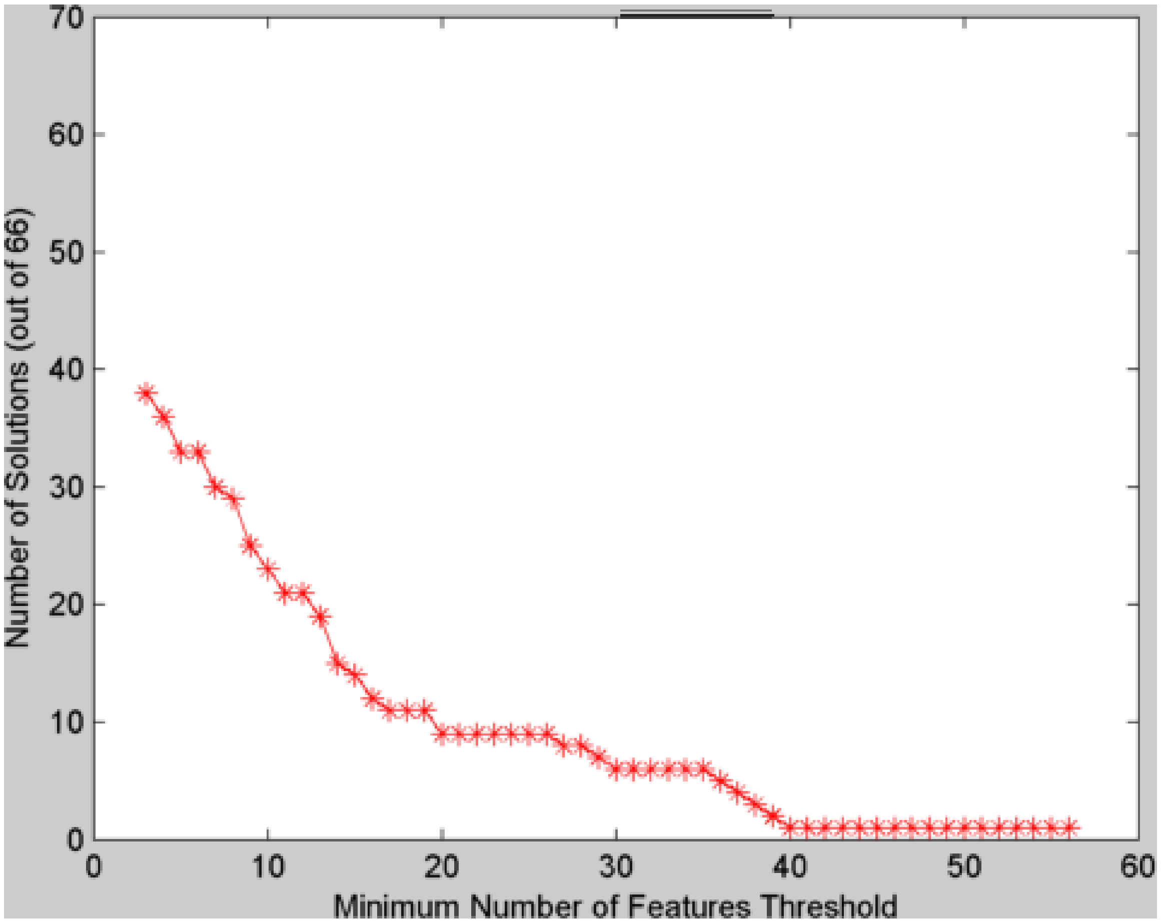
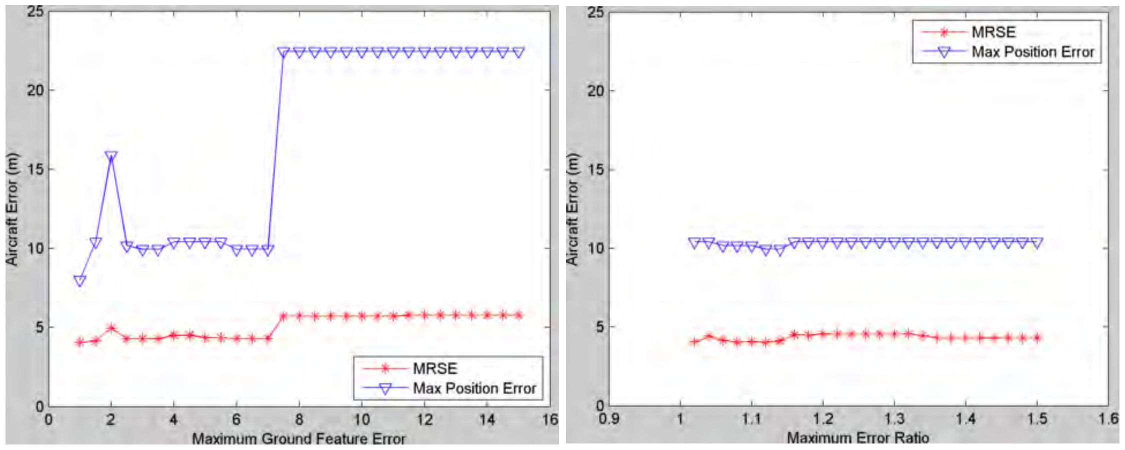


Figure 4.6: The number of swaths that resulted in a position estimation in relation to threshold parameter γ. Threshold α was held at 4 meters and β was held at 1.15.

Spherical Error (MRSE) metric is used

$$MRSE = \sqrt{\frac{\sum_{i=1}^{n}(x_i^2 + y_i^2 + z_i^2)}{n}} \tag{4.2}$$

where i is the aircraft position error estimate of each swath, and n is the total number of swaths in the flyover. Figure 4.7 and Figure 4.8 show the variation of the threshold parameters against the flight MRSE. Figure 4.7 (a) shows that other than a small but sharp increase in the MRSE if 8 or more features are used in calculating the correspondence transform, there is fairly minimal impact by changing parameter α. Changing parameter β,



(a) (b)

Figure 4.7: The MRSE of all the position estimates and the largest position error magnitude of each flyover in relation to changes in (a) threshold parameter α and (b) threshold parameter β, while the other thresholds are held constant.

in Figure 4.7 (b), has almost no meaningful effect on overall MRSE at all. This seems to put most of the importance in choosing parameters that generate the most total number of position estimates as shown in Figure 4.5 and Figure 4.6.

Varying γ shows a similar trend in the MRSE as it does in the total number of position estimates, shown in Figure 4.8. In Figure 4.8, the maximum individual 3D aircraft position error magnitude of all the estimates are also displayed to help show the tradeoff. Here, the trend is that the overall accuracy in terms of MRSE improves steadily as the minimum number of features required to calculate the correspondence transform is increased, to a minimum close to the reference point density. But, as shown in Figure 4.6, increasing the number of features required also reduces the total number of position solutions available, the two terms are in direct tradeoff. In addition, accepting solutions that use fewer features not only gradually increases the MRSE, but it greatly increases the largest individual 3D aircraft position error magnitude in that flyover case. To maximize the availability of

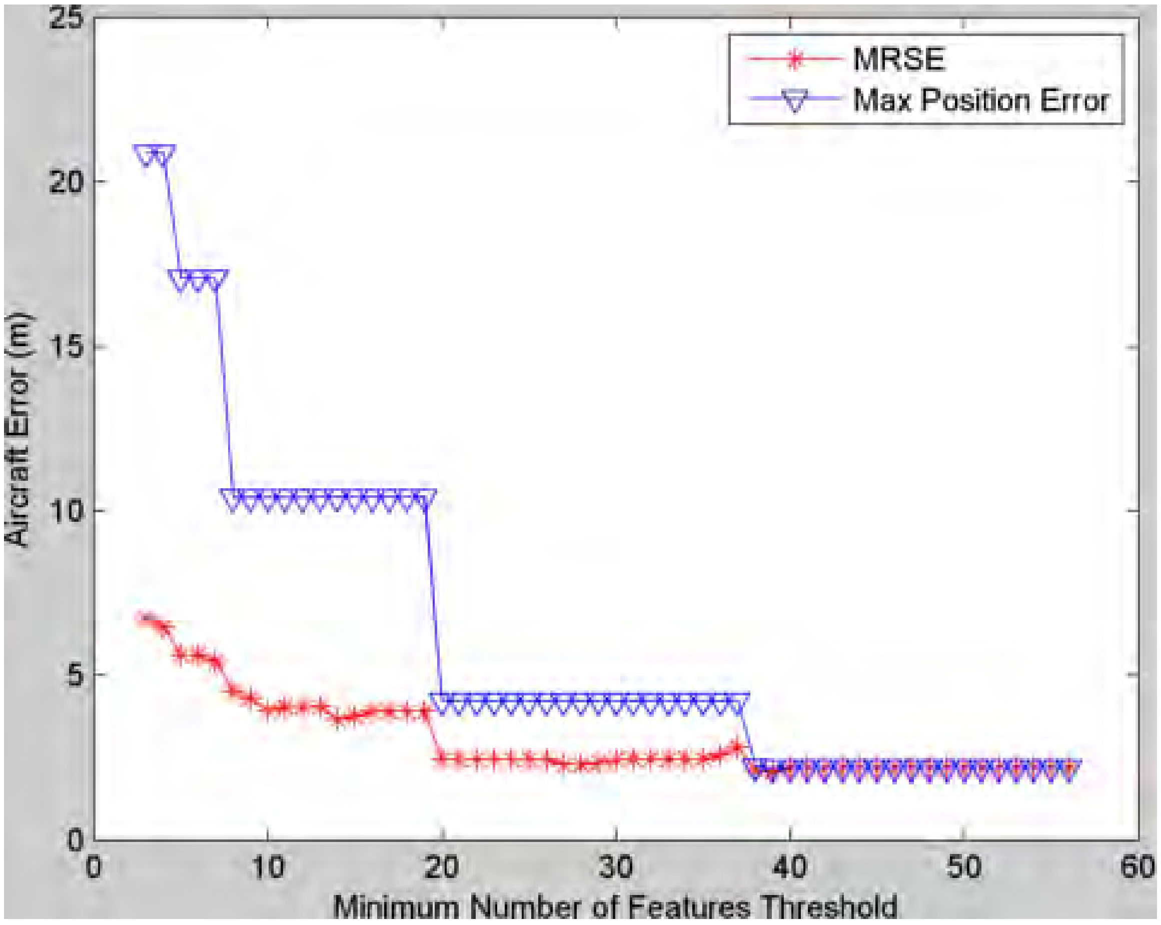
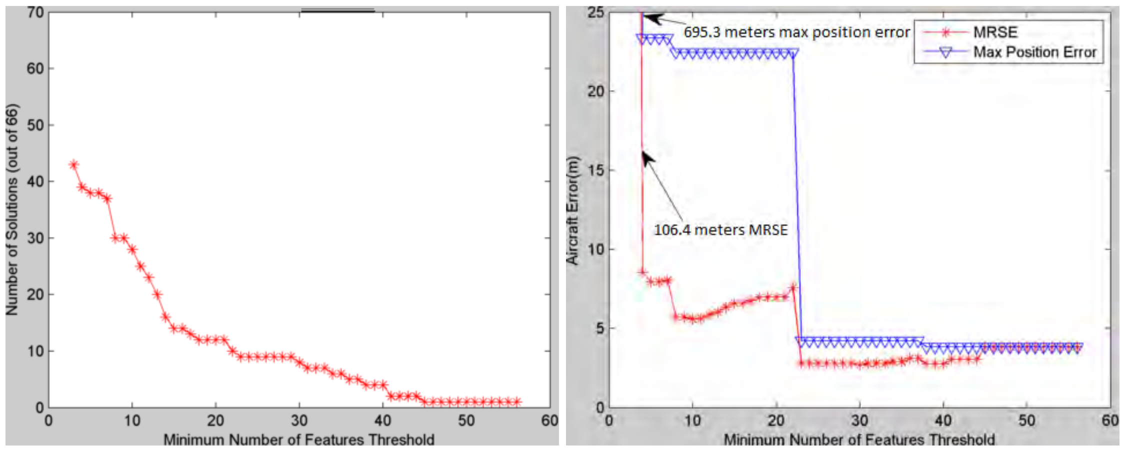


Figure 4.8: The MRSE of all the position estimates and the largest position error magnitude of each flyover in relation to threshold parameter γ. Threshold α was held at 4 meters and threshold β was held at 1.15.

position solutions while keeping the worst position error estimates below approximately 10 meters RMSE, a γ threshold of 8 features appears to be an acceptable choice. If less accuracy is acceptable in favor of more frequent position estimate availability, then this threshold could be lowered.

Care must be taken choosing threshold values, for example the trends given in Figure 4.8 by varying threshold γ are only valid when thresholds α is held constant at 4 meters, and threshold β is held constant at 1.15. It was observed that larger values of α and



(a) (b)

Figure 4.9: Example where threshold α is held at 50 meters and threshold β is held at 2, while threshold γ is allowed to vary against (a) the number of swaths that resulted in a position estimation and (b) the MRSE of all the position estimates in the flyover, with the flyover largest position error magnitude.

β would increase the occurrence matches but with much higher 3D aircraft position error magnitudes. An example of this is shown in Figure 4.9, where threshold α is increased to 50 meters and threshold β is increased to 2, but threshold γ is varied. Figure 4.9(a) shows that increasing the threshold values increase the number of solutions found over the course of the flyover. But Figure 4.9(b) also shows that the overall MRSE and the maximum 3D aircraft position error magnitudes are increased significantly. This example implies that while increasing the thresholds will allow for additional position estimations, the quality of those estimations may be significantly reduced. In addition, the larger maximum position error magnitudes imply that some of the individual position solutions can be hundreds of meters in error under these thresholds.

Availability of position estimations is also a success metric. If the thresholds are too stringent, there could possibly be large time outages from the last position estimate, thereby allowing the INS to acquire too much drift error. Varying all three threshold parameters and measuring the change in the maximum outage showed that the thresholds have little or no impact on this success metric. As implied in Figure 4.1 (a), the largest position estimate outage is between the 43rd and 55th swaths. That outage generates approximately a 120 second gap between the last position error estimation and the next in this flyover. The maximum outage only increases when the threshold parameters γ becomes so large that there are very few position error estimations at all. The region of the flyover concerned with the maximum outage is over the city of Athens itself which would normally be a very rich area for SIFT features. This lack of features is partially explained by being part of a long period of time were the collected ASPN swaths were continually over corners and edges of the selected OSIP reference tiles, and also frequently contained a large water feature (a river). It is suspected that this particular outage, and related scenarios, would be greatly reduced if the method for reference tile selection was not restricted to a single tile, thereby reducing the size of the outage.

Overall this analysis of the threshold parameters gives a few impressions. First is that the three parameters at the current settings are restrictive enough to ensure that the errors in the aircraft position estimates are below 10 meters error magnitude for this data set. Second is that variations in the threshold parameters apply control to two key metrics, the desired quality of features after RANSAC, and the total number of features required to obtain a certain level of accuracy. Depending on the requirements of an application, the thresholds could be loosened or tightened as needed. Any outliers present in the set of features used in the final correspondence transform have an noticeable impact on the resulting aircraft position error, and great effort should be used to detect and remove them. The only observed exception to this case is when there are dozens of features with only a few minor outliers. In these cases the many outweigh the few and the resulting aircraft position error in not significantly impacted. As for the number of features available, more is better in terms of minimizing errors in terms of MRSE and RMSE, minimizing time between position estimates, and maximizing the frequency with which a position estimate is achieved.

V. Conclusions

This thesis presents an algorithm that is able to calculate a aircraft position with an INS and a single scanning LiDAR sensor with no true GNSS requirement. The calculation is performed by utilizing LiDAR ranging information in combination with range-based SIFT features and other computer vision techniques, and a LiDAR based reference of the flyover region. This chapter discusses the conclusions of this research and describes areas noted for future work.

5.1 Summary of Results

The goal of this research was to demonstrate that using LiDAR range information could be used to produce position estimates for navigation using only one LiDAR sensor, an INS, and reference data, without the direct use of GNSS information. The proposed algorithm was successful in producing position estimates with 3D position errors approximately 10 meters or less. The amount of error was constrained through the use of three threshold parameters representing the maximum error between any ground feature and its reference match (α), the maximum ratio between the two largest matched feature errors (β), and the minimum number of features needed to calculate a correspondence transform (γ).

By adding both a positive and negative 20 meter error to the INS latitude, longitude, and height measurements, the sensitivity of the algorithm to variation in the initial conditions was tested. The results show that while changing the initial conditions does introduce some small variation in the resulting ground feature and aircraft position errors, the results in both cases are nearly the same. In addition, the variations observed are directly related to the method of selecting a single reference tile and the occurrence of collected LiDAR swaths to be located on the borders of those tiles. These tests show that the impact of at least 20 meter INS position errors in any or all directions has no substantial impact on the ability of the algorithm to produce low error position estimations.

A histogram of the absolute aircraft errors in each coordinate axis was also explored. The histograms implied that the aircraft position errors are generally zero mean, though there is a suggestion of a small southward bias. Most of the errors found in the aircraft position appear to be in the easting and northing directions, while the up axis has markedly less error and less error variance. The histograms also imply that errors in each axis could be fit to a Gaussian distribution, but further research is needed to draw stronger conclusions of the true error distributions.

The second set of tests explored the sensitivity of the algorithm to variation of the threshold parameters. The best selection of these thresholds for this data set were used to constrain the results to meet error goals, but also showed that different levels of accuracy in terms of MRSE and frequency of position estimates (availability) can be varied to suit particular application goals. The results also showed that in this algorithm the relative accuracy and availability are inversely related, creating an engineering tradeoff between desires for low MRSE and high availability.

In summary, this research presents an algorithm that can perform position estimation for use to update an INS by utilizing LiDAR range data without a GNSS receiver. The algorithm can provide these updates as the aircraft flies over a region contained within the reference data region. Despite the success of the algorithm in keeping position errors below 10 meters, there are several observations made or research areas available to improve or expand the capabilities of this algorithm. These are addressed in Section 5.2.

5.2 Future Work

Overall the algorithm performs well as a proof of concept. But a great deal of work remains to refine the algorithm or expand its capabilities. Several observations were noted when working with point cloud data and reading the literature, each leading to ideas that were not explored in this research. This section attempts to discuss the majority of these ideas.

The first area for possible improvements is in the algorithm itself. The algorithm is chiefly dependant on the number of SIFT ground features that survive RANSAC and the threshold conditions. Generally, any improvement that would help to detect more range features in both the collected and the reference range data would be beneficial, but it is suspected that this can be approached in several ways. In order of the algorithm steps, some suggested improvement are mentioned.

The OSIP reference data is precalculated and conditioned off-line, which improves speed of computation, but a great deal more could have been done. By far the most computation intensive steps in the algorithm are the creation of the range image and the detection of the SIFT features. Ideally, the reference data can be preprocessed into only what is required for the algorithm, specifically just a list of the SIFT features present, the coordinates of the features, and grouped by tile. Using just the minimum reference data required for the algorithm also has the advantage of storing only thousands of SIFT features instead of the often millions of LiDAR data points.

There are several accepted ways to project LiDAR points into a DEM, and each has an array of options for interpolating the space in between each LiDAR sample. The three most common were discussed in this thesis, and a linear interpolation with rasterization was utilized. This was convenient to use with the MATLAB software the algorithm was built and tested in. By far the largest consumers of LiDAR data are those interested in highly accurate mapping of the ground and those users frequently utilize GIS software with TIN based images. Adapting this algorithm to use TIN products might allow additional sources of reference data and processing software found in that industry. There are other proposed DEM creation methods, and they should be further explored with the hopes of improving calibration efforts and overall accuracy in matching.

The process to scale the range image into intensity values that SIFT will be able to process could be improved in several ways. The intended goal was to create range images that would have consistent gray levels based on the relative differences in elevation. The gray scale was achieved with the tradeoff of each swath is scaled based on the local minimum and maximum elevations. A consequence to local scaling is that simply adding or removing a scan line from a swath can introduce a new local maximum or minimum, and therefore create a DEM with significantly different scaling between elevation and gray levels. In addition two OSIP tiles were found to contain small clusters of erroneously extreme elevation points, causing the scaled DEM to be biased to very high or very low gray scales. Scale biased images are potentially very poor conditions for using SIFT as the features are only somewhat resistant to what is effectively an illumination change in the range image. Efforts to detect and eliminate extreme elevation values, or to create a normalized elevation scale between the collected and reference elevations should improve the ability of SIFT to generate good features.

Use of the RANSAC algorithm for outlier removal in SIFT features is well documented in the literature. Unfortunately RANSAC is much less frequently documented for use with true 3D data and the particular RANSAC algorithm used in this research was originally for planar homography. Homography works well for matches contained between two overlapping planes as is done in this research, but care must be taken to not introduce errors by including aspects of a camera model, which homography was designed for. Also, the most basic default settings were employed with the RANSAC algorithm, leaving room for improved performance over the current implementation. It was also observed that the use of RANSAC may be too early in the algorithm, requiring further outlier removal by iteratively applying the threshold conditions when calculating the final correspondence transform. Iteratively applying Horn's algorithm seems a somewhat redundant use of outlier removal steps. Instead, it would be interesting to simply take all the descriptor matched SIFT features and project them to their respective LiDAR ranged points, then perform a 3D RANSAC outlier removal step. This has the potential to both improve the number of features, reduce the number of outliers that survive RANSAC, and drop the iterative nature of the correspondence calculation.

As mentioned in the results, there are some serious performance limitations on the algorithm dependant on how the reference tiles are organized and selected. Changing this portion of the algorithm to allow concatenation of OSIP tiles alone would allow considerable improvements in the availability of the algorithm. It would also allow much greater initial errors to be present in the INS, and the potential search areas could be larger.

It was observed that there is some dependance of the algorithm on matching small scale SIFT features. The smaller features are in part due to the size of the collected swaths are much smaller than the size of a given reference tile. Reference tiles are then free to generate large scale SIFT features that simply can't be detected in the smaller swath data. This is perceived as a loss of possibly very useful large scale features, features created by hills, peaks and valleys, as opposed to small buildings and individual trees. To include large scale features, a ALS system could be set to scan much wider areas, though generally this would be at a cost of point density. Another way to support large scale features, is to filter LiDAR point clouds to include only the bare earth points, effectively removing vegetation and buildings. Bare earth filtering is frequently done in mapping activities and there are many methods available to implement bare earth point clouds in the literature.

The computer vision field has a very large selection of feature generating algorithms, some of which may be useful when used with LiDAR range-based images. There are also several different versions of SIFT available in the literature, and it is also possible that SIFT could be optimized to LiDAR range images. Other more advanced algorithms that can classify and segment point cloud data may also be useful in creating unique features for matching. A particularity interesting area for future research related to these concepts are geomorphic features. Usually for use with bare ground data, geomorphic features are fully 3D and not attributed to planar images or intensity values like SIFT features. Instead, geomorphic features are derived from 3D point clouds and their relation to neighboring points in the point cloud. Some geomorphic features that were present in the data used in this research were water networks, ridge networks, and channel heads. Ridge networks represent the local elevation peaks, hill tops, or ridge lines. Water networks are the opposite of ridge networks, representing the lowest elevation areas and valleys, but also the routes that water would take coming down from the ridges [3, 17, 61, 78, 86]. Channel heads are noted as the transition locations where a hill slope ends and a valley runoff begins [105]. One geomorphic approach uses a novel classification algorithm detects land types as Geomorphons [38]. Geomorphons can determine as many as 498 geometrically common landforms through the use of a scale and rotation invariant feature [75]. All geomorphic features have the potential to detect unique locations in LiDAR point clouds directly without the need to produce images.

There are some curious observations involving traits of SIFT features and how the algorithm processes images with respect to LiDAR range data. SIFT is scale invariant and this property is derived from the DoG process and the blurring effect it creates as it progresses though each octave on an image. By it's nature, ALS data is made up of scattered points and must be interpolated into an image for use with SIFT. Interpolation in this way is a form of image blurring. Also, Horn's correspondence algorithm is able to solve for the translation, rotation, and scale that least squares fits two data sets. Applying scale changes was not pursued in this research because it was unknown what exactly that would change with respect to LiDAR ranged, and orthorectified ground features. Scale changes on the other hand do create a better fitting correspondence transform, and when scales where calculated, they were always 1 or very close it (e.g; 0.99998). Exploring these properties could potentially have impacts on the accuracy of SIFT features located in range images and the source point cloud points, but appear to be unexplored in the literature. If further work on this topic permits, this would be an excellent area to begin optimizing a SIFT algorithm to ALS range-based data.

Finally, the characterization of aircraft error distributions need to be more fully explored. In this research a direct derivation of the relationship between the nominal aircraft position uncertainty, the ground feature points uncertainty, the uncertainty of the correspondence transform, and how that transform impacts the uncertainty of the a new aircraft position estimate is not performed. Having an error distribution properly characterized for this process would be invaluable for use with Kalman filter based navigation system. Although the histograms in this research suggest Gaussian distribution behavior, much more testing under other conditions should be performed to make this more certain. There are also suggestions in the literature [131] that the common practice of using RMSE to measure performance in algorithms processing LiDAR point cloud data may not be well characterized by the normal distribution at all.

Appendix: A

One of the benefits of working with SIFT features is that what creates good features can often be seen by human eyes and features common to both data sets can be quickly judged the same way. The difficult part of working with features is automating their use and setting useful conditions. As such it seems valuable to show the data set created by the ASPN flyover, and the features that matched with selected OSIP tile. This gives insight into how the SIFT algorithm with RANSAC was performing, the features impact on accuracy, what constituted a good image and what kind of terrain, and also illustrates just how powerful a constraint the single OSIP tile selection was on position estimate availability.

In this chapter the 66 swaths that are matched to reference tiles and the SIFT features that survived RANSAC and the thresholds are shown here. In each image, the collected ASPN swath scaled range image is shown on the left, the selected OSIP reference tile on the right. The yellow lines connecting the two represent the location of the SIFT features common to both images as determined by the descriptor match, filtered by RANSAC and the threshold conditions. Scattered lines indicate that a poor consistency in the resulting correspondence transform, meaning that outliers are still present, or there were too few features for RANSAC to build a good consensus. If after RANSAC the number of features is below the threshold of 8, the algorithm stops, rejects that swath, and starts processing the next one. Both well matched tiles and matching failure tiles are shown. The tiles are arranged in the order collected from the flight data, and follow the numbered *x* axis in Figure 4.1 and Figure 4.3. Each of these figures represents the +20 meters error case. In the following figures, a “weak outlier” is defined as a feature match that survived RANSAC and meets the threshold constraints, but is in error near the threshold of 4 meters. These are generally only present on large sets of matched features, and their impact on the resulting correspondence and aircraft position error is noticeable, but small.

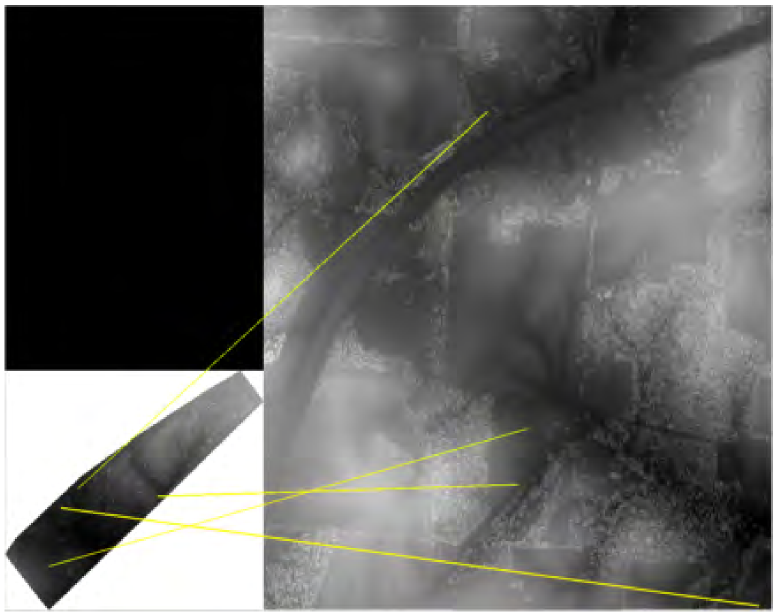


Figure A.1: This swath is a mix of forest and farmland, but too few features are found to make a match. This tile is shortly after takeoff and at a much lower altitude then the rest.

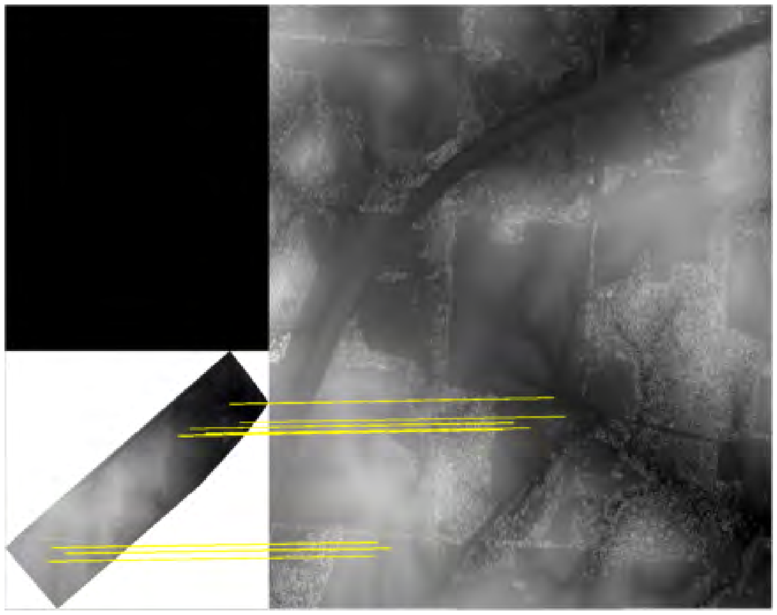


Figure A.2: This swath is a mix of forest and farmland and matches 9 features with an 3D aircraft position magnitude error of 2.77 meters.

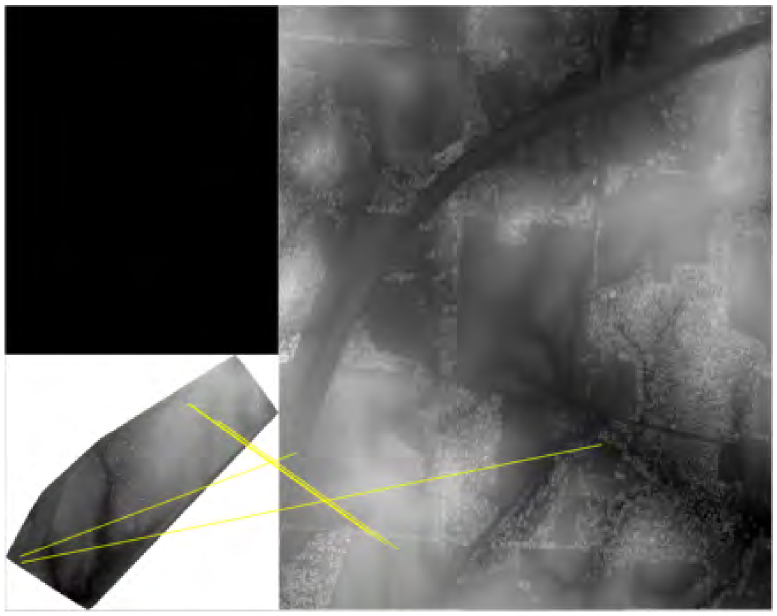


Figure A.3: This swath failed because of too few features in farmland, although 3 appear to be correct. Less than 50% of the swath is present in the tile.

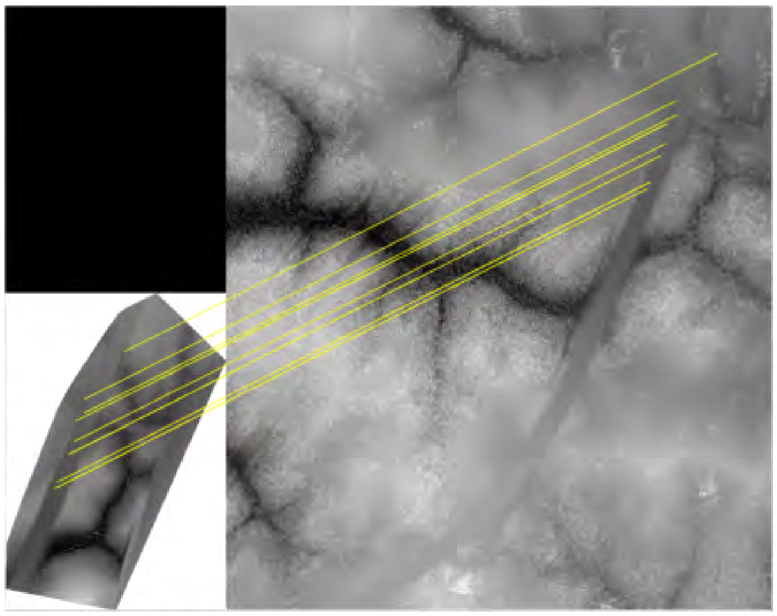


Figure A.4: This swath matched 11 features with a 3D aircraft position magnitude error of 3.76 meters with exclusively the road terrain.

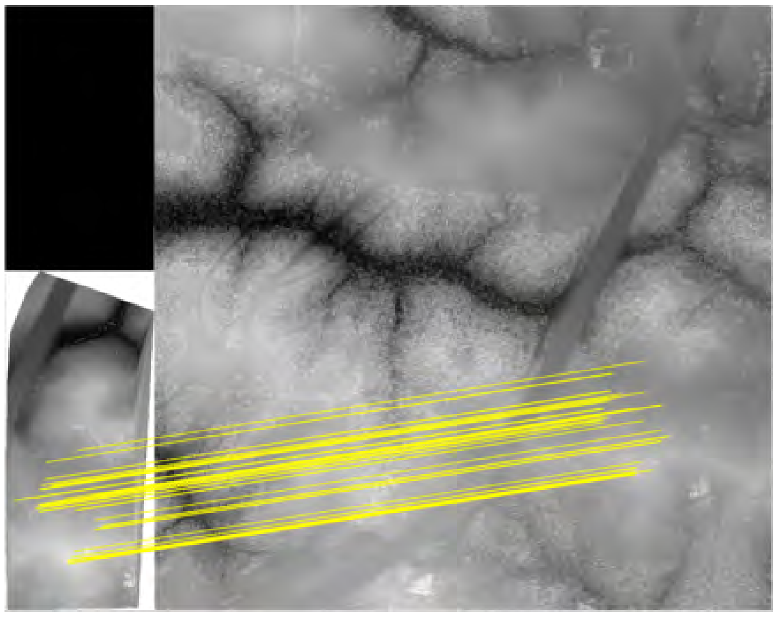


Figure A.5: While a road is present, this swath matched 26 features on farmland for an 3D aircraft position magnitude error of 3.44 meters. Close inspection shows two matches are weak outliers.

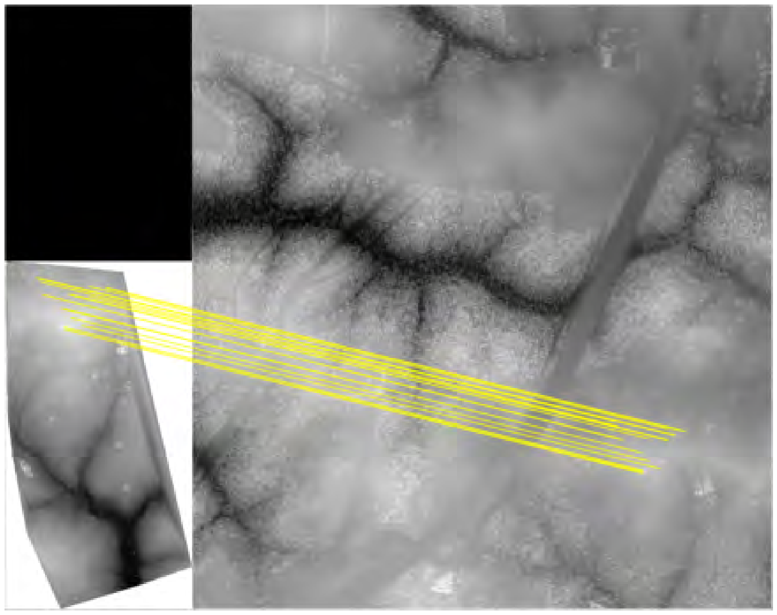


Figure A.6: This swath matched 13 features on farmland for an 3D aircraft position magnitude error of 5.44 meters. Close inspection shows two matches are weak outliers.

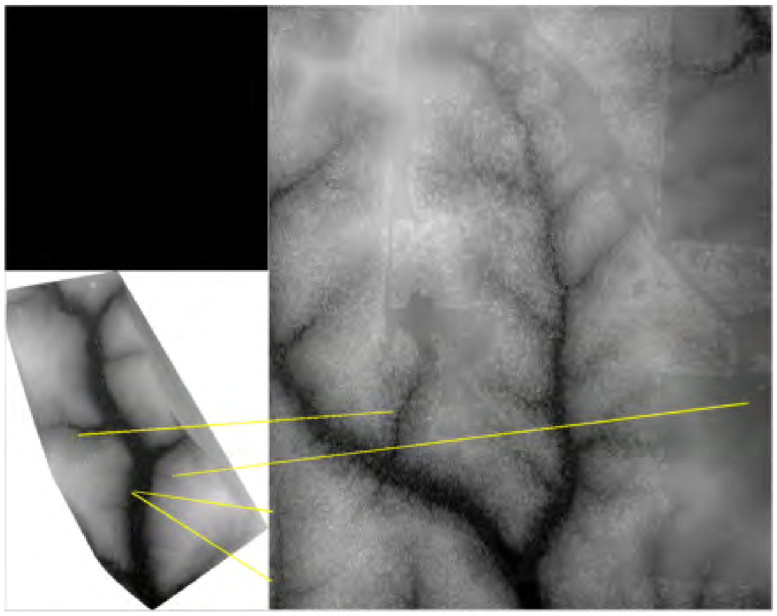


Figure A.7: This swath failed because of too few features in forest. Less than 50% of the swath is present in the tile.

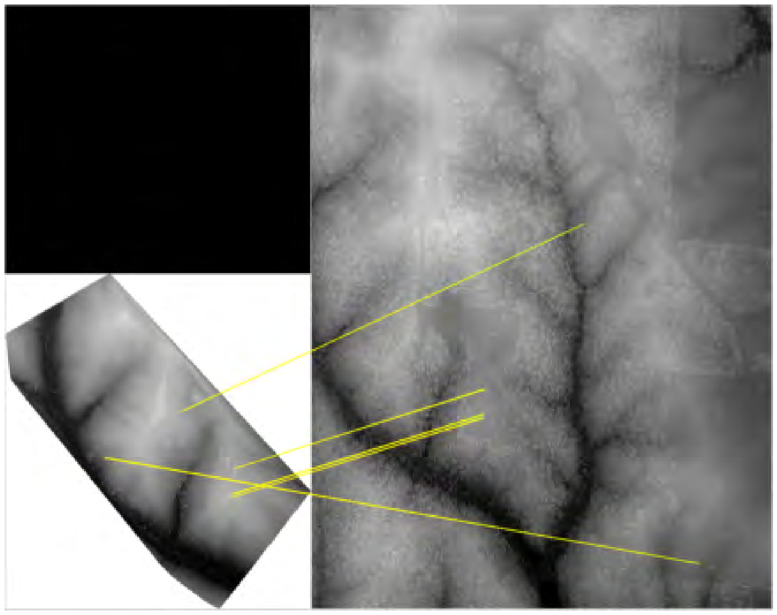


Figure A.8: This swath failed because of too few features in forest and farmland, although 3 appear to be correct in the farmland.

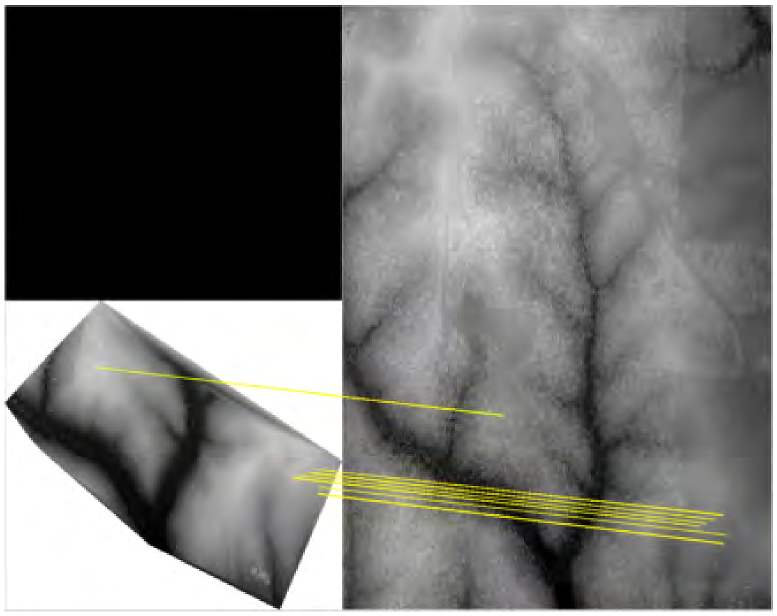


Figure A.9: This swath failed because of too few features in forest and farmland, although all 7 appear to be correct.

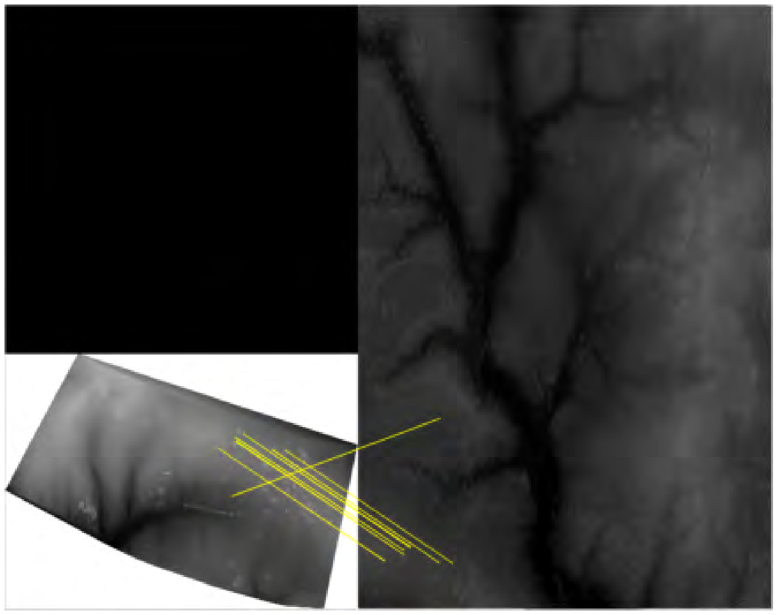


Figure A.10: This swath failed because of too few features in farmland, although only one is an outlier. Less than 50% of the swath is present in the tile. The tile is unusually dark and close inspection reveals a small cluster of abnormally high elevation pixels present.

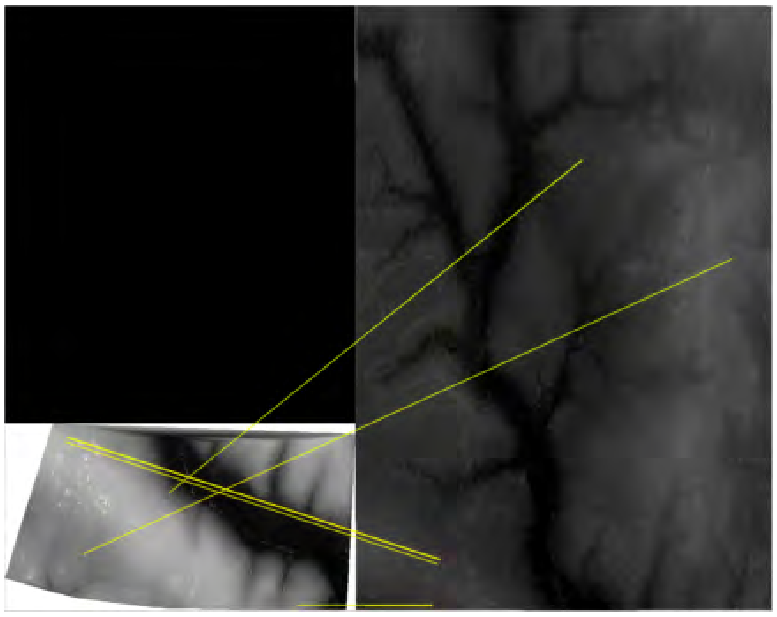


Figure A.11: This swath failed because of too few features in farmland, although 2 appear correct. Less than 50% of the swath is present in the tile. The tile is unusually dark and close inspection reveals a small cluster of abnormally high elevation pixels present.

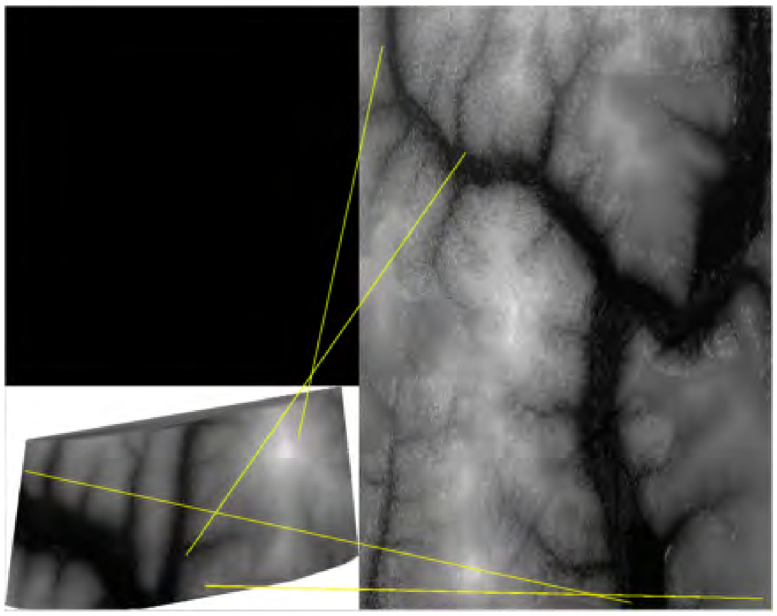


Figure A.12: This swath failed because of too few features in forest. Less than 50% of the swath is present in the tile.

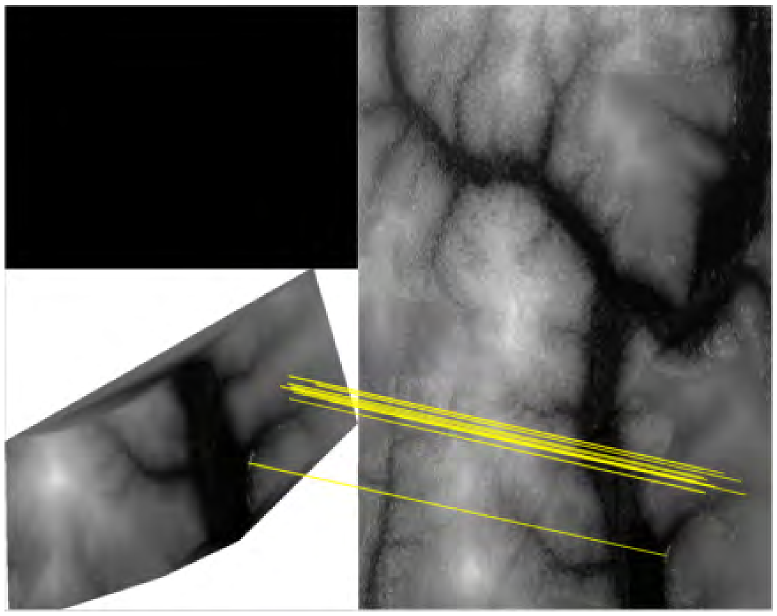


Figure A.13: This swath matched 12 features on farmland for an 3D aircraft position magnitude error of 2.11 meters. The river present appears shallow enough to generate LiDAR signal returns.

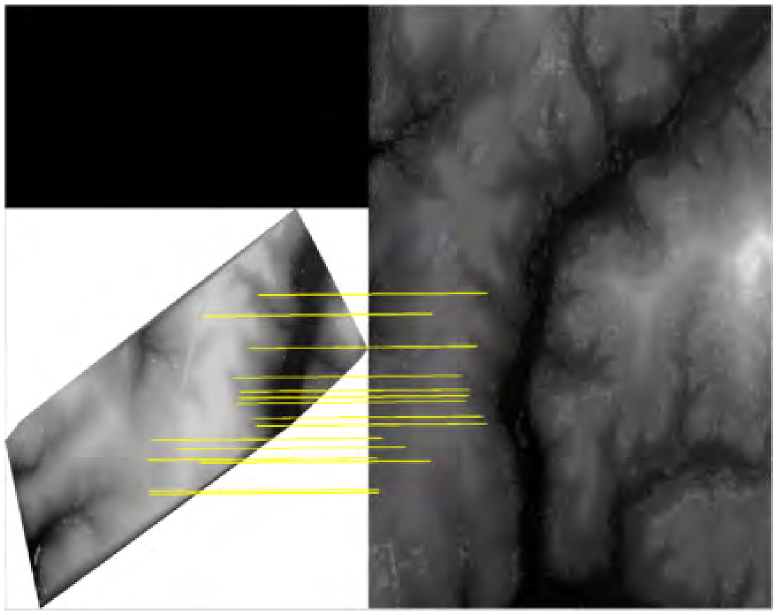


Figure A.14: This swath matched 15 features on farmland for an 3D aircraft position magnitude error of 2.70 meters.

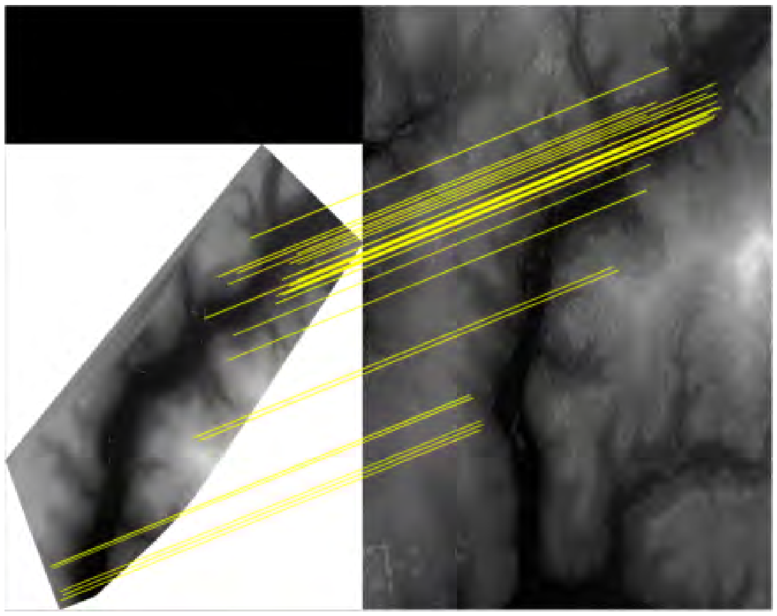


Figure A.15: This swath matched 29 features on farmland for an 3D aircraft position magnitude error of 1.63 meters.

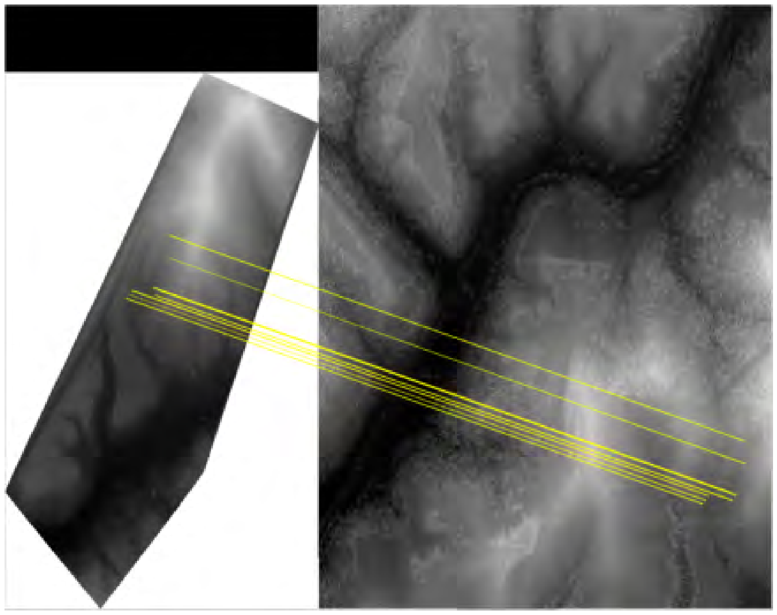


Figure A.16: This swath matched 10 features on farmland for an 3D aircraft position magnitude error of 6.31 meters. Less than 50% of the swath is present in the tile. Close inspection shows 2 matches is are weak outlier.

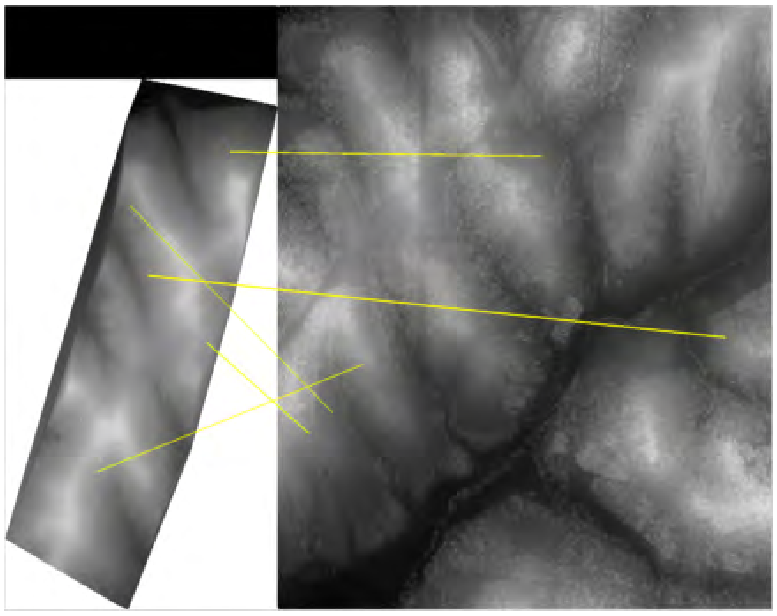


Figure A.17: This swath failed because of too few features in forest and farmland, although 1 appears to be correct.

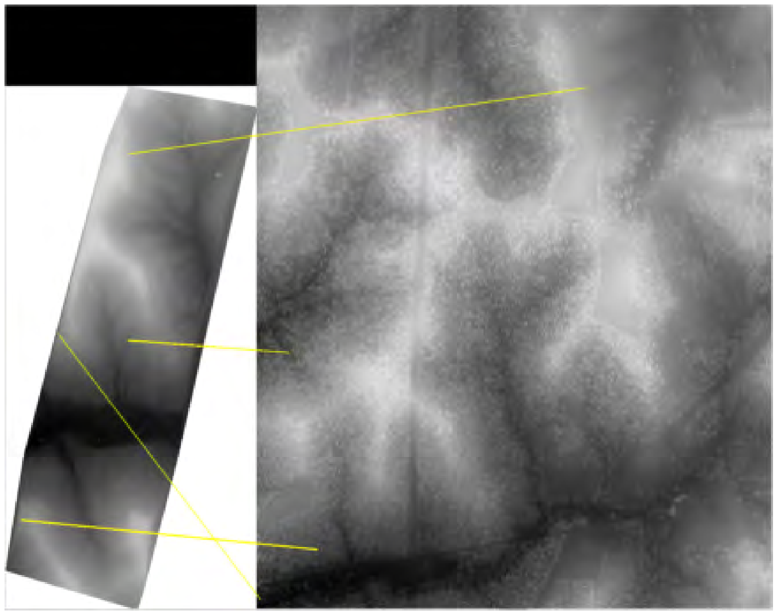


Figure A.18: This swath failed because of too few features in forest, none are correct.

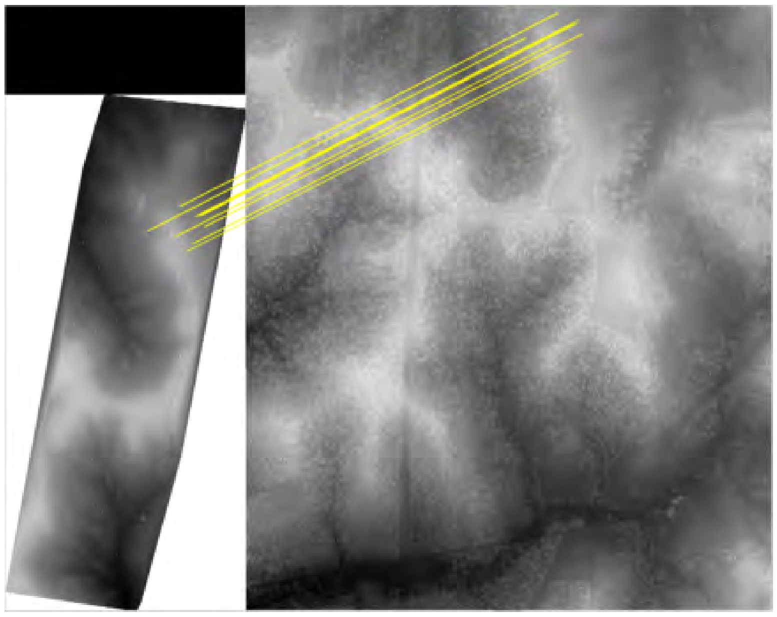


Figure A.19: This swath matched 10 features on forest and farmland for an 3D aircraft position magnitude error of 3.06 meters.

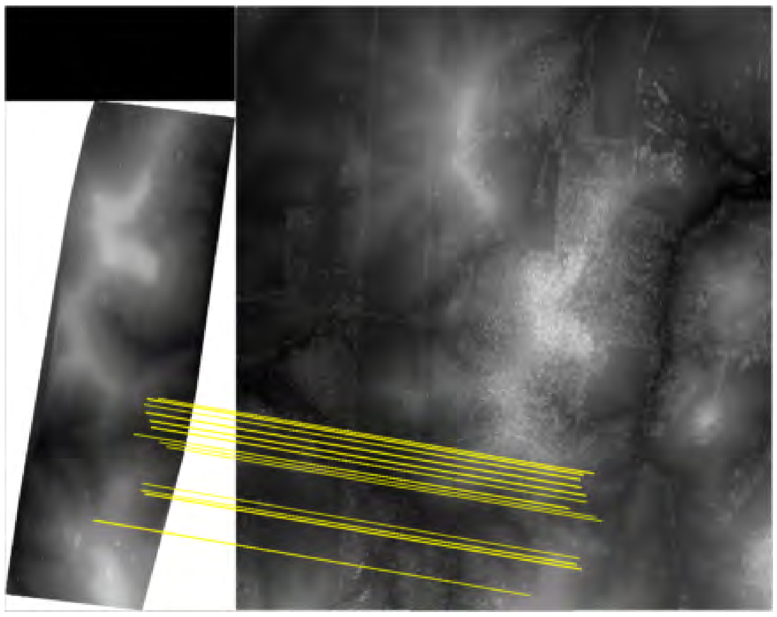


Figure A.20: This swath matched 13 features on forest and farmland for an 3D aircraft position magnitude error of 1.22 meters.

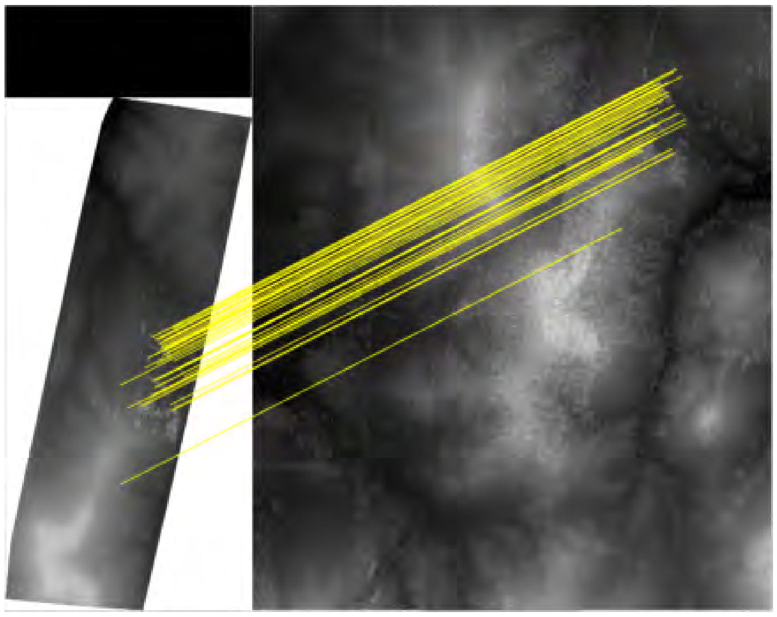


Figure A.21: This swath matched 38 features on forest and farmland for an 3D aircraft position magnitude error of 2.23 meters. Close inspection shows 2 matches are weak outliers.

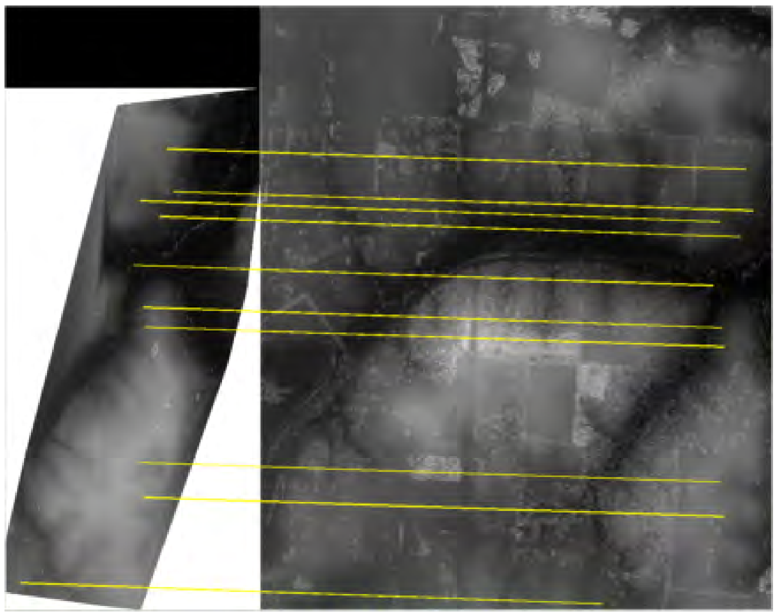


Figure A.22: This swath matched 11 features on farmland for an 3D aircraft position magnitude error of 1.33 meters.

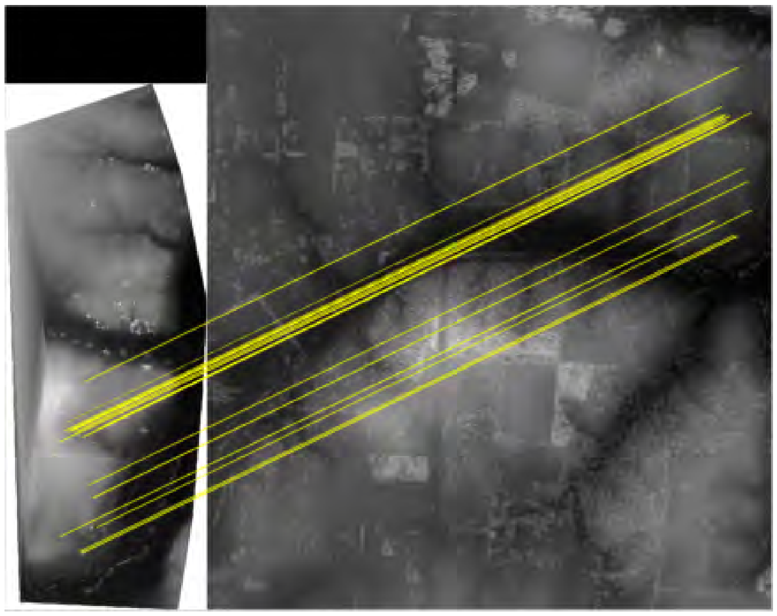


Figure A.23: This swath matched 16 features on farmland for an 3D aircraft position magnitude error of 4.07 meters. Less than 50% of the swath is present in the tile. Close inspection shows 2 matches are weak outliers.

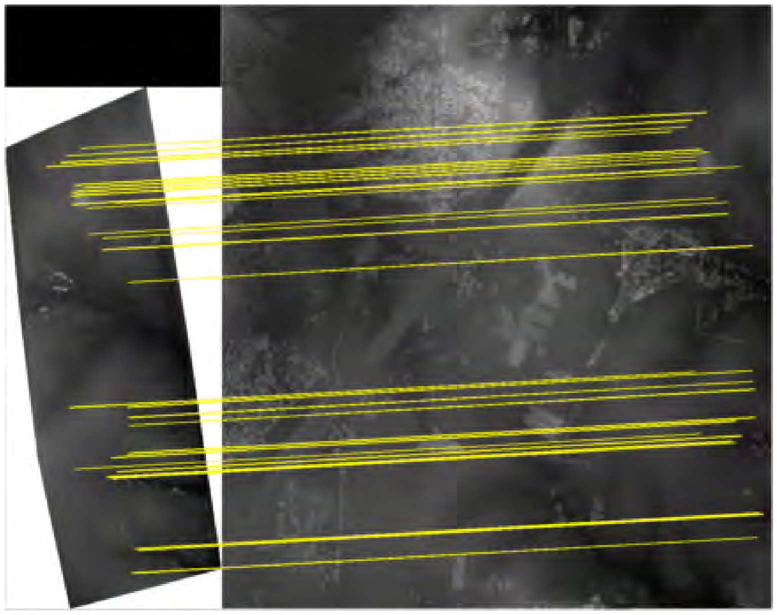


Figure A.24: This swath matched 29 features on farmland for an 3D aircraft position magnitude error of 1.83 meters. Less than 50% of the swath is present in the tile. Close inspection shows 3 matches are weak outliers.

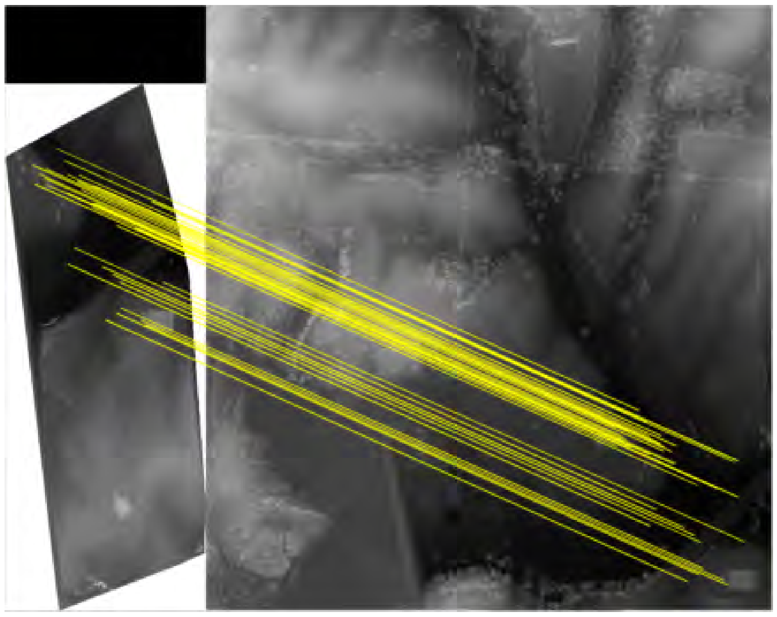


Figure A.25: This swath matched 36 features on farmland for an 3D aircraft position magnitude error of 1.46 meters. Less than 50% of the swath is present in the tile. Close inspection shows 4 matches are weak outliers.

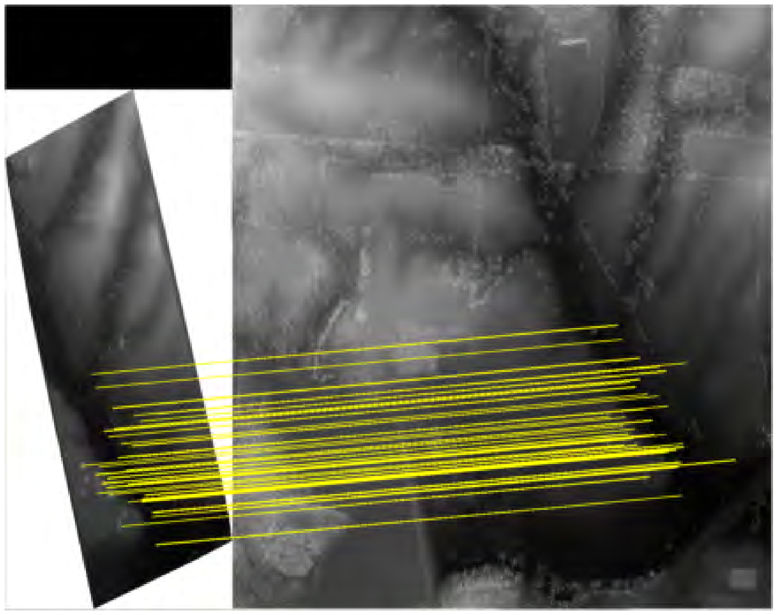


Figure A.26: This swath matched 37 features on farmland for an 3D aircraft position magnitude error of 4.22 meters. Close inspection shows 3 matches are weak outliers.

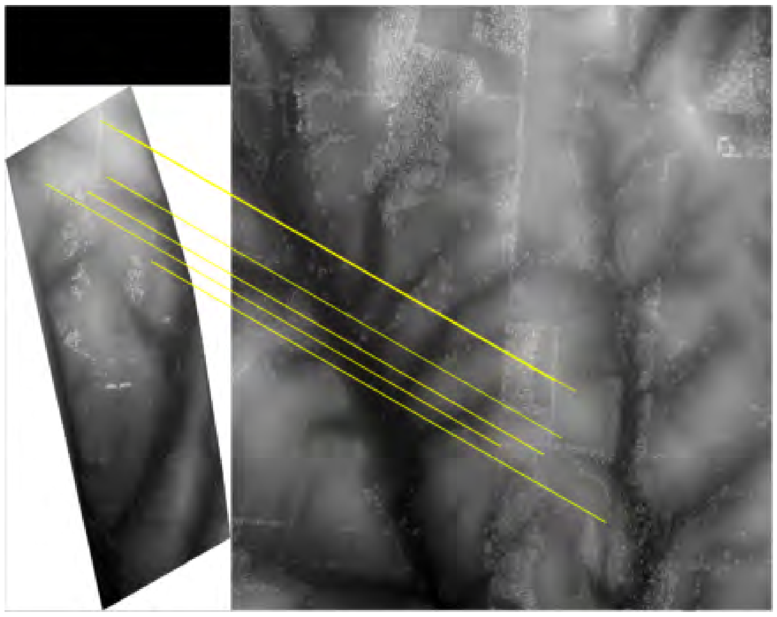


Figure A.27: This swath failed because of too few features in farmland, although all 6 appear correct Less than 50% of the swath is present in the tile.

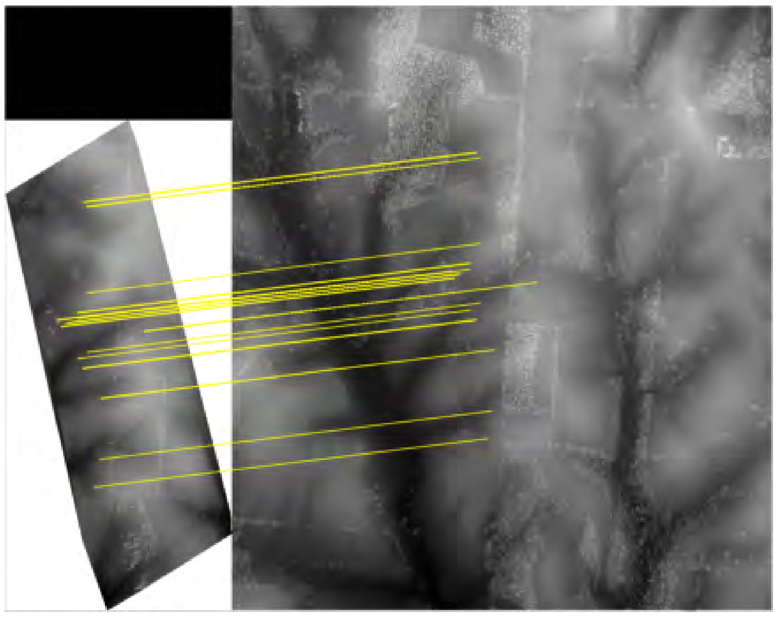


Figure A.28: This swath matched 17 features on farmland for an 3D aircraft position magnitude error of 2.61 meters. Close inspection shows 2 matches are weak outliers.

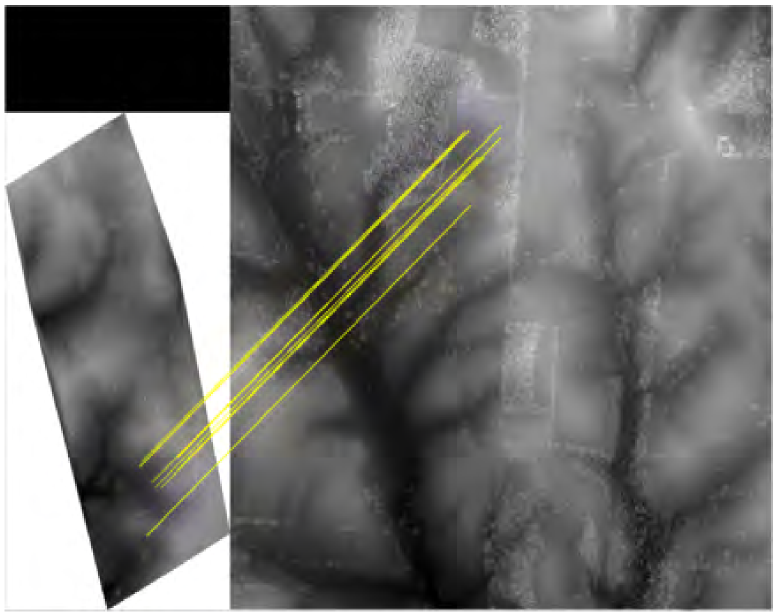


Figure A.29: This swath matched 9 features in farmland for an 3D aircraft position magnitude error of 9.94 meters. Less than 50% of the swath is present in the tile. Close inspection shows 2 matches are weak outliers.

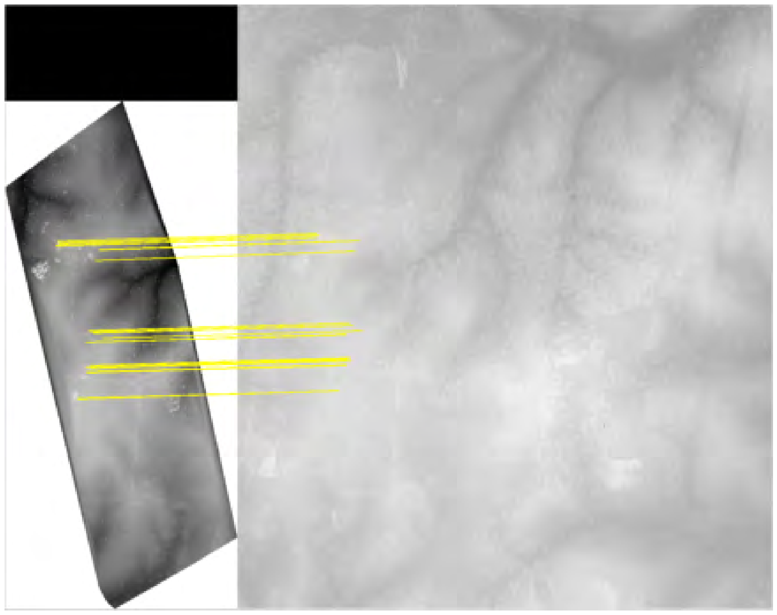


Figure A.30: This swath matched 14 features in farmland for an 3D aircraft position magnitude error of 1.73 meters. The tile is unusually bright and close inspection reveals a small cluster of abnormally low elevation pixels present.

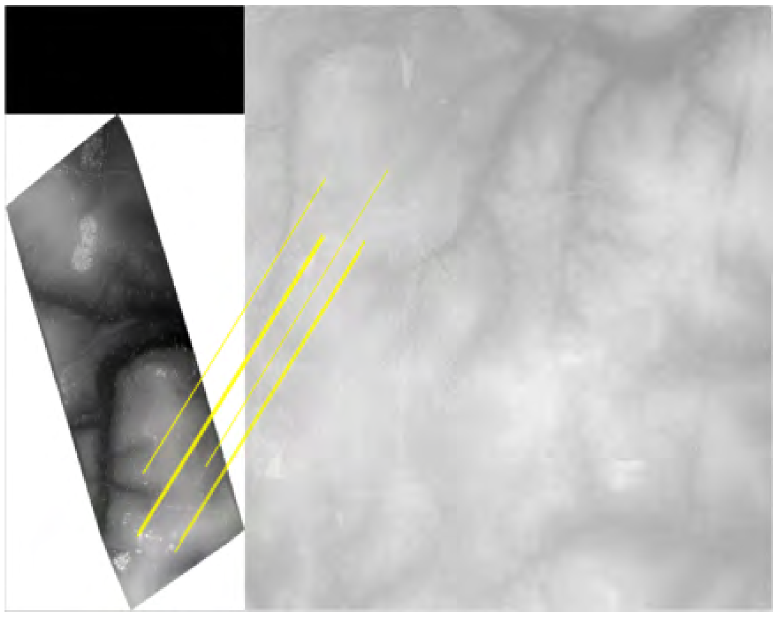


Figure A.31: This swath failed because of too few features in farmland, although all 7 appear correct. The tile is unusually bright and close inspection reveals a small cluster of abnormally low elevation pixels present.

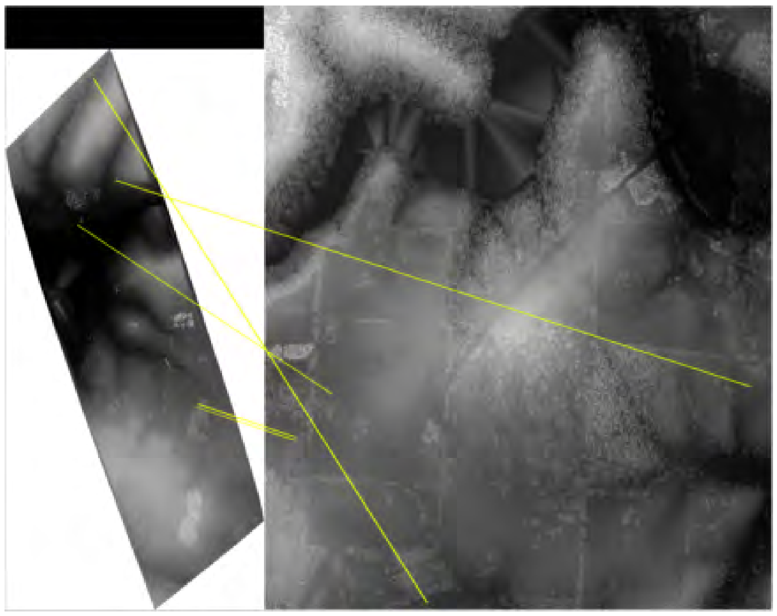


Figure A.32: This swath failed because of too few features in farmland, although 2 appear correct Less than 50% of the swath is present in the tile, and a deep water area.

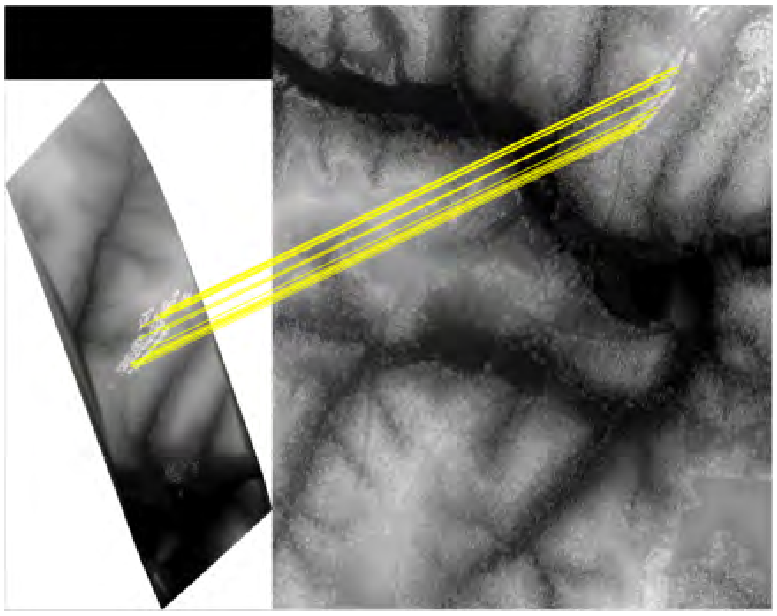


Figure A.33: This swath matched 19 features on forest with a water feature for an 3D aircraft position magnitude error of 10.41 meters. Close inspection shows 2 matches are weak outliers. The entire set of features seem to be from a group of tall trees.

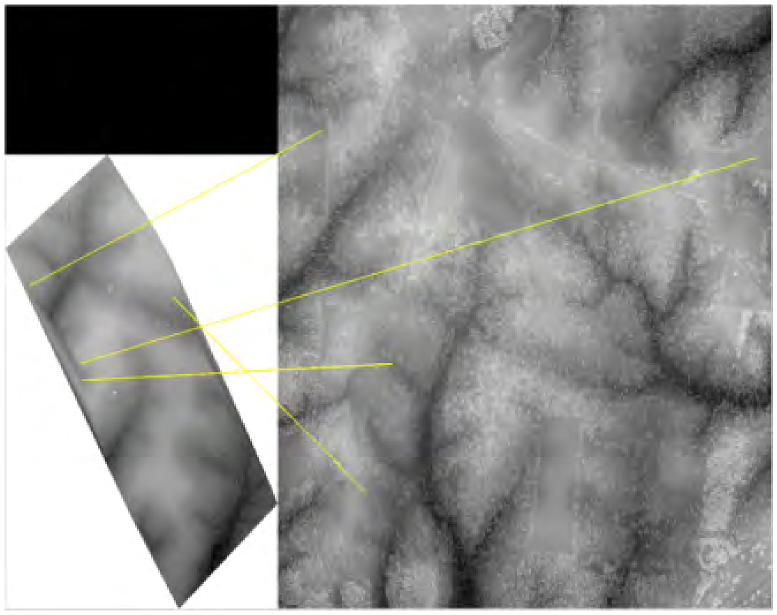


Figure A.34: This swath failed because of too few features in forest and farmland, none appear correct.

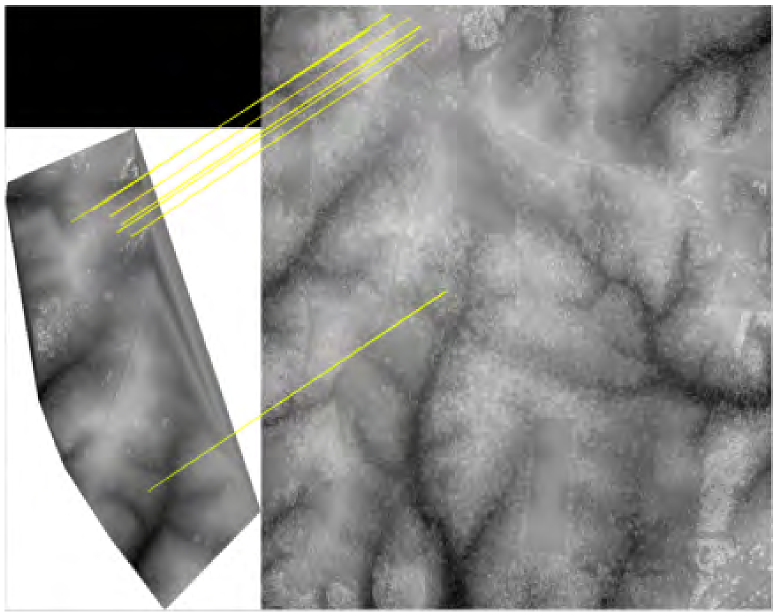


Figure A.35: This swath failed because of too few features in forest and farmland, although all 7 appear correct.

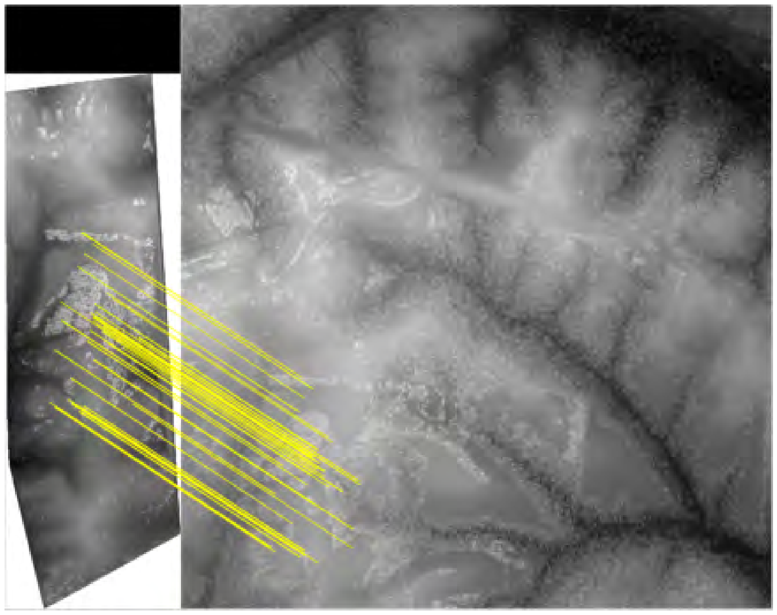


Figure A.36: This swath matched 39 features on forest and farmland for an 3D aircraft position magnitude error of 1.98 meters. Close inspection shows 5 matches are weak outliers. Most of the features seem to be from groups of tall trees.

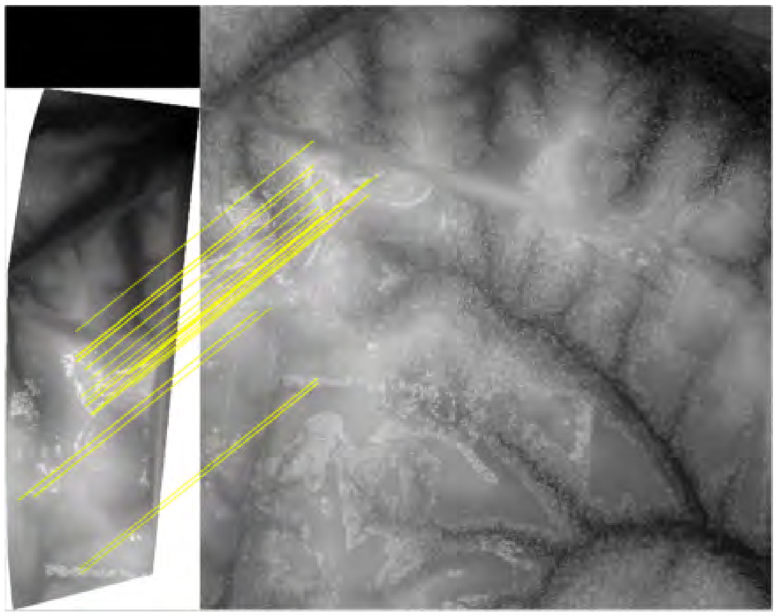


Figure A.37: This swath matched 39 features on forest and farmland with a road for an 3D aircraft position magnitude error of 2.29 meters. Close inspection shows 3 matches are weak outliers. Most of the features seem to be from groups of tall trees.

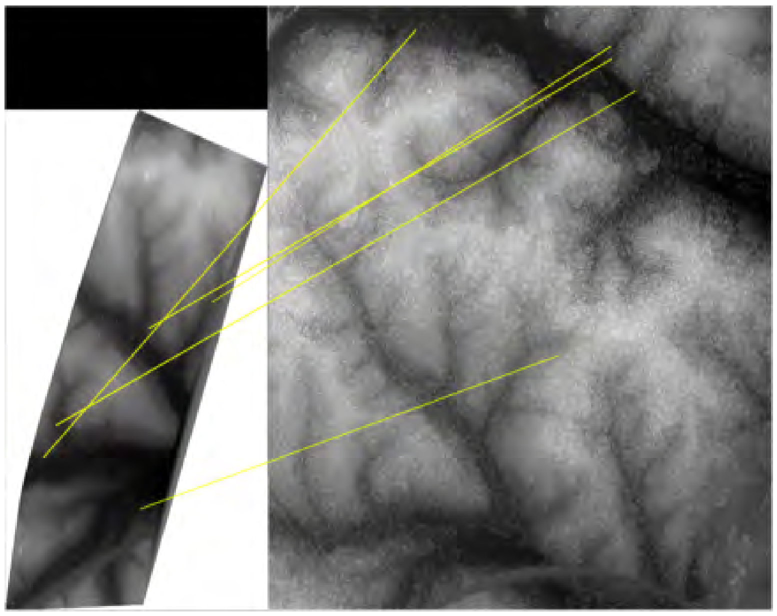


Figure A.38: This swath failed because of too few features in forest, none appear correct.

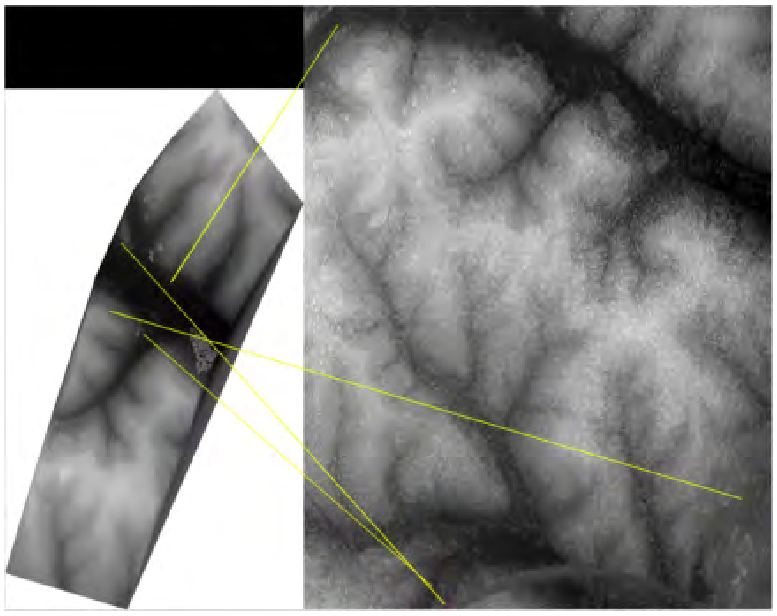


Figure A.39: This swath failed because of too few features in forest, none appear correct.

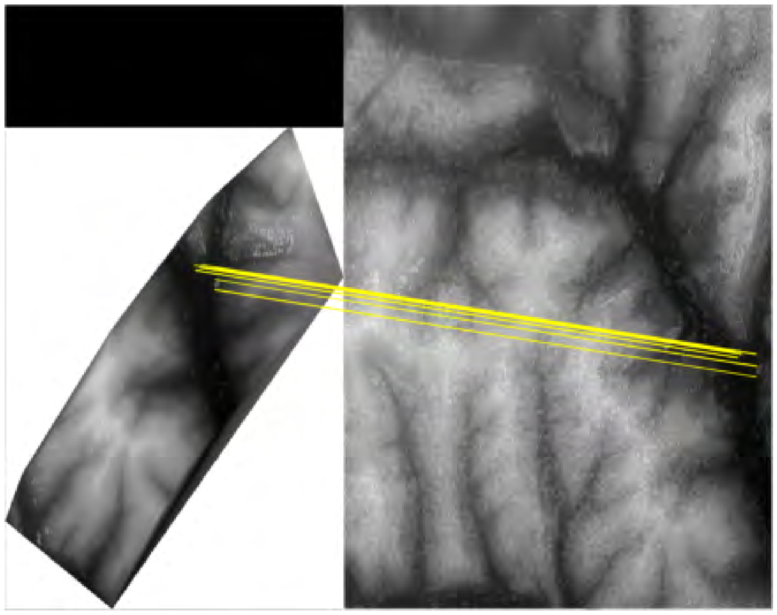


Figure A.40: This swath failed because of too few features in forest, although 6 of the 7 appear correct.

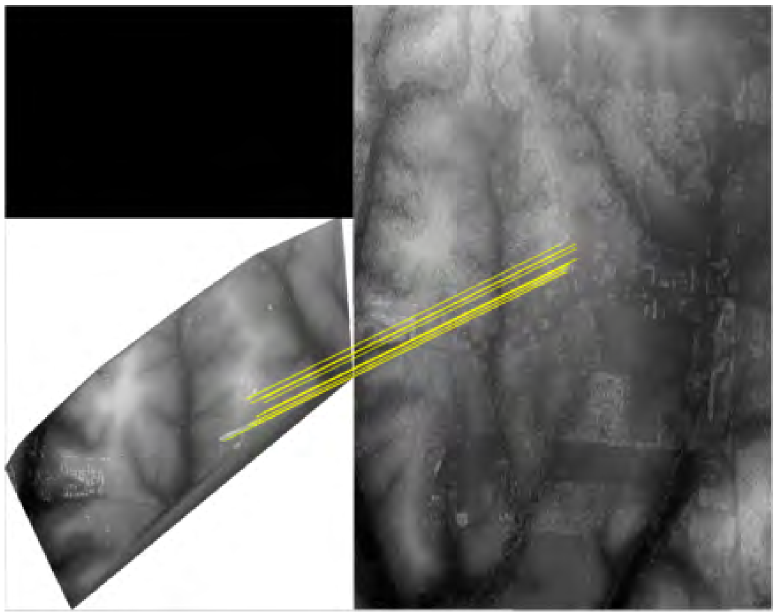


Figure A.41: This swath matched 12 features on forest and farmland for an 3D aircraft position magnitude error of 5.71 meters. Close inspection shows 3 matches are weak outliers.

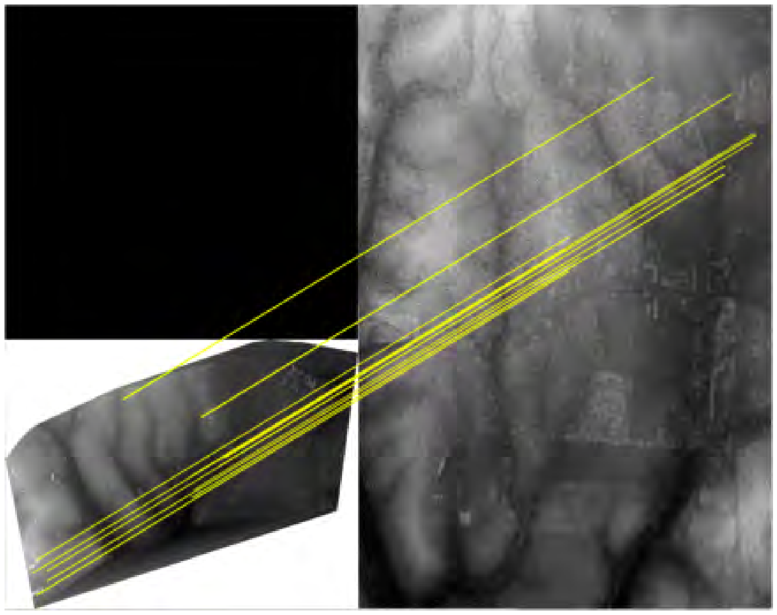


Figure A.42: This swath matched 13 features on forest and farmland for an 3D aircraft position magnitude error of 3.52 meters. Close inspection shows 2 matches are weak outliers.

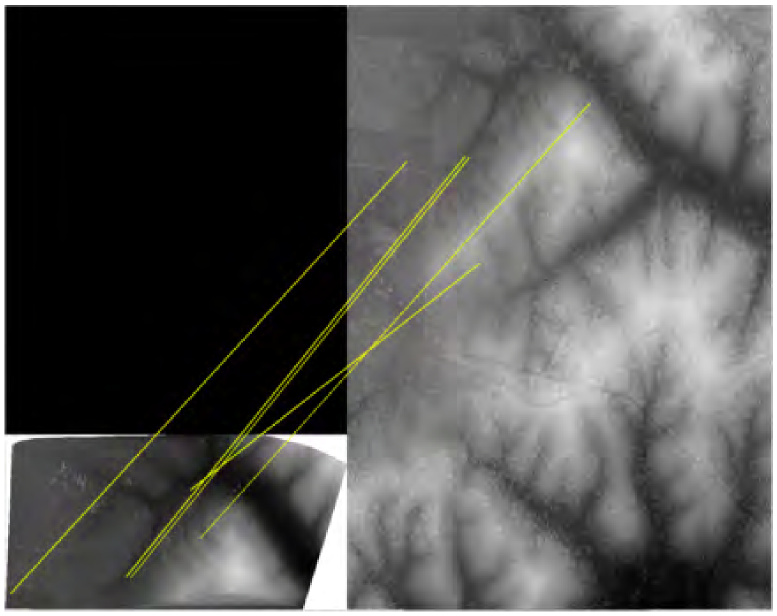


Figure A.43: This swath failed because of too few features in forest and farmland, although 2 appear correct.

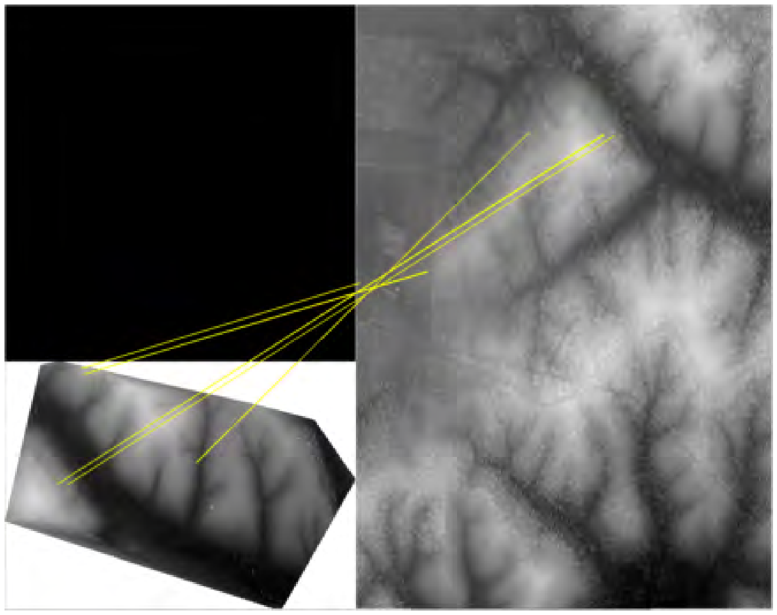


Figure A.44: This swath failed because of too few features in forest, although 2 appear correct.

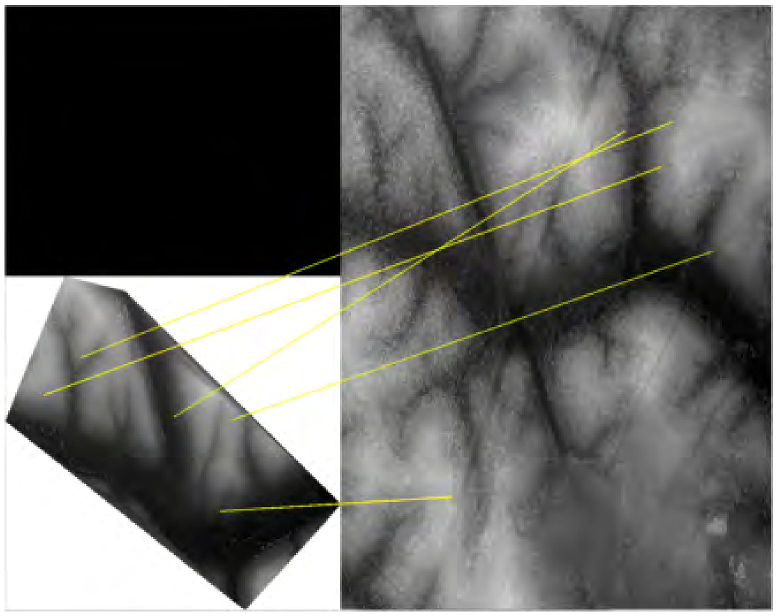


Figure A.45: This swath failed because of too few features in forest with a small road, none appear correct.

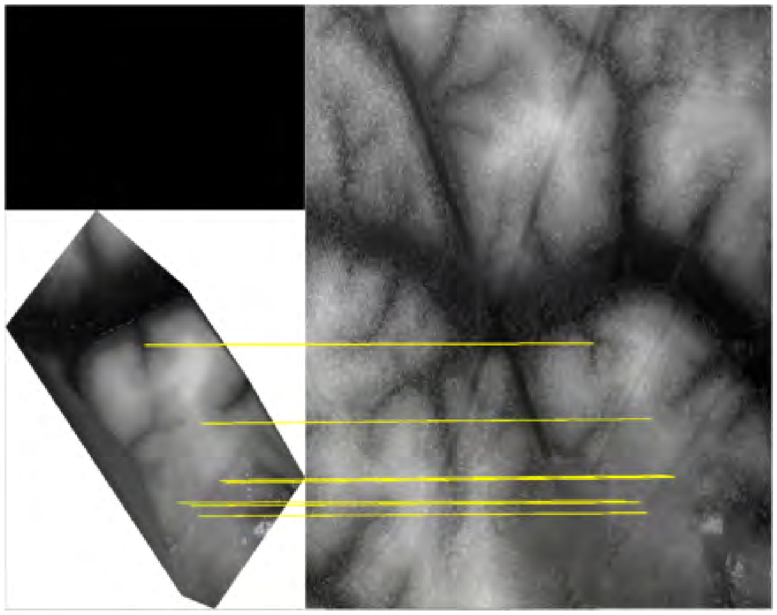


Figure A.46: This swath failed because of too few features in forest and farmland, although all 7 appear correct.

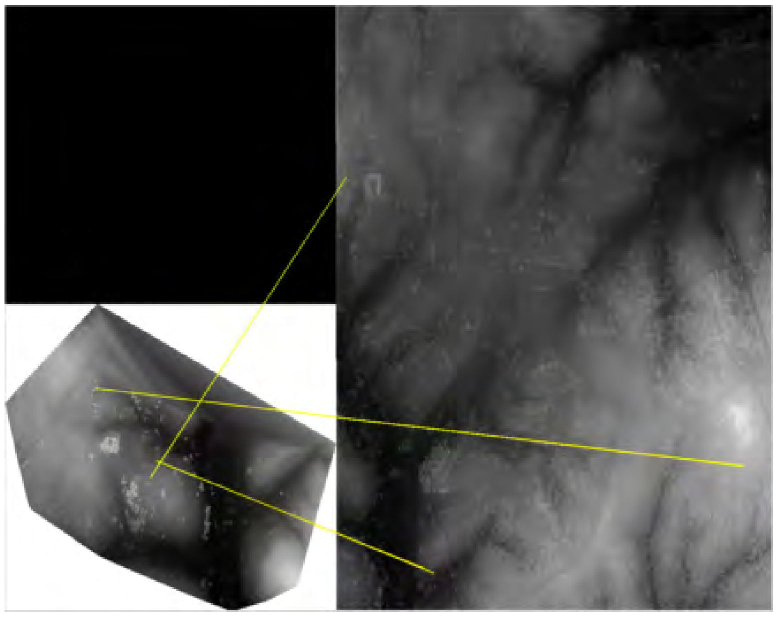


Figure A.47: This swath failed because of too few features in farmland, none appear correct. Less than 50% of the swath is present in the tile.

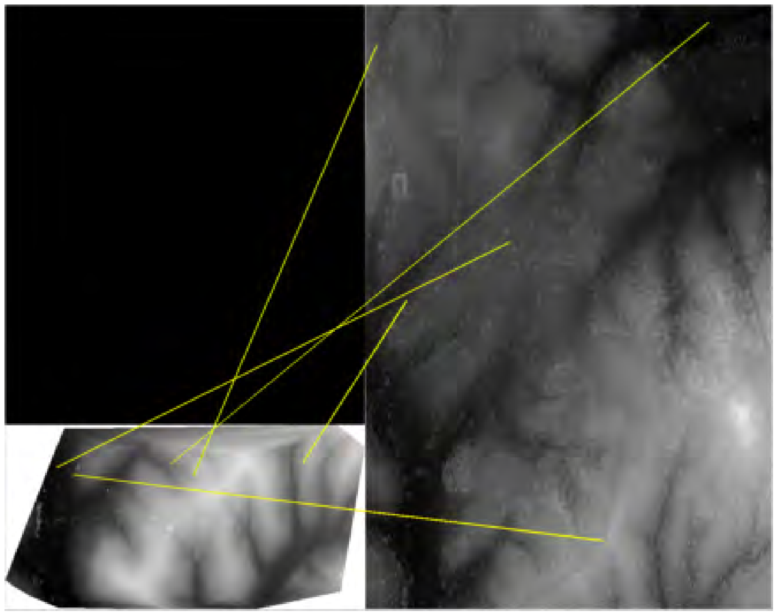


Figure A.48: This swath failed because of too few features in farmland, none appear correct. Less than 50% of the swath is present in the tile.

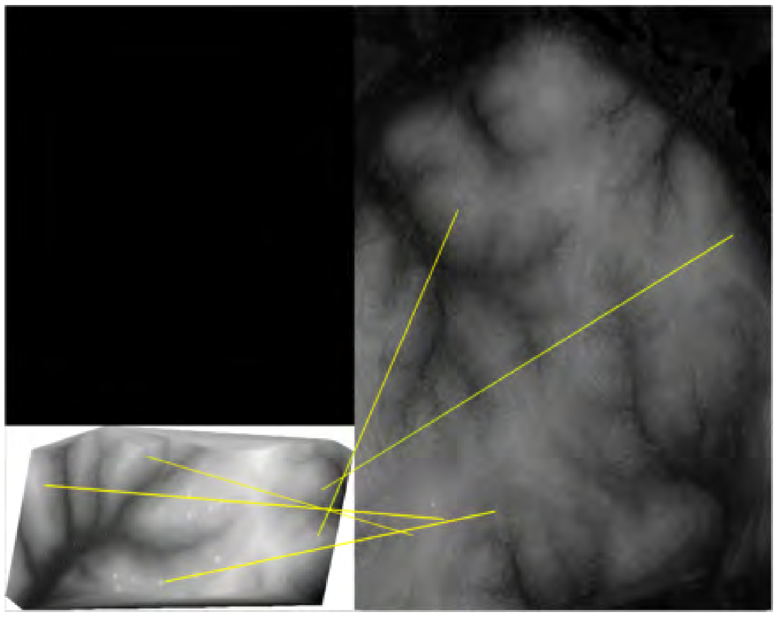


Figure A.49: This swath failed because of too few features in farmland, none appear correct. Less than 50% of the swath is present in the tile.

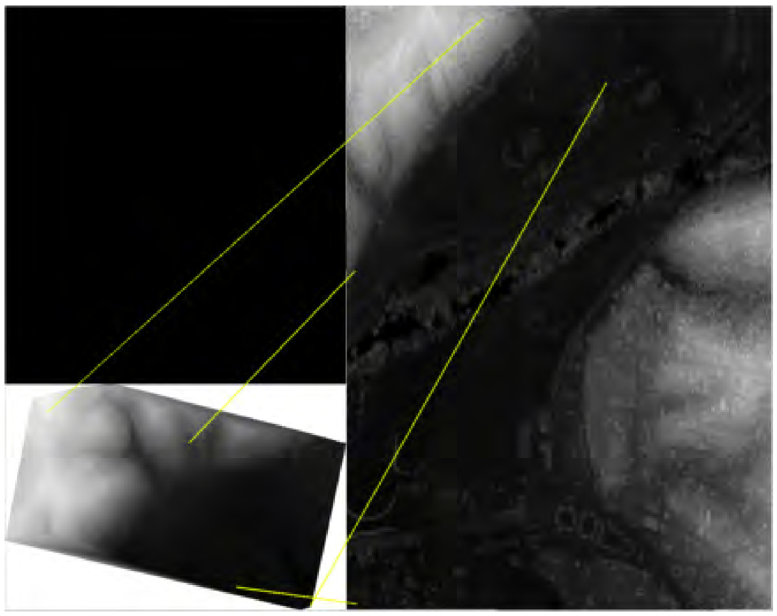


Figure A.50: This swath failed because of too few features in forest and city, none appear correct. Less than 50% of the swath is present in the tile.

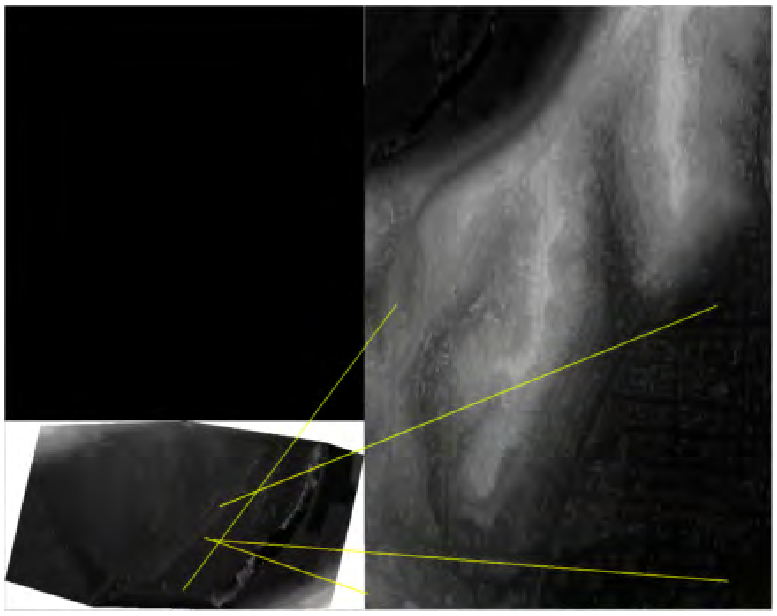


Figure A.51: This swath failed because of too few features in city with a river, none appear correct. Less than 50% of the swath is present in the tile.

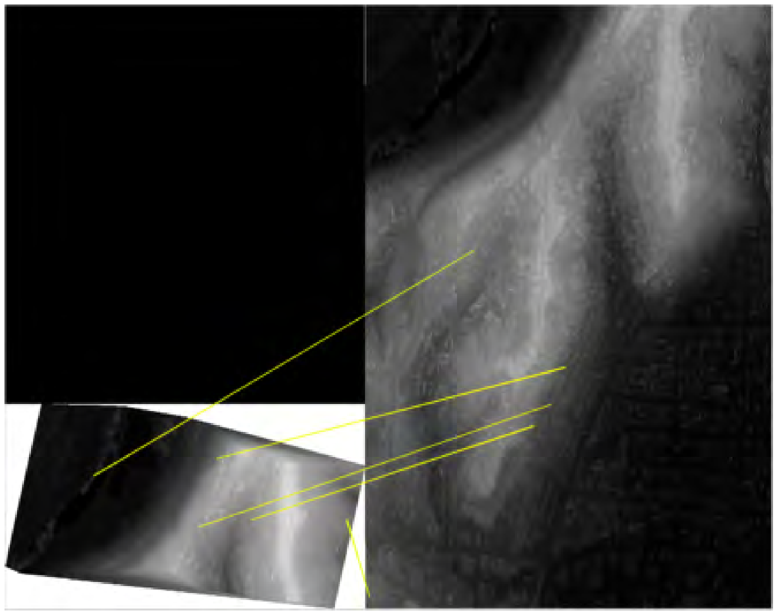


Figure A.52: This swath failed because of too few features in city with a river, none appear correct.

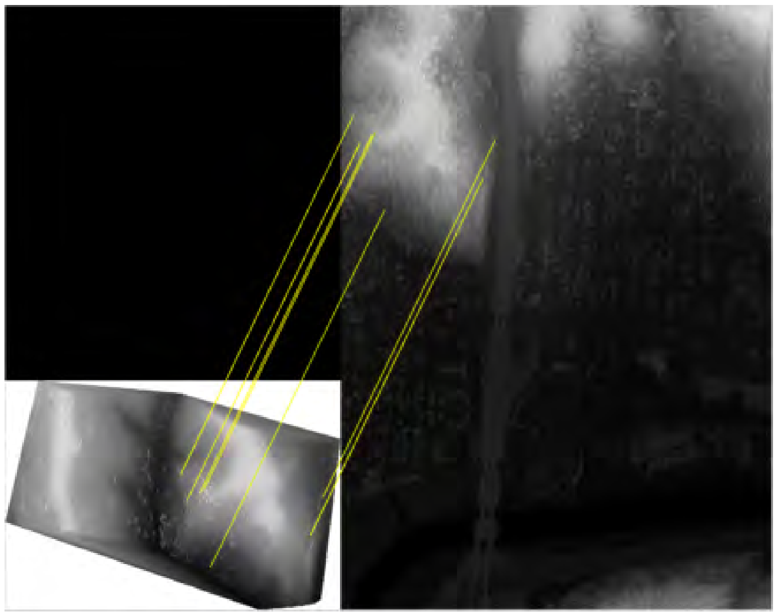


Figure A.53: This swath failed because of too few features in city with forest, although all 7 appear correct. Less than 50% of the swath is present in the tile.

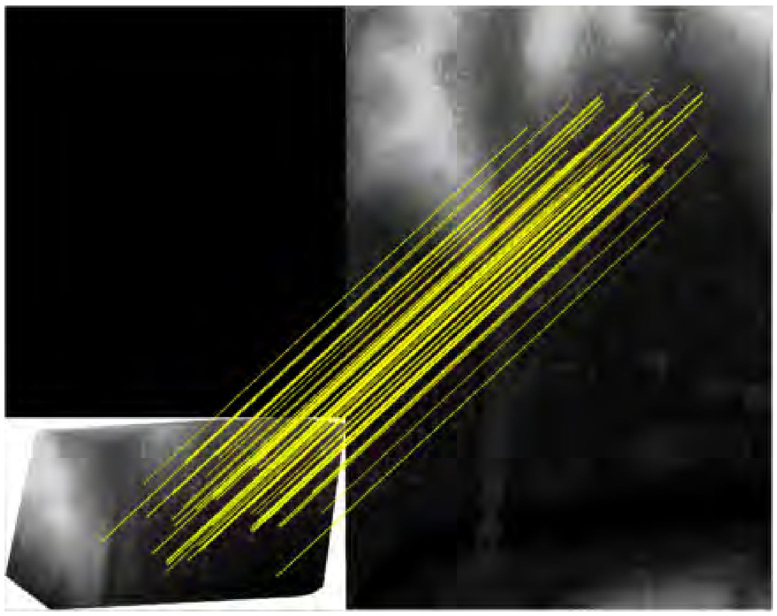


Figure A.54: This swath matched 58 features in city with forest and a road for an 3D aircraft position magnitude error of 2.20 meters. Close inspection shows 5 matches are weak outliers.

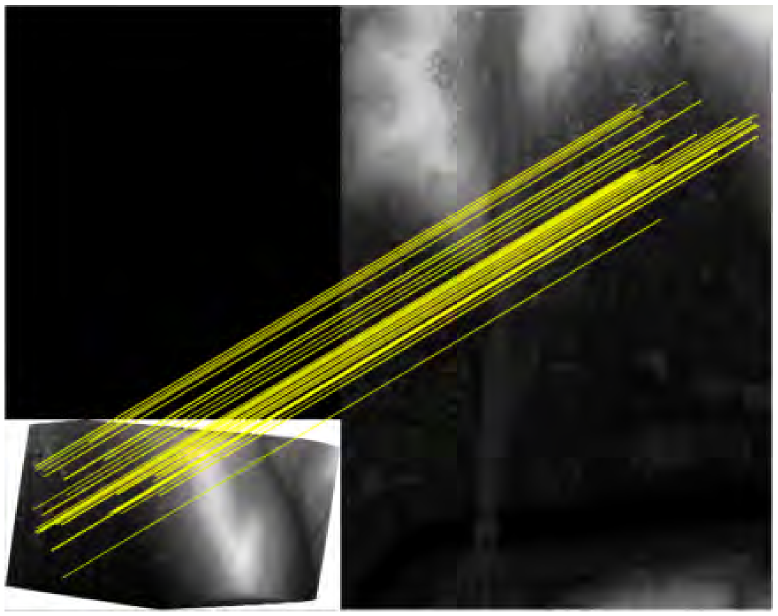


Figure A.55: This swath matched 35 features in city with forest for an 3D aircraft position magnitude error of 1.60 meters. Close inspection shows 2 matches are weak outliers. Less than 50% of the swath is present in the tile.

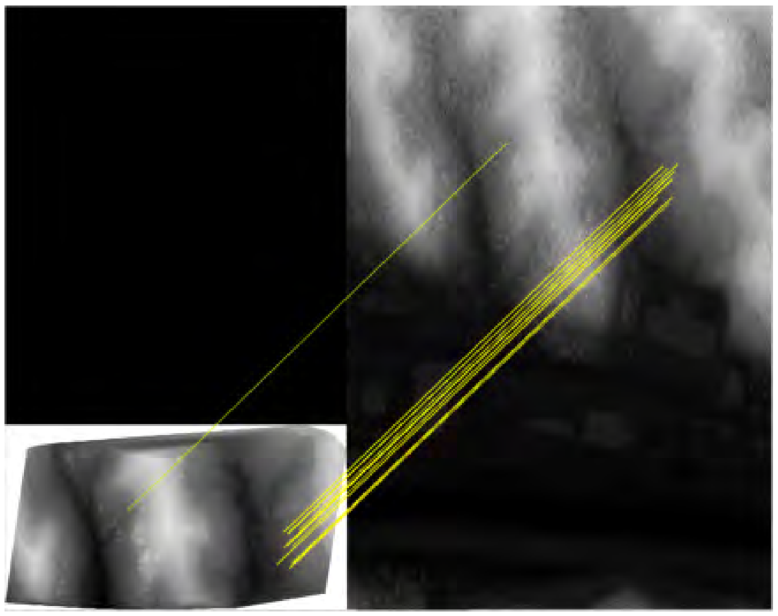


Figure A.56: This swath matched 12 features in forest for an 3D aircraft position magnitude error of 5.05 meters. Close inspection shows 2 matches are weak outliers.

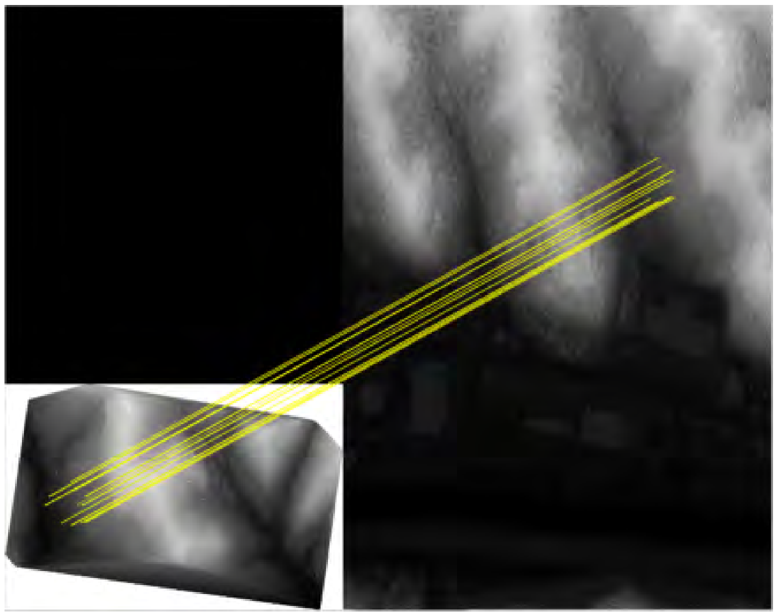


Figure A.57: This swath matched 13 features in forest for an 3D aircraft position magnitude error of 8.19 meters. Close inspection shows 2 matches are weak outliers. Less than 50% of the swath is present in the tile.

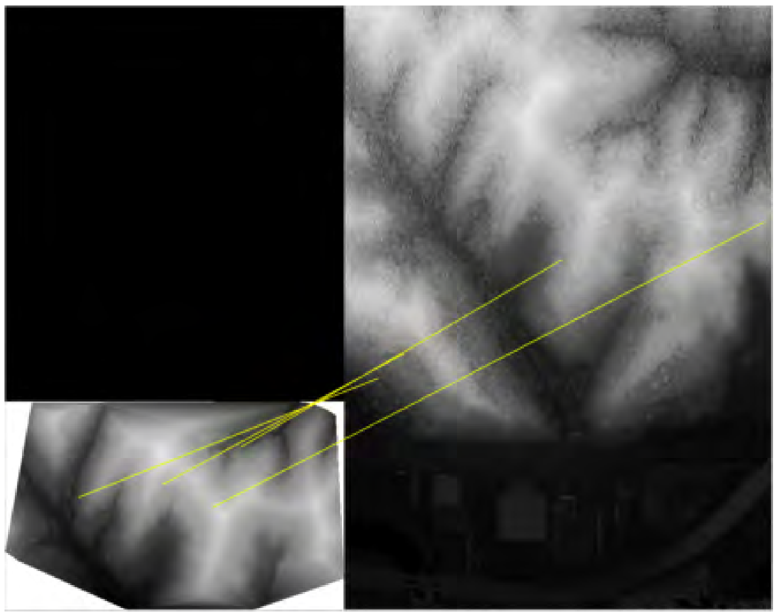


Figure A.58: This swath failed because of too few features in forest, none appear correct.

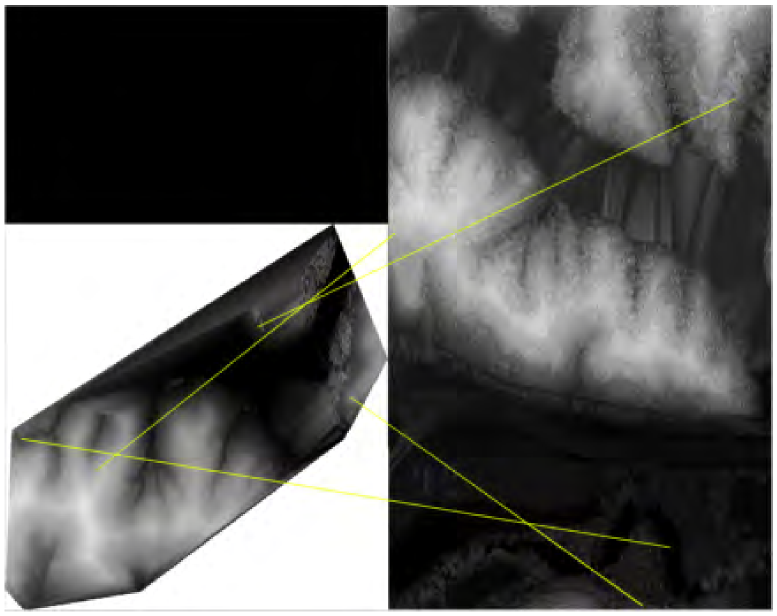


Figure A.59: This swath failed because of too few features in forest with a large deep water area and less than 50% of the swath is present in the tile. None of the features appear correct.

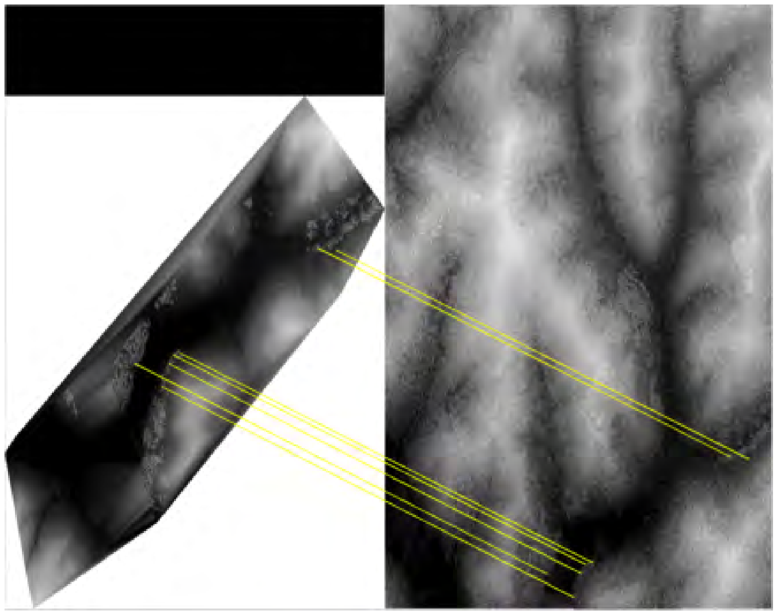


Figure A.60: This swath failed because of too few features in forest, although all 7 appear correct.

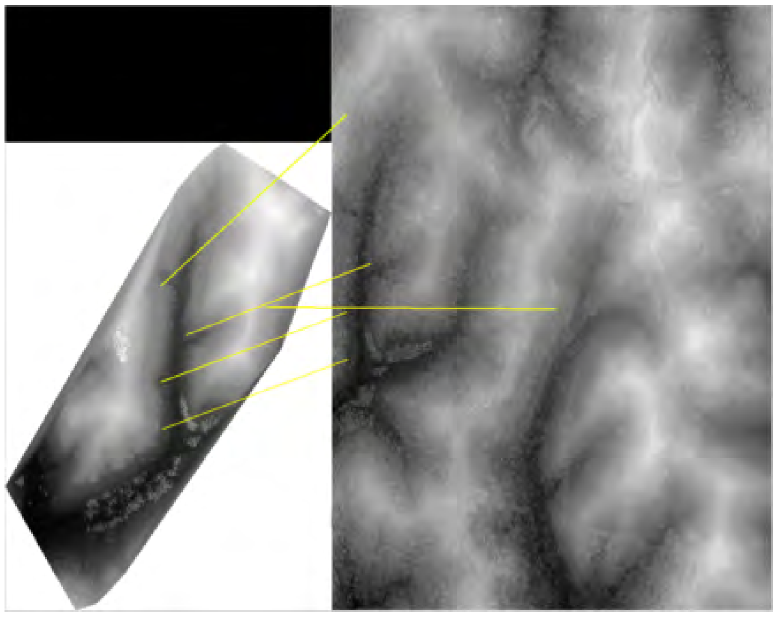


Figure A.61: This swath failed because of too few features in forest, although 3 appear correct. Less than 50% of the swath is present in the tile.

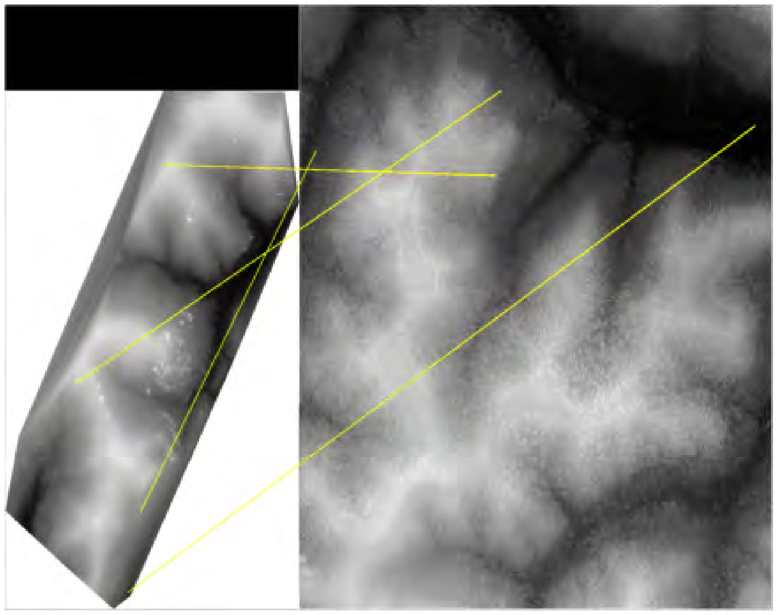


Figure A.62: This swath failed because of too few features in forest, none appear correct. Less than 50% of the swath is present in the tile.

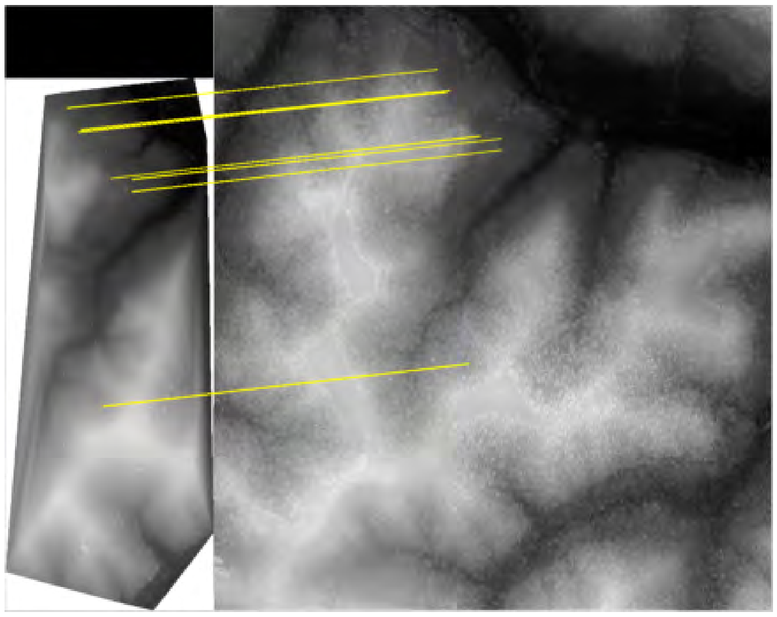


Figure A.63: This swath failed because of too few features in forest, although all 7 appear correct.

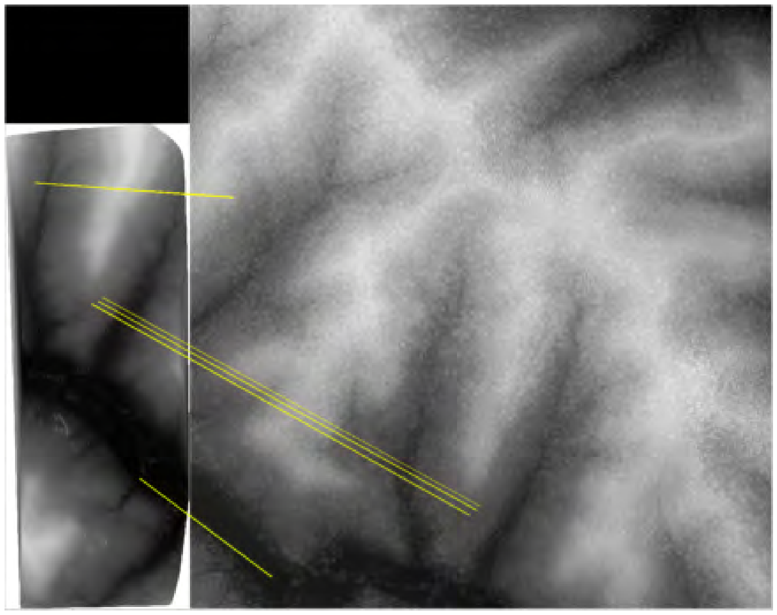


Figure A.64: This swath failed because of too few features in forest, although 3 appear correct.

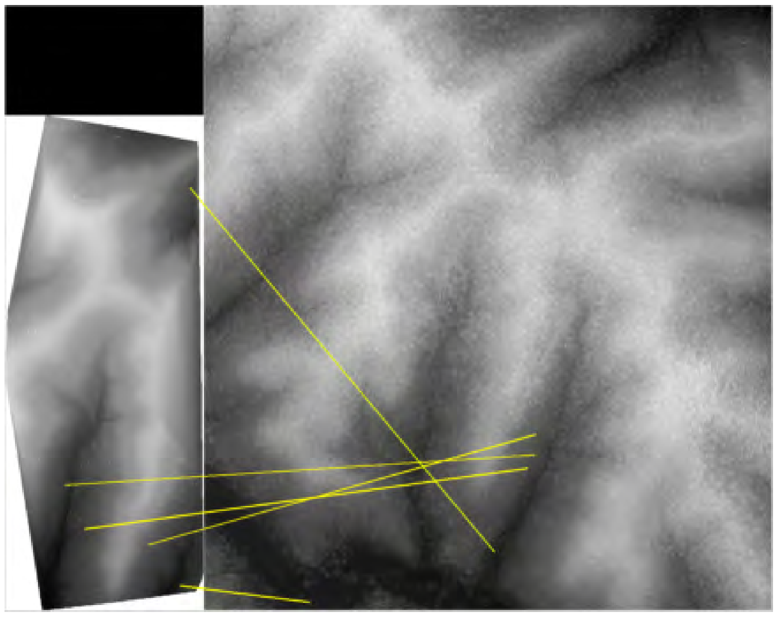


Figure A.65: This swath failed because of too few features in forest, only 1 appears correct.

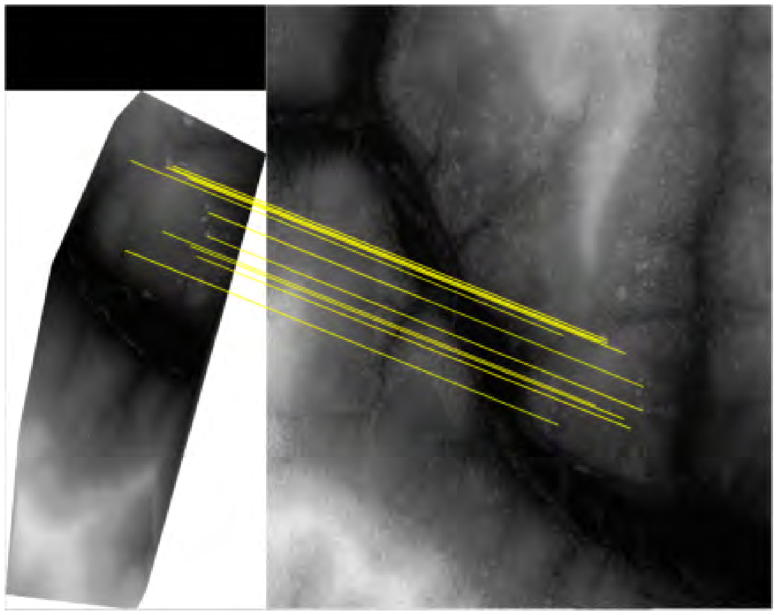


Figure A.66: This swath matched 13 features in forest and farmland for an 3D aircraft position magnitude error of 7.49 meters. Close inspection shows 2 matches are weak outliers.

Bibliography

[1] Abdelrahman, Mostafa, Asem Ali, Shireen Elhabian, and Aly A Farag. "Solving Geometric Co-Registration Problem of Multi-Spectral Remote Sensing Imagery Using SIFT-Based Features Toward Precise Change Detection". *Advances in Visual Computing*, 607–616. Springer, 2011.

[2] Abedini, Abbas, Michael Hahn, and Farhad Samadzadegan. "An Investigation into the Registration of LIDAR Intensity Data and Aeral Images Using the SIFT Approach". *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVII. Part B1:169–175, 2008.

[3] Ai, Tinghua and Jingzhong Li. "A DEM Generalization by Minor Valley Branch Detection and Grid Filling". *ISPRS Journal of Photogrammetry and Remote Sensing*, 65(2):198–207, 2010.

[4] Ali, Tarig and Ali Mehrabian. "A novel Computational Paradigm for Creating a Triangular Irregular Network (TIN) from LiDAR data". *Nonlinear Analysis: Theory, Methods and Applications*, 71(12):624–629, 2009.

[5] Alix, Dan, Karl Walli, and John Raquet. "Error Characterization of Flight Trajectories Reconstructed using Structure from Motion". *Applied Imagery Pattern Recognition Workshop: Sensing for Control and Augmentation, 2013 IEEE (AIPR*, 1–15. IEEE, 2013.

[6] Allen, Andrew CM, Christopher Langley, Raja Mukherji, Manny Nimelman, Jean de Lafontaine, David Neveu, and Jeffrey W Tripp. "Full-Scale Testing and Platform Stabilization of a Scanning Lidar System for Planetary Landing". *SPIE Defense and Security Symposium*, 696004–696004. International Society for Optics and Photonics, 2008.

[7] Atapattu, Suresh A. "The Douglas DC-3", March 2014. URL http://www.airliners.net/aircraft-data/stats.main?id=188.

[8] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded Up Robust Features". *Computer Vision–ECCV 2006*, 404–417. Springer, 2006.

[9] Besl, Paul J and Neil D McKay. "Method for Registration of 3-D Shapes". *Robotics-DL tentative*, 586–606. International Society for Optics and Photonics, 1992.

[10] Bodensteiner, Christoph, Wolfgang Huebner, Kai Jüngling, Jürgen Müller, and Michael Arens. "Local Multi-Modal Image Matching Based on Self-Similarity". *Image Processing (ICIP), 2010 17th IEEE International Conference on*, 937–940. IEEE, 2010.

[11] Britting, Kenneth R. *Intertial Navigation System Analisys*. GNSS Technology and Applications. Artech House, 2010.

[12] Bryson, M. and S. Sukkarieh. "An Information-Theoretic Approach to Autonomous Navigation and Guidance of an Uninhabited Aerial Vehicle in Unknown Environments". *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 3770–3775. August 2005.

[13] Campbell, Jacob, M Uijt de Haag, Frank van Graas, and Steve Young. "Light Detection and Ranging-Based Terrain Navigation – A Concept Exploration". *Proceedings of the 16th International Technical Meeting of the Satellite Division of the Institute of Navigation*, 461–469. ION GPS/GNSS, Portland, OR, September 2003.

[14] Chen, Yang and Gérard Medioni. "Object Modelling by Registration of Multiple Range Images". *Image and vision computing*, 10(3):145–155, 1992.

[15] Congalton, Russell G. "LAS Specification Version 1.4–R13", July 2013. URL http://www.asprs.org/a/society/committees/standards/LAS_1_4_r13.pdf.

[16] Craymer, M., R. Ferland, and R. Snay. "Realization and Unification of NAD83 in Canada and the U.S. via the ITRF". *Towards an Integrated Global Geodetic Observing System (IGGOS)*, volume 120 of *International Association of Geodesy Symposia*, 118–121. Springer Berlin Heidelberg, 2000.

[17] Detto, Matteo, Helene C Muller-Landau, Joseph Mascaro, and Gregory P Asner. "Hydrological Networks and Associated Topographic Variation as Templates for the Spatial Organization of Tropical Forest Vegetation". *PloS one*, 8(10):e76296, 2013.

[18] Diel, D.D., P. DeBitetto, and S. Teller. "Epipolar Constraints for Vision-Aided Inertial Navigation". *Application of Computer Vision, 2005. WACV/MOTIONS '05 Volume 1. Seventh IEEE Workshops on*, volume 2, 221–228. January 2005.

[19] El Mokni, H. and F. Govaers. "Coupled Laser Inertial Navigation System for Pedestrian Tracking". *Positioning Navigation and Communication (WPNC), 2011 8th Workshop on*, 176–179. April 2011.

[20] Eo, Yang Dam, Mu Wook Pyeon, Sun Woong Kim, Jang Ryul Kim, and Dong Yeob Han. "Coregistration of Terrestrial Lidar Points by Adaptive Scale-Invariant Feature Transformation with Constrained Geometry". *Automation in Construction*, 25:49–58, 2012.

[21] Fischler, Martin A and Robert C Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". *Communications of the ACM*, 24(6):381–395, 1981.

[22] Golden, Joe P. “Terrain Contour Matching (TERCOM): A Cruise Missile Guidance Aid”. *Proc. SPIE 0238, Image Processing for Missile Guidance*, volume 10. December 1980.

[23] Gonzalez, Rafael C. and Richard E. Woods. *Digital Image Processing*. Prentice Hall, second edition, 2002.

[24] Grejner-Brzezinska, Dorota, Charles Toth, Lee Young-Jin, and Jaehong Oh. “Aerial Navigation in GPS-denied Environments using a Closed-feedback Error Loop Between the Navigation and Imaging Sensors”. *Proceedings of the 21st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS 2008)*, 2638–2644. Savannah, GA, September 2008.

[25] Grejner-Brzezinska, Dorota A, Charles K Toth, Hongxing Sun, Xiankun Wang, and Chris Rizos. “A Robust Solution to High-Accuracy Geolocation: Quadruple Integration of GPS, IMU, Pseudolite, and Terrestrial Laser Scanning”. *Instrumentation and Measurement, IEEE Transactions on*, 60(11):3694–3708, 2011.

[26] Grewal, M. S., L. R. Weill, and A. P. Andrews. *Global Positioning Systems, Inertial Navigation, and Integration*, chapter Appendix C: Coordinate Transformations, 324–369. John Wiley & Sons, Inc., New York, USA, 2002.

[27] Uijt de Haag, M., A. Vadlamani, J.L. Campbell, and J. Dickman. “Application of Laser Range Scanner Based Terrain Referenced Navigation Systems for Aircraft Guidance”. *Electronic Design, Test and Applications, 2006. DELTA 2006. Third IEEE International Workshop on*, 268–274. January 2006.

[28] Ujit de Haag, Maarten, Don Venable, and Mark Smearcheck. “Integration of an Inertial Measurement Unit and 3D Imaging sensor for Urban and Indoor Navigation of Unmanned Vehicles”. *Proceedings of the National Technical Meeting of The Institute of Navigation*, 829–840. The Institute of Navigation, San Diego, CA, January 2007.

[29] Habib, Ayman, K Bang, Ana Paula Kersting, and Dong-Cheon Lee. “Error Budget of LiDAR Systems and Quality Control of the Derived Data”. *Photogrammetric Engineering and Remote Sensing*, 75(9):1093–1108, 2009.

[30] Habib, Ayman, Ki In Bang, Ana Paula Kersting, and Jacky Chow. “Alternative Methodologies for LiDAR System Calibration”. *Remote Sensing*, 2(3):874–907, 2010.

[31] Hamel, Jean-Francois, David Neveu, and Jean de Lafontaine. “Feature Matching Navigation Techniques for Lidar-Based Planetary Exploration”. *AIAA Guidance, Navigation and Control Conference and Exhibit*. 2006.

[32] Hebel, Marcus and Uwe Stilla. “LiDAR-Supported Navigation of UAVs Over Urban Areas”. *Surveying and Land Information Science*, 70(3):139–149, 2010.

[33] Hollowell, J. "Heli/SITAN: A Terrain Referenced Navigation Algorithm for Helicopters". *Position Location and Navigation Symposium, 1990. Record. The 1990's - A Decade of Excellence in the Navigation Sciences. IEEE PLANS '90., IEEE*, 616–625. March 1990.

[34] Honeywell. "HG1700 Inertial Measurment Unit", 2012. URL http://www51.honeywell.com/aero/common/documents/myaerospacecatalog-documents/Missiles-Munitions/HG1700_Inertial_Measurement_Unit.pdf.

[35] Horn, Berthold KP. "Closed-Form Solution of Absolute Orientation Using Unit Quaternions". *JOSA A*, 4(4):629–642, 1987.

[36] Horn, Berthold KP, Hugh M Hilden, and Shahriar Negahdaripour. "Closed-Form Solution of Absolute Orientation Using Orthonormal Matrices". *JOSA A*, 5(7):1127–1135, 1988.

[37] Horton, Todd W. "Understanding Adjustments of NAD83 and State Plane Coordinates", February 2012. URL http://www.iplsa.org/docs/handouts/S1015B_Horton_NAD83.pdf.

[38] Jasiewicz, Jarosaw and Tomasz F Stepinski. "Geomorphonsa Pattern Recognition Approach to Classification and Mapping of Landforms". *Geomorphology*, 182:147–156, 2013.

[39] Jiaping Zhao, Suya You. "Road Network Extraction from Airbone LiDAR Data using Scene Context". *International Workshop on Point Cloud Processing*. June 2012.

[40] Johnson, A.E. and A. Miguel San Martin. "Motion Estimation from Laser Ranging for Autonomous Comet Landing". *Robotics and Automation. Proceedings. ICRA '00. IEEE International Conference on*, volume 1, 132–138. 2000.

[41] Johnson, Andrew, Reg Willson, Yang Cheng, Jay Goguen, Chris Leger, Miguel Sanmartin, and Larry Matthies. "Design Through Operation of an Image-Based Velocity Estimation System for Mars Landing". *International Journal of Computer Vision*, 74(3):319–341, 2007.

[42] Johnson, Andrew E and Tonislav I Ivanov. "Analysis and Testing of a LIDAR-Based Approach to Terrain Relative Navigation for Precise Lunar Landing". *Proc. AIAA Guidance Navigation and Control Conference (AIAA-GNC 2011)*. 2011.

[43] Johnson, Andrew E and James F Montgomery. "Overview of Terrain Relative Navigation Approaches for Precise Lunar Landing". *Aerospace Conference, 2008 IEEE*, 1–10. IEEE, 2008.

[44] Kalman, Rudolph Emil. "A New Approach to Linear Filtering and Prediction Problems". *Journal of basic Engineering*, 82(1):35–45, 1960.

[45] Kaplan, E. D. and C. J. Hegarty. *Understanding GPS, Principles and Applications*. Artech House, second edition, 2006.

[46] Ke, Yan and Rahul Sukthankar. "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors". *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, II–506. IEEE, 2004.

[47] Kim, Jungho, Ouk Choi, and In So Kweon. "Efficient Feature Tracking for Scene Recognition Using Angular and Scale Constraints". *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, 4086–4091. IEEE, 2008.

[48] de Lafontaine, Jean, Arkady Ulitsky, Jeffrey W Tripp, Robert Richards, Michael Daly, and Christian Sallaberger. "LAPS: The Development of a Scanning Lidar System with GNC for Autonomous Hazard Avoidance and Precision Landing". *Proceedings of SPIE*, volume 5418, 81–93. 2004.

[49] Laky, Sandor, Piroska Zaletnyik, C Toth, and Bence Molnar. "Sparse Representation of Full Waveform Lidar Data". *Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International*, 7496–7499. IEEE, 2012.

[50] Legat, Klaus. "Approximate Direct Georeferencing in National Coordinates". *ISPRS Journal of Photogrammetry and Remote Sensing*, 60(4):239–255, 2006.

[51] Leonard, J.J. and H.F. Durrant-Whyte. "Simultaneous Map Building and Localization for an Autonomous Mobile Robot". *Intelligent Robots and Systems '91. 'Intelligence for Mechanical Systems, Proceedings IROS '91. IEEE/RSJ International Workshop on*, 1442–1447 vol.3. Nov 1991.

[52] Lingua, Andrea, Davide Marenchino, and Francesco Nex. "Performance Analysis of the SIFT Operator for Automatic Feature Extraction and Matching in Photogrammetric Applications". *Sensors*, 9(5):3745–3766, 2009.

[53] Liu, Yufeng and S. Thrun. "Results for Outdoor-SLAM Using Sparse Extended Information Filters". *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on*, volume 1, 1227–1233. September 2003.

[54] Lo, Tsz-Wai Rachel and J Paul Siebert. "Local Feature Extraction and Matching on Range Images: 2.5D SIFT". *Computer Vision and Image Understanding*, 113(12):1235–1250, 2009.

[55] Lowe, David G. "Distinctive Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision*, 60:91–110, 2004.

[56] Luke, Robert H, James M Keller, Marjorie Skubic, and Steven Senger. "Acquiring and Maintaining Abstract Landmark Chunks for Cognitive Robot Navigation". *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, 2566–2571. IEEE, 2005.

[57] Ma, Yi, Stefano Soatto, Jana Kosecka, and S. Shankar Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models*, volume 26. springer, 2004.

[58] Markiel, J. N., Dorota Grejner-Brzezinska, Charles Toth, William Woodward, and Jim Moore. "Underwater Mapping and Navigation: Applications of 3D Feature Extraction Algorithms to 3D Sonar Datasets". *Proceedings of the International Technical Meeting of The Institute of Navigation*, 448–458. The Institute of Navigation, San Diego, CA, January 2010.

[59] Maybeck, Peter S. *Stochastic Models, Estimation, and Control*, volume 3. Academic press, 1982.

[60] McManus, Colin, Paul Furgale, and Timothy D Barfoot. "Towards Lighting-Invariant Visual Navigation: An Appearance-Based Approach Using Scanning Laser-Rangefinders". *Robotics and Autonomous Systems*, 61(8):836–852, 2013.

[61] Meisels, Amnon, Sonia Raizman, and Arnon Karnieli. "Skeletonizing a DEM into a Drainage Network". *Computers & Geosciences*, 21(1):187–196, 1995.

[62] Mesas-Carrascosa, Francisco Javier, Isabel Luisa Castillejo-Gonzalez, Manuel Sanchex de La Orden, and Alfonso Garcia-Ferrer Porras. "Combining LiDAR Intesity with Aerial Camera Data to Discriminate Agricultural Land Uses". *Computers and Electronics in Agriculture*, 84:36–46, February 2012.

[63] Micheals, Ross J and Terrance E Boult. "On the Robustness of Absolute Orientation". *Proceeding of the International Association for Science and Technology Development (IASTED) Conference on Robotics and Automation*. 2000.

[64] Morel, Jean-Michel and Guoshen Yu. "ASIFT: A New Framework for Fully Affine Invariant Image Comparison". *SIAM Journal on Imaging Sciences*, 2(2):438–469, 2009.

[65] Muja, Marius and David G. Lowe. "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration". *International Conference on Computer Vision Theory and Application VISSAPP'09)*, 331–340. INSTICC Press, 2009.

[66] Muss, J. D., N. Aguilar-Amuchastegui, D. J. Mladenoff, and G. M. Henebry. "Analysis of Waveform Lidar Data Using Shape-Based Metrics". *Geoscience and Remote Sensing Letters, IEEE*, 10(1):106–110, January 2013.

[67] National Oceanic and Atmospheric Administration (NOAA) Costal Services Center. "Lidar 101: An Introduction to Lidar Technology, Data, and Applications". Revised, Charleston, SC: NOAA Costal Services Center, November 2012.

[68] Newman, P. and H. Durrant-Whyte. "Using Sonar in Terrain-Aided Underwater Navigation". *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, volume 1, 440–445. May 1998.

[69] NIMA Technical Report TR8350.2. *Department of Defense World Geodetic System 1984, Its Definition and Relationships With Local Geodetic Systems*. Technical Report 3rd ed, National Imagery and Mapping Agency, June 2004.

[70] Nixon, Mark S. and Alberto S. Aguado. *Feature Extraction and Image Processing for Computer Vision*. Academic Press, Elsevier, 3rd edition, 2012.

[71] Noureldin, Aboelmagd, TashfeenB. Karamat, and Jacques Georgy. "Basic Navigational Mathematics, Reference Frames and the Earths Geometry". *Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration*, 21–63. Springer Berlin Heidelberg, 2013.

[72] Oh, Jaehong, Youngjin Lee, Charles K Toth, and Dorota Brzezinska. "Explotation of LIDAR Range Measurements to Help Pushbroom Sensor Modeling for Accurate EOP Estimation". *ASPRS Annual Conference*, March 2007.

[73] Oh, Jaehong, Charles Toth, and Dorota Grejner-Brzezinska. "A Terrain Referenced Navigation Based on LiDAR Breakline Matching". *Proceedings of the 2011 International Technical Meeting of The Institute of Navigation*, 868–879. San Diego, CA, January 2011.

[74] Ohio Office of Information Technology. "Ohio Statewide Imagery Program", August 2006. URL http://ogrip.oit.ohio.gov/ProjectsInitiatives/StatewideImagery.aspx.

[75] Ojala, Timo, Matti Pietikainen, and Topi Maenpaa. "Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns". *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[76] Ordnance Survey. *D00659 A Guide to Coordinate Syatems in Great Britain v2.2*. Ordance Survey, Adanac Dr SOUTHAMTON UK, SO16 0AS, December 2013. URL http://www.ordnancesurvey.co.uk/docs/support/guide-coordinate-systems-great-britain.pdf.

[77] Parsons, Timothy J. "Towards Robust Image Matching Algorithms". *28th Annual Technical Symposium*, 436–444. International Society for Optics and Photonics, 1984.

[78] Passalacqua, Paola, Tien Do Trung, Efi Foufoula-Georgiou, Guillermo Sapiro, and William E Dietrich. "A Geometric Framework for Channel Network Extraction from LiDAR: Nonlinear Diffusion and Geodesic Paths". *Journal of Geophysical Research: Earth Surface (2003–2012)*, 115(F1), 2010.

[79] Penn State. "Fundamental Conecpts 2: Datums and Coordinate Systems", 2006. URL https://courseware.e-education.psu.edu/courses/bootcamp/lo04/cg.html.

[80] Petrie, Gordon. "Airborne Topographic Laser Scanners", February 2011. URL http://www.riegl.com/fileadmin/user_upload/Press/Petrie_Airborne_Topographic_Laser_Scanners_GEO_1_2011.pdf.

[81] Raquet, John F and Michael Giebner. "Navigation Using Optical Measurements of Objects at Unknown Locations". *Proceedings of the 59th Annual Meeting of The Institute of Navigation and CIGTF 22nd Guidance Test Symposium*, 282–290. 2001.

[82] Rentsch, M and P Krzystek. "LiDAR strip adjustment using automatically reconstructed roof shapes". *Proceedings of International Archives of the Photogrammetry, Remote Sensing and Spatial Information*, 158–164, 2009.

[83] Richmond, Richard D. and Stephen C. Cain. *Direct-Detection LADAR Systems*, volume TT85 of *Tutorial Texts in Optical Engineering*. SPIE Press, 2010.

[84] RIEGL. "RIEGL LMA-Q680i", May 2014. URL http://www.riegl.com/nc/products/airborne-scanning/produktdetail/product/scanner/23/.

[85] Robins, Alan. "Recent Developments in the TERPROM Integrated Navigation System". *Proceeding of the ION 44th Annual Meeting*. 1998.

[86] Rueda, Antonio, José M Noguera, and Carmen Martínez-Cruz. "A Flooding Algorithm for Extracting Drainage Networks from Unprocessed Digital Elevation Models". *Computers & Geosciences*, 59:116–123, 2013.

[87] Rusinkiewicz, Szymon and Marc Levoy. "Efficient Variants of the ICP Algorithm". *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, 145–152. IEEE, 2001.

[88] Rusu, Radu Bogdan and Steve Cousins. "3D is here: Point Cloud Library (PCL)". *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, May 2011.

[89] Salvi, Joaquim, Carles Matabosch, David Fofi, and Josep Forest. "A Review of Recent Range Image Registration Methods with Accuracy Evaluation". *Image and Vision Computing*, 25(5):578–596, 2007.

[90] Sanjeev Arulampalam, M, Simon Maskell, Neil Gordon, and Tim Clapp. "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking". *Signal Processing, IEEE Transactions on*, 50(2):174–188, 2002.

[91] Sazdovski, V., P. M G Silson, and A. Tsourdos. "Inertial Navigation Aided by Simultaneous Localization and Mapping". *Intelligent Systems (IS), 2010 5th IEEE International Conference*, 43–48. July 2010.

[92] Se, Stephen, David Lowe, and Jim Little. "Global Localization Using Distinctive Visual Features". *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, 226–231. IEEE, 2002.

[93] Shan, Jie and Charles K. Toth. *Topographic Laser Ranging and Scanning, Principles and Processing*. CRC Press, first edition, 2009.

[94] Sharp, Gregory C, Sang W Lee, and David K Wehe. "ICP Registration using Invariant Features". *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(1):90–102, 2002.

[95] Sithole, George and George Vosselman. "Experimental Comparison of Filter Algorithms for Bare-Earth Extraction from Airborne Laser Scanning Point Clouds". *ISPRS Journal of Photogrammetry and Remote Sensing*, 59(1):85–101, 2004.

[96] Skaloud, Jan and Klaus Legat. "Theory and Reality of Direct Georeferencing in National Coordinates". *ISPRS Journal of Photogrammetry and Remote Sensing*, 63(2):272–282, 2008.

[97] Skaloud, Jan and Derek Lichti. "Rigorous Approach to Bore-Sight Self-Calibration in Airborne Laser Scanning". *ISPRS Journal of Photogrammetry and Remote Sensing*, 61(1):47–59, 2006.

[98] Snay, Richard A. "Using the HTDP Software to Transform Spatial Coordinates Across Time and Between Reference Frames". *Surveying and Land Infromation Systems*, 59(1):16–25, 1999.

[99] Snay, Righard A. and Tomas Soler. "Modern Terrestrial Reference Systems (Part 1)". *Professional Surveyor*, 19(10):32–33, December 1999. URL http://www.colorado.edu/geography/gcraft/notes/datum/datum.html.

[100] Soler, T. and R. Snay. "Transforming Positions and Velocities Between the International Terrestrial Reference Frame of 2000 and North American Datum of 1983". *Journal of Surveying Engineering*, 130(2):49–55, 2004.

[101] Soler, Tomas. "A Compendium of Transfomation Fomulas Useful in GPS Work". *Journal of Geodesy*, 72:482–490, April 1998.

[102] Stem, James E. *NOAA MAnual NOS NGS 5, State Plane Coordinate Systems of 1983*. US Department of Commerce, NOAA, National Ocean Service, Charting and Geodetic Services, Rockville, MD, September 1989. URL http://www.ngs.noaa.gov/PUBS_LIB/ManualNOSNGS5.pdf.

[103] Susca, Sara. "GNSS-Independent Navigation Solution Using Integrated LiDAR Data". *Proceedings of the 2010 International Technical Meeting of The Institute of Navigation*, 205–213. 2001.

[104] Tang, Wang and Robert L McClintock. "Terrain correlation suitability". *SPIE's International Symposium on Optical Engineering and Photonics in Aerospace Sensing*, 50–58. International Society for Optics and Photonics, 1994.

[105] Tarolli, Paolo and Giancarlo Dalla Fontana. "Hillslope-to-Valley Transition Morphology: New Opportunities from High Resolution DTMs". *Geomorphology*, 113(1):47–56, 2009.

[106] The National Coordination Office. "GPS Applications", September 2013. URL http://www.gps.gov/applications/.

[107] The National Geodetc Survey. "The NGS Geoid Page", December 2012. URL http://www.ngs.noaa.gov/GEOID/.

[108] The National Geodetc Survey. "HTDP - Horizontal Time-Dependent Positioning", July 2013. URL http://www.ngs.noaa.gov/TOOLS/Htdp/Htdp.shtml.

[109] The State of Ohio. "Title 1, Chapter 157: Ohio Coordinate System", October 1985. URL http://codes.ohio.gov/orc/157.

[110] Titterton, D. H. and J. L. Weston. *Strapdown Inertial Navigation Technology*, volume 207 of *Progress in Astronautics and Aeronautics*. MIT Lincoln Laboratory, second edition, 2009.

[111] Toth, C, DA Grejner-Brzezinska, JH Oh, and JN Markiel. "Terrain-Based Navigation: A Tool to Improve Navigation and Feature Extraction Performance of Mobile Mapping Systems". *Boletim de Ciências Geodésicas*, 15(5), 2009.

[112] Toth, Charles, Dorota A Grejner-Brzezinska, and Young-Jin Lee. "Terrain-Based Navigation: Trajectory Recovery from LiDAR Data". *Position, Location and Navigation Symposium, 2008 IEEE/ION*, 760–765. IEEE, 2008.

[113] Toth, Charles, Hui Ju, and Dorota Grejner-Brzezinska. "Matching between Different Image Domains". Uwe Stilla, Franz Rottensteiner, Helmut Mayer, Boris Jutzi, and Matthias Butenuth (editors), *Photogrammetric Image Analysis*, volume 6952 of *Lecture Notes in Computer Science*, 37–47. Springer Berlin Heidelberg, 2011.

[114] Toth, Charles K, Dorota A Grejner-Brzezinska, and YJ Lee. "Recovery of Sensor Platform Trajectory from LiDAR Data using Reference Surfaces". *Proceedings of the 13th FIG Symposium and the 4th IAG Symposium, Lisbon, Portugal*. 2008.

[115] Toth, Charles K, Hui Ju, and Dorota A Grejner-Brzezinska. "Experiences with using SIFT for mulitple image domain matching". *Proceedings of ASPRS 2010 Annual Conference, San Diego, CA-USA*. 2010.

[116] True, S. A. "Planning the Future of the World Geodetic System 1984". *Position Location and Navigation Symposium, 2004. PLANS 2004*, 639–648. Monterey, CA, April 2004.

[117] Ulas, Cihan and Hakan Temeltas. "A Robust Feature Extraction Method and Semantic Data Association for 6D SLAM". *World Automation Congress (WAC), 2012*, 1–6. June 2012.

[118] Umeyama, Shinji. "Least-Squares Estimation of Transformation Parameters Between Two Point Patterns". *IEEE Transactions on pattern analysis and machine intelligence*, 13(4):376–380, 1991.

[119] Ussyshkin, Valerie and Livia Theriault. "Airborne Lidar: Advances in Discrete Return Technology for 3D Vegetation Mapping". *Remote Sensing*, 3(3):416–434, February 2011.

[120] Vadlamani, A and M Uijt de Haag. "Aerial Vehicle Navigation Over Unknown Terrain Environments Using Flash LADAR and Inertial Measurement". *Proceedings of ION NTM 2007*, 22–24, 2007.

[121] Vadlamani, Ananth K and Maarten Uijt de Haag. "Dual Airborne Laser Scanners Aided Inertial for Improved Autonomous Navigation". *Aerospace and Electronic Systems, IEEE Transactions on*, 45(4):1483–1498, October 2009.

[122] Vandapel, Nicolas, Raghavendra Donamukkala, and Martial Hebert. "Experimental Results in Using Aerial Ladar Data for Mobile Robot Navigation". *Field and Service Robotics*, 103–112. Springer, 2006.

[123] Vandapel, Nicolas, Raghavendra Rao Donamukkala, and Martial Hebert. "Unmanned Ground Vehicle Navigation Using Aerial Ladar Data". *The International Journal of Robotics Research*, 25(1):31–51, 2006.

[124] Veth, Michael J. *Fusion of Imaging and Inertial sensors for navigation*. Dissertation, Air Force Insitute of Technology, September 2006.

[125] Volpe, John A. "Vulnerability Assessment of the Transportation Infrastructure Relying on the Global Positioning System", 2001. URL http://www.navcen.uscg.gov/pdf/vulnerability_assess_2001.pdf.

[126] Wan, E.A. and R. Van der Merwe. "The unscented Kalman filter for nonlinear estimation". *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, 153–158. 2000.

[127] Wang, Jianjun, Lijun Xu, Xiaolu Li, and Xiangrui Tian. "Simulation on Impact of Random Attitude Measurement Errors on Point Cloud and 3D Image of ALS". *Imaging Systems and Techniques (IST), 2011 IEEE International Conference on*, 60–64. IEEE, 2011.

[128] Young, Steven D, Maarten Uijt de Haag, and Jacob Campbell. "An X-band Radar Terrain Feature Detection Method for Low-Altitude SVS Operations and Calibration Using Lidar". *Defense and Security*, 110–124. International Society for Optics and Photonics, 2004.

[129] Yun, Sukchang, Young Jae Lee, and Sangkyung Sung. "IMU/Vision/Lidar Integrated navigation system in GNSS denied environments". *Aerospace Conference, 2013 IEEE*, 1–10. March 2013.

[130] Zamora, Erik and Wen Yu. "Recent Advances on Simultaneious Localization and Mapping for Mobile Robots". *IETE Tech Rev*, 30(6):490–496, 2013.

[131] Zandbergen, Paul A. "Positional Accuracy of Spatial Data: Non-Normal Distributions and a Critique of the National Standard for Spatial Data Accuracy". *Transactions in GIS*, 12(1):103–130, 2008.

[132] Zhang, Jun, Weisong Liu, and Yirong Wu. "Novel Technique for Vision-Based UAV Navigation". *Aerospace and Electronic Systems, IEEE Transactions on*, 47(4):2731–2741, October 2011.

[133] Zhang, Zhengyou. "Iterative Point Matching for Registration of Free-Form Curves and Surfaces". *International journal of computer vision*, 13(2):119–152, 1994.

[134] Zhao, Feng, Qingming Huang, and Wen Gao. "Image Matching by Normalized Cross-Correlation". *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 2, 729–732. IEEE, 2006.

[135] Zhou, Weizhen, Jaime Valls Miró, and Gamini Dissanayake. "Information-Efficient 3-D Visual SLAM for Unstructured Domains". *Robotics, IEEE Transactions on*, 24(5):1078–1087, 2008.

[136] Zuliani, Marco. "RANSAC for Dummies", 2009. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.139.2808&rep=rep1&type=pdf.

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB No. 0704–0188*

1. REPORT DATE *(DD–MM–YYYY)* 26–12–2014	**2. REPORT TYPE** Master's Thesis	**3. DATES COVERED** *(From — To)* Sep 2012–Dec 2014

4. TITLE AND SUBTITLE Terrain Referenced Navigation Using SIFT Features in LiDAR Range-Based Data	**5a. CONTRACT NUMBER**
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Leines, Matthew T., Capt, USAF	**5d. PROJECT NUMBER** N/A
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB, OH 45433-7765	**8. PERFORMING ORGANIZATION REPORT NUMBER** AFIT-ENG-MS-14-D-47
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Labs, Sensors Directorate Spectrum Warfare Division, Navigation & Communication Branch 2241 Avionics Circle, Write-Patterson Air Force Base, OH 45433 Phone: 937-938-4414, Email: jacob.campbell.3@us.af.mil	**10. SPONSOR/MONITOR'S ACRONYM(S)** AFRL/RYWN
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

13. SUPPLEMENTARY NOTES
This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT
The use of GNSS in aiding navigation has become widespread in aircraft. The long term accuracy of INS are enhanced by frequent updates of the highly precise position estimations GNSS provide. Unfortunately, operational environments exist where constant signal or the requisite number of satellites are unavailable, significantly degraded, or intentionally denied. This thesis describes a novel algorithm that uses scanning LiDAR range data, computer vision features, and a reference database to generate aircraft position estimations to update drifting INS estimates. The algorithm uses a single calibrated scanning LiDAR to sample the range and angle to the ground as an aircraft flies, forming a point cloud. The point cloud is orthorectifed into a coordinate system common to a previously recorded reference of the flyover region. The point cloud is then interpolated into a Digital Elevation Model (DEM) of the ground. Range-based SIFT features are then extracted from both the airborne and reference DEMs. Features common to both the collected and reference range images are selected using a SIFT descriptor search. Geometrically inconsistent features are filtered out using RANSAC outlier removal, and surviving features are projected back to their source coordinates in the original point cloud. The point cloud features are used to calculate a least squares correspondence transform that aligns the collected features to the reference features. Applying the correspondence that best aligns the ground features is then applied to the nominal aircraft position, creating a new position estimate. The algorithm was tested on legacy flight data and typically produces position estimates within 10 meters of truth using threshold conditions.

15. SUBJECT TERMS
LiDAR, Airborne, Terrain Referenced Navigation, SIFT, Features, Range Image

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Dr. John F. Raquet (ENG)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(include area code)*
U	U	U	UU	181	(937) 255-3636 x4580 John.Raquet@afit.edu

Standard Form 298 (Rev. 8–98)
Prescribed by ANSI Std. Z39.18