

AUTONOMOUS 3D RECONSTRUCTION FROM  
IRREGULAR SPACED LIDAR AND AERIAL IMAGERY

by

NICHOLAS SVEN SHORTER  
B.S.E.E. University of Central Florida, 2005  
M.S.E.E. University of Central Florida, 2006

A PhD Candidacy Proposal submitted in partial fulfillment of the requirements  
for the degree of Doctor of Philosophy of Science  
School of Electrical Engineering and Computer Science  
in the College of Engineering and Computer Science  
at the University of Central Florida  
Orlando, Florida

Spring Term  
2007

## **ABSTRACT**

As more data sources have become abundantly available, an increased interest in 3D reconstruction has emerged in the image processing academic community. Applications for 3D reconstruction of urban and residential buildings consist of urban planning, network planning for mobile communication, tourism information systems, spatial analysis of air pollution and noise nuisance, microclimate investigations, and Geographical Information Systems (GISs). Previous, classical, 3D reconstruction algorithms solely utilized aerial photography. With the advent of LIDAR systems, current algorithms explore using captured LIDAR data as an additional feasible source of information for 3D reconstruction.

This proposal presents an outline for the development of an autonomous 3D reconstruction algorithm. The algorithm will analyze key features extracted from both LiDAR data and aerial imagery of a given scene and, without user intervention, isolate building structures and then reconstruct 3D models depicting those building structures. The motivations behind already determining several key characteristics about the design of the reconstruction algorithm (such as using a data dependent as opposed to model based algorithm) are presented. Comparisons to previous works contrasting design methodologies are also included.

## TABLE OF CONTENTS

ABSTRACT.....	2
TABLE OF CONTENTS.....	3
CHAPTER ONE: INTRODUCTION TO LIDAR AND 3D RECONSTRUCTION.....	5
1.1 Applications of 3D Reconstruction.....	5
1.2 LIDAR Overview.....	6
1.3 LiDAR Noise Sources and Limitations .....	8
1.4 Problem Statement.....	10
CHAPTER TWO: EXISTING RECONSTRUCTION METHODOLOGIES .....	12
2.1 Data Driven and Model based Methods.....	12
2.2 Triangulation Methodologies.....	16
2.2 Interpolating Irregular Raw LIDAR to Fixed Intervals .....	19
CHAPTER THREE: ALGORITHM DESIGN.....	22
3.1 Algorithm Functional Block Diagram .....	22
3.2 Building Detection.....	23
3.3 Building Extraction.....	24
3.4 Building Reconstruction .....	26
3.5 Image Mapping .....	27
3.6 Algorithm Novelty.....	28
CHAPTER FOUR: MILESTONES.....	30
APPENDIX A: SEQUENTIAL GREEDY INSERTION STEPS .....	31
APPENDIX B: PLANAR COEFFICIENTS .....	33

APPENDIX C: DELAUNAY TRIANGULATION CIRCLE TEST .....	36
APPENDIX D: PERPENDICULAR DISTANCE FROM AN ARBITRARY POINT TO A GIVEN PLANE .....	40
APPENDIX E: GREEDY INSERTION TRIANGULATION .....	44
APPENDIX F: GREEDY INSERTION TRIANGULATION FILTERING MODIFICATION	50
LIST OF REFERENCES .....	55

# **CHAPTER ONE: INTRODUCTION TO LIDAR AND 3D RECONSTRUCTION**

The concept of deriving three-dimensional models from various sources of data has existed for several decades now. Known as the 3D Reconstruction problem, methodologies for solving this problem and even extending its application have evolved with the advent of new technologies which deliver new and/or improved sources of data. The research presented focuses on 3D Reconstruction via primarily using Light Detection and Ranging (LIDAR) data and aerial photographs.

First, various applications of 3D reconstruction are introduced. Second, a basic overview of LIDAR is presented. Third, the various sources of noise in which corrupt the data sources utilized for 3D reconstruction are discussed. Finally, the specific problem in which this area of research addresses is developed.

## **1.1 Applications of 3D Reconstruction**

Applications of 3D Reconstruction have valued use for both militaristic and commercial purposes. For the military, the analysis of LiDAR data can be used for target recognition applications. Work in training volume correlation filters (VCFs) to recognize tanks and other military vehicles within LiDAR data has recently been developed [33]. There is investigation underway for mounting LiDAR sensors on unmanned aerial vehicles (UAVs) [29]. This would enable aerial surveying of terrains in which military forces were denied access too. Scenes surveyed by an UAV or high flying plane with a LiDAR sensor could then be reconstructed into 3D models.

For commercial uses, the demand for 3D models of buildings has applications such as urban planning, network planning for mobile communication, spatial analysis of air pollution and noise nuisances, microclimate investigations, geographical information systems, and security services. For entertainment purposes, 3D reconstruction can be used for tourism information systems. Tourists, instead of using 2D maps to find their way around theme parks, could use a kiosk to view a virtual 3D walk through of the park. Also, 3D Reconstruction has change detection applications [51], where buildings demolished by natural disasters, are automatically identified via histogram difference thresholding comparing a pre and post event LiDAR analysis.

## **1.2 LIDAR Overview**

Mounted on the aircraft, helicopter or plane, collecting the LIDAR data, is a Global Positioning System (GPS), an Inertial Navigation System (INS) and a LIDAR sensor system. The GPS returns the longitude and latitude coordinates of the aircraft's current position. The INS tracks the altitude of the LIDAR sensor. The LIDAR sensor system is an active remote sensing instrument which consists of an emitter and receiver. The emitter sends out a pulse of light (electromagnetic radiation) into the atmosphere. The telescope (receiver) measures the intensity of the signal scattered back to the sensor system after that signal has interacted with various constituents in the atmosphere [48],[56]. The time from departure to return is also recorded thus enabling the calculation of the distance of the sensor to the target (or in this case the sampled terrain).

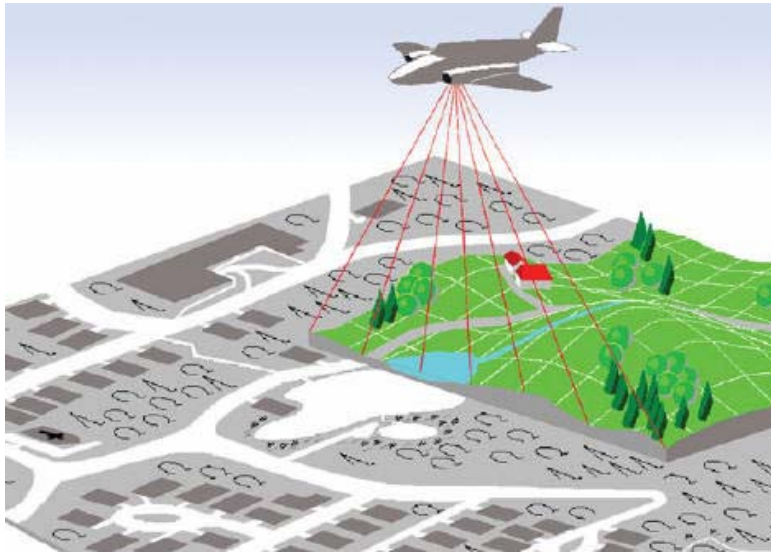


Figure 1: Capturing LIDAR Data

In the case of a building surface, the laser will reflect off of the building's surface and return to the sensor. In the case of tree foliage or vegetation, two possible scenarios arise: the laser beam could pass through the foliage or vegetation and hit the ground, or the laser could interact with the foliage or vegetation. The first pulses to return to the sensor are labeled First Return Pulses. These pulses consist of laser beams which interacted with the top of foliage and vegetation and building structures. The last pulses to return to the sensors are labeled Last Return Pulses. These pulses consist of laser beams which passed through foliage and/or vegetation and interacted with the ground. These pulses also consist of laser beams which interacted with building surfaces. Based on the time it takes from the emission of the laser from the sensor to the return of the laser beam after it has interacted with a given target, the range from the sensor to the target can be calculated. Differences between first and last return pulses for vegetation and building edges are relatively high. Differences between first and last return pulses for building surfaces are relatively low. When the laser beam, emitted, from the LIDAR

sensor hits a building surface, the majority of the beam is immediately reflected leaving minimal difference between first and last returns.

While some primitive LIDAR systems only return the longitude, latitude, and elevation of a given returned point, newer systems can capture sampling time, longitude, latitude, elevation, the intensity of the returned signal, and the first and last return pulses. The data set considered for this research set does contain sampling time, longitude, latitude, returned signal intensity, first and last return pulse attributes and is commonly referred to as 'Fairfield' test set. The 'Fairfield' test set, provided by Dr. Simone Clode and Dr. Franz Rottensteiner and procured by AAMHatch, covers two square kilometers of both an urban and residential area of Fairfield, Australia. The data set comes with both the LIDAR data, in ASCII format, and a corresponding aerial photograph of the terrain. The aerial photograph has 15 centimeter pixel resolution. This means that each pixel in the aerial photograph corresponds to 15 centimeters in the actual terrain depicted. The LIDAR data has approximately a 1 point per  $1.3\text{m}^2$  point spacing density.

### **1.3 LiDAR Noise Sources and Limitations**

Several phenomena contribute as sources of noise during the processes involved in which the LiDAR data is procured. Furthermore, not only do various sources of noise corrupt the captured data, but also limitations within the sensors themselves further distort the data. It is imperative to understand these sensor limitations and noise sources so that they can intelligently be accounted for.

For the Fairfield test set, Simone et. al. in [17] have reported that first and last returns differing in less than 4.6 meters in elevation are not valid. The reason for this was found to be a

limitation in the LIDAR sensor itself. The sensor had to reset itself before a second return could be recorded. If a second return comes back to the sensor before the reset time has passed, a dual return was being recorded. Hence if the first and last return were less than 4.6 meters apart, the two returns arrived back at the laser before it could reset itself in time to record the second return, resulting in the LIDAR system simply recording the same return for both first and last return pulses. The algorithm proposed will make use of both returns by only triangulating both returns if their elevation difference is greater than 4.6 meters.

There is a plethora of noise sources influencing the processes involved with procuring LiDAR data. Filin reports that because the geolocation of a single point results from the integration of three subsystems (GPS, INS, and LiDAR), it is possible that errors can come from any combination of all three sensors [18]. Specifically, errors resulting from the three subsystems include but are not limited to the following: a constant offset in the range determination, inaccurate scan angle determination, mounting bias from the misalignment between the INS and LiDAR sensor, GPS offset and drift, and INS system drift. These reported errors only exist as systematic discrepancies between the integration and limitations of the sensors themselves. As the electromagnetic radiation that is the laser is propagated through the atmosphere both its intensity and path are distorted by the interaction between the beam and the atmosphere itself [48],[56]. An additional aspect in which influences the LiDAR procurement process is artifacts resulting from scan angles not uniformly interacting with the sampled terrain due to obstructions from the terrain itself. Consider as an aircraft flies adjacent to a tall skyscraper, the laser pulses will interact with the side of that tall skyscraper but not with the terrain behind it resulting in a shadowing effect [51].

## **1.4 Problem Statement**

The problem in which the 3D reconstruction academia attempts to tackle is the 3D reconstruction of models based on multiple sources of data. The 3D reconstruction aims to tactically fuse independent, correlated forms of data to derive the most accurate 3-dimensional model from the depicted sources' data. Ideally, the algorithm will fuse as many data sources that are available and perform the 3D reconstruction autonomously with no user intervention (this includes the adjustment of parameters from building to building or adjustment based on data set characteristics, ie maximum building size). One can think of 3D reconstruction as a black box with the input as a collection of one or more of the following sources of data: LIDAR data, aerial photography and GIS plans. The outputs of this box therefore are virtual, 3D models of the terrain depicted by the original sources of input data.

The ideal 3D reconstruction algorithm would have the following attributes. The algorithm should be able to intelligently recognize both complex (non planar) building structures as well as simple ones. The algorithm should be generalized such that it is capable of analyzing LiDAR in the most general case which is raw (irregularly spaced) LiDAR data as well as the specific case where the LiDAR data is rasterized / grid-ed. The algorithm should ideally be able to incorporate sources as they are available. For example, the algorithm should start with a fundamental requirement, raw LiDAR data with at least  $1.5 \text{ m}^2$  point density, and then be capable of including additional sources of information: aerial imagery, multispectral imagery and geographic information system ground plans. The algorithm should reconstruct the buildings as a collection of simple structures (roof planes, domes, walls, etc.) as opposed to a collection of LiDAR points or triangles existent in a triangular mesh. The reconstructed

structural entities (walls, roofs, etc.) should be identified as members of a group corresponding to a single structure (commercial building, house, etc.) existent in the real scene. As part of identifying those structural entities as members to a building structure, the buildings depicted in the scene should therefore be automatically isolated as different buildings by the algorithm. Due to the large nature of data sets and the consumer need for the algorithm to operate without a trained technician, the level of automation that the algorithm should operate at should be such that minimal or ideally no user intervention is necessary. Therefore, ideally, no a priori information, no parameter adjustment, no user interaction aiding reconstruction steps, should be required by the ideal algorithm. If both the aerial imagery is available as well as the LiDAR data, then the reconstructed models should have the aerial images projected onto the models for a more appealing reconstruction.

The research presented in this proposal aims to complete all of the above ideal attributes save the multiple input sources as available. Due to GIS ground plans being rarely available for certain given areas in conjunction with aerial imagery and LiDAR data almost always being readily available, the following input restrictions are mandated. The proposed algorithm will require both LiDAR data of at least 1.5 m<sup>2</sup> point density and aerial imagery of at least 25 cm pixel resolution.

## CHAPTER TWO: EXISTING RECONSTRUCTION METHODOLOGIES

### 2.1 Data Driven and Model based Methods

Several defining traits characteristic of a given 3D reconstruction algorithm distinguishes it from other algorithms, traits such as the following: the sources of data the algorithm operates on and the technique the algorithm utilizes to approximate the building surface.

Some algorithms only consider the use of stereo pairs of aerial images procured from satellites, planes and helicopters. A stereo pair of images is two images having a significant amount of overlap depicting the same scene. Huguet et. al. develop a building segmentation method called Color-Based Watershed Segmentation in [24] and employ it to realize 3D Reconstruction of urban scenes from low altitude images in [25]. However, several portions of their algorithm are still undergoing extensive experimental validation. In [49], Suveg and Vosselman present an autonomous 3D reconstruction algorithm which uses a set of basic building models to approximate buildings depicted in a sequence of images and 2D GIS maps (which contains building outlines or footprints). The algorithm presented in that paper assumes all buildings can be reconstructed from simple building models with flat, gable or hip roof. Furthermore, the algorithm also assumes that the 2D GIS building footprint maps are available for a given area and that those buildings can be partitioned into a collection of rectangles.

The advent of LIDAR sensor systems created a whole new genre of 3D reconstruction algorithms which made use of LIDAR data as an additional data source. Another model based approximation algorithm, processing LIDAR data instead of images and GIS maps, is presented in [34] by Mass and Vosselman. As with [49], model based reconstruction assumes the given

depicted building can be accurately approximated by a combination of pre-existing building model via adjusting parameters associated with those models. Hu et. al. in [22] use both LiDAR and aerial imagery to realize model based reconstruction. While their algorithm is capable of successfully reconstructing non planar, irregular buildings; LiDAR point rasterizing and user intervention (during several steps) is necessary for their algorithm.

In contrast to model based LIDAR 3D Reconstruction approaches, several data driven approaches have been developed. These data driven 3D Reconstruction algorithms typically begin by separating building points from non building points. They then group like points together and then derive a model to approximate those points that yields minimum error from the original points. This approach approximates segmented areas with planes and sometimes is able to handle other 3D shapes. Then data driven approaches typically merge those 3D entities to form three-dimensional structures depicting the captured LIDAR scene. The quintessential difference between the two approaches (model and data) is that the data driven approach forms its approximating virtual model from the LiDAR data while the model based approach fits a collection of pre-existing models to the LiDAR data.

Rottensteiner and Briese in [39] construct buildings from LIDAR data by detecting characteristics in the data that delineate buildings from their surroundings and then detecting characteristics specific to those buildings. The LIDAR data is separated into building and non-building regions via the algorithm described in [38]. This process is implemented by using morphological filters for computing a digital terrain model (DTM) and then applying a thresholding technique to the height differences between the DTM and the digital surface model (DSM). Note that in a DTM and DSM the LIDAR data is interpolated to fixed point spacings. The DSM is a model depicting both terrain and non terrain (building) points. By using

morphological filters and other filtering techniques, everything but the terrain is removed from the DSM, thus creating a digital terrain model (DTM). Then roof planes are detected via a curvature based segmentation technique, and then grouped into polyhedral building models.

As in [39], Chen et. al. in [16], follow a similar procedure. First, ground from non ground interpolated points are separated. Then ‘building regions’ are segmented and coplanarity is analyzed to shape the roof of the building regions. Finally a patented Split-Merge-Shape (SMS) method is employed to create building models from the aforementioned gathered information. While the roofs are generated from the raw LIDAR data, building and non-building classification is still done with DTM and DSM thresholding techniques.

Fujii and Arikawa use both LIDAR data and aerial images in [19]. First LIDAR data, interpolated at fixed intervals, is analyzed for line segments forming object contours. Identifying these contours as buildings makes way for building extraction. Contours in the LIDAR data are then registered with those of aerial imagery of the same scene and then texture mapping from the imagery onto the LIDAR data occurs. A voting technique using the Hough transform is implemented to minimize mismatching.

In [32], Overby et. al make use of a three-dimensional extension of the Hough transform for extracting planes from point cloud data. Geometric and other constraints further refine those planes and vote whether or not to reject them. These planes are then merged to form three-dimensional models.

There is a genre of data dependant algorithms which focus on a different manner of analyzing strictly the LiDAR points. Rather than analyzing the points, these reconstruction algorithms instead analyze triangles which the LIDAR points form. Several data driven LIDAR model 3-D reconstruction algorithms currently exist in the literature ([36],[21],[32],[31]) which

utilize triangulated irregular networks (TINs) to construct model approximations of depicted urban and residential scenes. Triangulated irregular networks are a 3-dimensional depiction of LIDAR point cloud data represented with a series of connected, non-overlapping triangles which have no intersecting edges. Three methods of utilizing the TIN structure to extract information from LIDAR point cloud data exist as follows: clustering approach; least squares approximation approach; TIN region growing algorithm approach.

The following methods use the least squares approximation in conjunction with the TIN region growing to extract 3-D features from the point cloud data. Morgan and Habib in [36] use a region growing TIN algorithm, based on the least-squares adjustment, to extract building facades from the transformed point cloud data (transformed to the triangulated feature space). Chen et. al., in [32], also use a region growing TIN algorithm, considering both the height difference between triangles and the angle difference between normal vectors of neighboring triangles for merging criterion for planar approximation. In region growing approaches, the normal vector of a considered triangle is analyzed. If the normal vectors of triangles adjacent to the originally considered triangle fall within a certain threshold from the normal vector of the originally considered triangle, the adjacent triangles are then clustered to the same label as that of the originally considered triangle. Then other adjacent triangles are checked and the process repeats. If the normal vectors of the adjacent triangles ever exceed the threshold in comparison to the normal vectors of the originally considered triangles, then a new cluster label is created.

Hoffman however uses the clustering approach in [21] to group together triangles in the TIN that contain similar properties. In [21], the position of each triangle is mapped out in spherical coordinates which are the dimensions of the triangles that are clustered.

Lattuada et. al. in [31] detail several advantages for describing a geo-model with a three dimensional triangulation. These details have been enumerated in [Table 1].

Table 1: Advantages of Representing LIDAR as a TIN

3-D Triangulated Irregular Networks Advantages	
1	the generation algorithm is fully automatic and therefore objective
2	space is uniquely defined and cells are spatially indexed
3	size of elements can be adjusted locally as a function of the complexity of the model
4	the model can easily be edited manually
5	topology is derived from neighborhood relationships
6	constrained triangulation means we can use vectors or surface constrains (i.e. to represent trends)
7	use of triangular elements is the perfect choice for visualization since this is the basis for rendering techniques
8	good accuracy and approximation compared to block models
9	integral properties are efficient and easy to calculate
10	we can easily extract from the 3D solid representation of an object the 3D triangulated surface which is its boundary
11	spatial searches and relational queries are easy to implement
12	good performance of Boolean operations

## **2.2 Triangulation Methodologies**

Several different methodologies exist for triangulating a dataset. One of the most popular techniques, the Delaunay triangulation, attempts to maximize the lesser two internal angles in a given triangle for all triangles. A detailed derivation of the equations associated with the Delaunay triangulation circle test is presented in section C of the appendix.

For only 2 dimensions, the Delaunay triangulation method, given four points, chooses the diagonal that that splits the quadrilateral formed by the four points into two triangles. These triangles are such that the lesser of their internal angles are maximized. However, as already mentioned, this property of Delaunay triangulation, as proved by [8], only holds in 2 dimensions. It is possible to produce a Delaunay triangulation for a 3-dimensional data set; the z-coordinate is

simply ignored. Methods that consider the z-coordinate for triangulation are referred to as data dependent triangulation methods.

Wang et. al. in [52] compare the Delaunay triangulation process against several other triangulation processes when approximating two different terrains from Digital Surface Models (DSMs). Several conclusions derived from the paper are presented. The quality of the generated TIN is dependent on both the vertex placement and connection. Processes that iteratively select points during triangulation grossly outperform processes that separate the point selection and triangulation procedures. While TIN generation from separate procedures is comprehended and implemented with ease, the separation of the procedures suffers from the following drawbacks. Point selection via filters is very sensitive to data errors and surface variations. Point selection is a static process (as opposed to the dynamic adaptive process). When a point is chosen and inserted, the configuration of the TIN is modified and therefore the importance of the remaining unused points changes. All computational efforts executed to find the surface specific points are not utilized in the final construction of the TIN and are thus wasted. Therefore the implementation of an algorithm integrating point selection and triangulation as a unified procedure, while being more complex than algorithms that separate the procedures, is preferred due to the increase in performance and the ability of this preferred methodology to overcome the aforementioned drawbacks.

Among all of the triangulation methods tested, Wang et. al. found the sequential greedy insertion algorithm performed the best in terms of accuracy. While the sequential greedy insertion algorithm is briefly described in [52], a more detailed version of the algorithm's description is presented in [20]. Although in [52] the greedy sequential algorithm is tested on a DSM, the algorithm can be easily modified to work with irregular point spacings instead. Later

described in the algorithm design section, this facet will be important as the points to be triangulated exist as irregular point spacings.

In [52], Wang et. al. elaborate on the usefulness of TINs, explaining that the variable resolution and high capability of capturing significant terrain features makes TINs attractive modeling techniques for surface reconstruction and representation. Two fundamental rules for triangles constructed from a TIN exist as follows: triangle edges do not intersect one another; and triangles cannot overlap each other.

A plethora of triangulation methods for digital surface models exist. Geological information system (GIS) users typically prefer to interpolate the irregularly spaced raw LIDAR points, producing a digital surface model, and operate on the DEM with conventional image processing algorithms. With the point spacings existent as a 2-dimensional array of values, it is possible to operate on the array or matrix with image processing kernel functions. However, critics of these interpolated range images or DSMs argue they are an aberration which oversimplifies terrain modeling [30]. Obviously working with the raw LIDAR data and generating a TIN from it instead of working with the interpolated data will yield more accuracy. It is important to ensure that this increase in accuracy is worth the complexity associated with the irregular spacing of the raw data and the inability to use the conventional image processing algorithms existent for regular point spacings.

One topic, typically raised during TIN discussions, is the architecture of the data structure used to encode the TIN. In [27], Kidner et. al. argue that ultimately for each particular type of application there exists an optimal data structure. Therefore, no singular data structure can be optimal for all applications. It is therefore necessary to define and model a chosen data structure architecture based on a formulated problem definition.

Garland and Heckber's Greedy Insertion Triangulation algorithm is the procedure chosen to triangulate the LiDAR data for this proposal. This algorithm is described in detail in Appendix E. The algorithm has been modified to incorporate a noise filtering technique. This has been documented in Appendix F. Preliminary conclusions drawn from implementing that noise filtering technique are presented in [43], [44], and [45].

## **2.2 Interpolating Irregular Raw LIDAR to Fixed Intervals**

All of the above mentioned algorithms in the previous sections vary from one another based on the sources of input utilized and the methodology implemented to realize 3D reconstruction. The majority of the above discussed methods use an interpolation of the LIDAR data in some form or another for some part of the algorithm, in most cases for distinguishing building from non-building points. For the most part, algorithms will use some combination of LIDAR data, aerial imagery and/or GIS ground plan data for 3D reconstruction. Most of the algorithms apply some version of a thresholding technique from the DSM and DTM differences for distinguishing ground points from non ground points. For the 3D reconstruction process, methods either strategically grouped the coplanar data by extracting features or attempted to model the data by fitting pre-existing models. The methods grouping coplanar data did so by recognizing key structure characteristics such as break points and ridges. Some of the algorithms mentioned made use of various versions of the Hough transform to realize this.

Several data driven methods make use of applying a thresholding technique to the height differences between a Digital Terrain Model (DTM) and a Digital Surface Model (DSM) to distinguish building from non building regions. By applying morphological filtering techniques

such as [57] and [4] to a DSM, it is possible to create a DTM, a model representing only the terrain, without buildings, trees, cars, etc. The morphological filters typically exist as kernel functions, relatively small matrices, which operate on larger matrices. The original LIDAR data, existing at irregular point intervals, is interpolated to fixed point intervals. Therefore the row  $(X_1, X_2, X_3, \dots)$  and column  $(Y_1, Y_2, Y_3, \dots)$  position of a given LIDAR point in a DSM matrix corresponds to its longitude and latitude, respectively. The value of the cell in the matrix corresponds to the elevation  $(Z_{XY})$  at the interpolated location.

	$X_1$	$X_2$	$X_3$	$\dots$
$Y_1$	$Z_{11}$	$Z_{12}$	$Z_{13}$	$\dots$
$Y_2$	$Z_{21}$	$Z_{22}$	$Z_{23}$	$\dots$
$Y_3$	$Z_{31}$	$Z_{32}$	$Z_{33}$	$\dots$
$\vdots$	$\dots$	$\dots$	$\dots$	$\dots$

Figure 2: LIDAR at Interpolated Spaces

Interpolating the data to fixed point intervals does simplify the problem and enable the use of kernel filtering functions. While morphological filtering may not be used in all cases, still many algorithms ([16], [17], [19], [22], [23], [32],[36], [40], [41], [46], [47], [54]) typically continue to interpolate to fixed intervals to make use of conventional methods.

However, in [17], Clode et. al., report the limits of their building detection technique and how interpolating the LIDAR points only adds to the inaccuracy and limitations of their method and all methods in general using DSMs. The accuracy in which a given algorithm can delineate building from non building regions is dependent on the laser divergence and the flying height of the aircraft procuring the LIDAR data, or ultimately the laser footprint uncertainty. However, if the data is interpolated to fixed intervals, then the limitation of the accuracy is worsened. With interpolation to fixed intervals, now a given algorithm's uncertainty is not simply a function of the laser foot print uncertainty, but a function of the laser foot print uncertainty and the point spacing combined. In [50], Vosselman elaborates that when the irregular points are interpolated, in instances where heights are interpolated between ground points and points on vegetation or buildings, the height differences in the interpolated data will be reduced. These instances increase the difficulty of making correct classifications distinguishing ground points from non-ground points. Morgan and Habib also favor the use of the raw, irregularly distributed LiDAR data [36] and propose a morphological filtering technique for building extraction in [35].

## CHAPTER THREE: ALGORITHM DESIGN

In chapter one, a brief overview of applications of reconstruction, how LiDAR data is procured and the sources of noise which affect that procurement process was presented. In chapter two, a literature review for 3D reconstruction algorithms using various types of input sources and reconstruction methodologies was presented. Furthermore, a literature review of existing triangulation algorithms was also presented.

### 3.1 Algorithm Functional Block Diagram

In this chapter, several design strategies are developed to outline how the goals listed in the problem statement section will be realized. An overall block diagram of the proposed algorithm is presented:

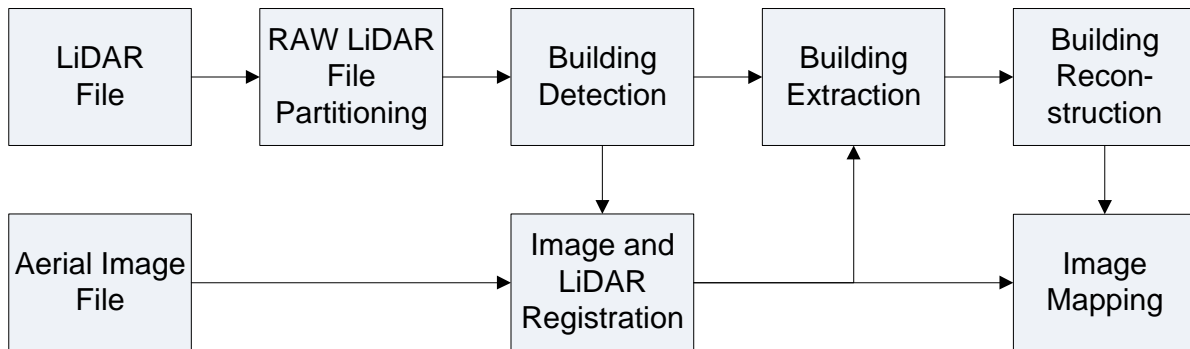


Figure 3 – System Block Diagram

The original LiDAR file, which can be as large as two square kilometers or several gigabytes in size, is partitioned into smaller segments. The proposed algorithm then proceeds to operate on these smaller (neighborhood block size) segments, reconstructing them one at a time.

### 3.2 Building Detection

After partitioning, terrain and non terrain points are distinguished via the following building detection method:

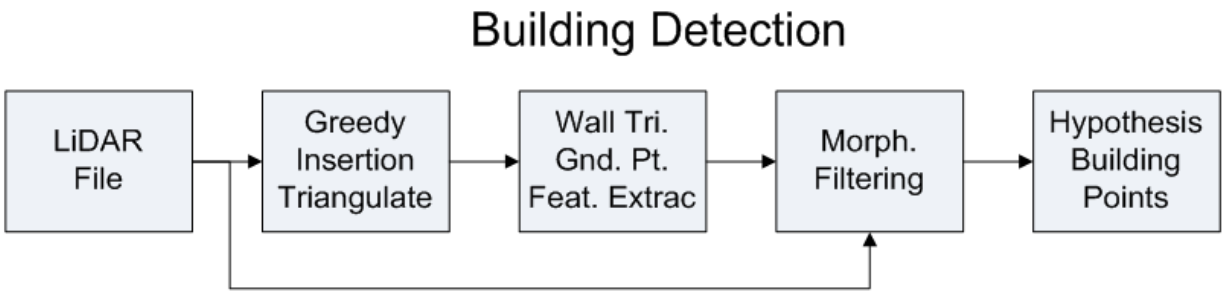


Figure 4 - Proposed Building Detection Method

The partitioned LiDAR file is triangulated. From this triangulation wall triangles and ‘hypothesis ground points’ are identified. Note the following figures:

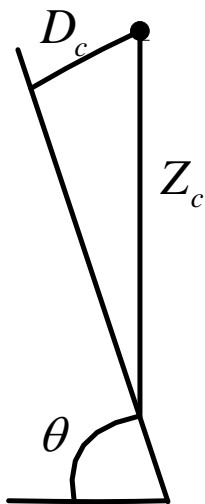


Figure 5 - Pitch of Roof Plane (Theta)

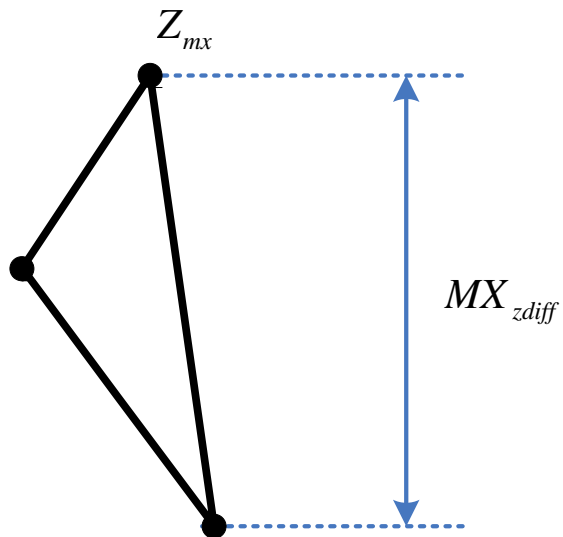


Figure 6 - Triangle Elevation Difference

The maximum difference in elevation between the two farthest points in a single triangle separated in terms of elevation is calculated. If that value (labeled MXzdiff in [Figure 6](#)) is greater than a threshold value (2 meters or about 6 ½ feet) and the pitch (labeled as theta in [Figure 5](#)) of the triangle is greater than 60 degrees from the normal axis, then that triangle is considered a wall triangle. The point lowest in elevation in that given triangle is then considered a ‘hypothesis ground point’. Because wall triangles exist at discontinuities in the LiDAR image where triangles of greater than 2 meters in length are formed, the ‘hypothesis ground point’ is not necessarily a true ground point (a point belonging to the terrain).

### **3.3 Building Extraction**

The building detection algorithm will separate terrain from non terrain points. The labeled building points then along with other key features from other LiDAR points are fed into the following building extraction algorithm:

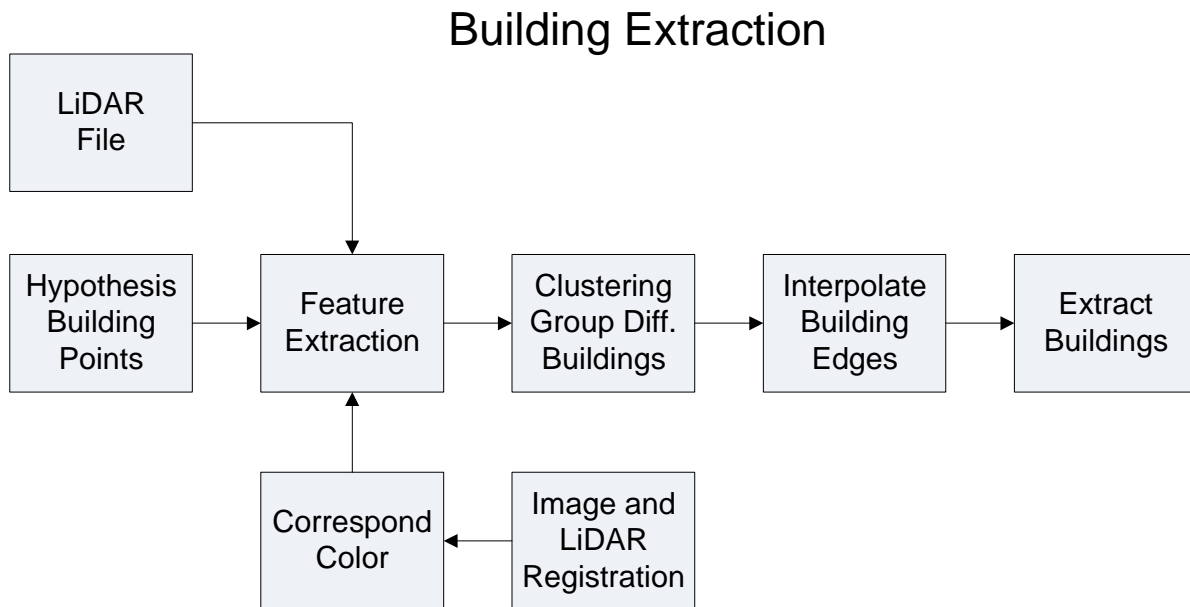


Figure 7 - Building Extraction Block Diagram

Recall that the ‘building detection’ stage really only separated terrain from non terrain points. The buildings still have to be automatically identified and extracted as individual, separate entities. Note that this is different from simply identifying building from non building LiDAR points. Each building must be recognized as its own entity which is different from near by buildings.

There are several features in which identify buildings as being different from non terrain points (specifically vegetation) and being different from other buildings. Typically vegetation LiDAR points will be highly discontinuous rather than close to smooth. The first and last return pulse elevation difference will be minimal even negligible for building structures. However for vegetation that first and last return pulse will be noticeable. Furthermore, in the triangulation of the LiDAR data, the normal vectors of triangles containing a single vegetation point should differ in orientation significantly – meaning the points closest to a single vegetation point should

differ significantly in elevation. The majority of vegetation points will typically correspond to a location in the aerial imagery where the pixel color is a shade of green (as opposed to grey, brown or other traditional building colors). Each building point within the same building should be spatially close to one another in terms of longitude and latitude. Furthermore, the points within a single building should be bounded by both the exterior wall triangles in the LiDAR TIN and the building edges detected from edge detection of the aerial imagery. Another way of stating this is, if all the points (both terrain and non terrain) are triangulated, all the triangles formed by the building points should be connected with one another. The bounded building area should not encompass terrain points.

The clustering algorithm for differentiating the buildings from their surroundings for the building extraction procedure will either be an algorithm already existing (Fuzzy ART, an optimized K-means, etc.) in the literature or something completely novel.

### **3.4 Building Reconstruction**

The proposed framework for building reconstruction is as follows:

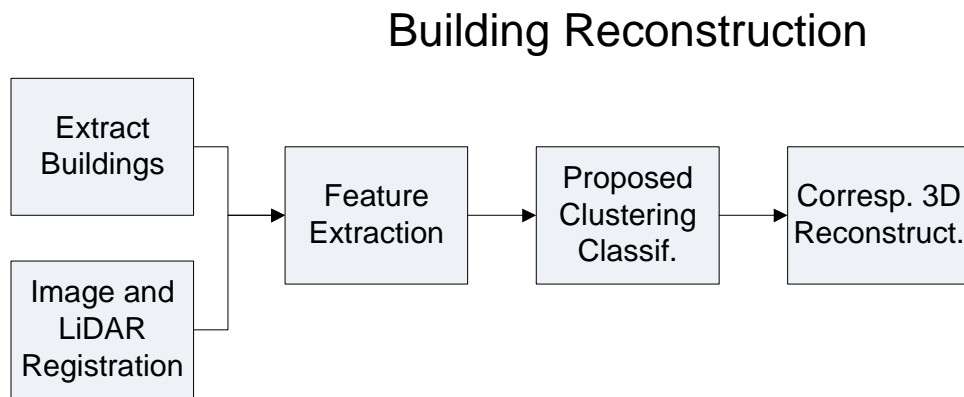


Figure 8 – Building Reconstruction Block Diagram

The extracted buildings along with the corresponding portion of the aerial imagery will be fed into a proposed clustering algorithm which will then classify the various roof structures. The clustering algorithm should be capable of handling simple (rectangular buildings with gable roofs) as well as complex (buildings with curved edges and dome shaped roofs) structures. Several features will be used to determine the structure and complexity of the extracted building. The normal vectors of the triangulated extracted building points and the relation of those normal vectors of triangles to adjacent triangle normal vectors will determine the rate of change in a given building surface. If that change or normal vector difference is minimal, ideally the triangles are then part of planar surface, whereas if the change is significant the triangles might be depicting a curved surface or roof discontinuity. In addition to normal vectors, the longitude, latitude and elevation of the LiDAR points as well as first and last return pulse difference can also identify key features. The building points should be spatially close together in terms of longitude and latitude and discontinuities can be identified by height differences. Building clutter such as air conditioning vents, chimneys, etc. might also have a different color than the overall roof color.

### **3.5 Image Mapping**

The image mapping will simply be a process where significant, high level features such as corresponding roof ridges and building corners are identified in both the reconstructed models and the aerial image. Then calculations will take place to uncover the transformation (rotation, scaling and perspective change) necessary to map the image onto the reconstructed model.

### **3.6 Algorithm Novelty**

In meeting the goals outlined in the previous section, this work will yield several novel contributions. First and foremost this proposed work will investigate reconstruction and building detection procedures in which will be generalized for irregularly spaced LiDAR data. The majority of algorithms existent in the literature currently focus on interpolating the data and arguably introduce unnecessary interpolation errors. Furthermore, operating on the interpolated data seems more of a programming convenience rather than a necessity which yields significant computational savings.

This proposed work will investigate developing a new building detection and extraction method that will involve minimal user intervention. At most, the only intervention that may be required is the a-priori knowledge of the largest building size existent in the data set, however research will be done in possibly even removing that a priori parameter. The building detection will make use of the raw LiDAR data as well as the triangulation of that data. Furthermore, in order to aid with automated reconstruction, buildings will automatically be identified and isolated for reconstruction.

For reconstruction, a new clustering algorithm will be proposed to specifically tailor to reconstruction from LiDAR and aerial imagery (as opposed to a general clustering algorithm tailored for all clustering applications). Finally, a method for automatically registering the LiDAR data to the aerial imagery will also be investigated. Several authors currently manually register these sources. The manual registration of imagery to a given 3D model requires a trained technician to identify key corresponding feature points within the two sources for successful registration. To speed up the processing of large scale data sets and eliminate the

need for the presence of trained technicians, automating this process, among several others, will be a significant priority in this proposed research.

## CHAPTER FOUR: MILESTONES

The duration of my dissertation studies will span out as follows. During the summer of 2007, I will investigate and possibly experiment implementing systematic noise removal techniques for the LiDAR data. I will continue debugging and optimizing my C implementation of the Greedy Insertion Triangulation algorithm. Albeit it currently, successfully triangulates 10,000 points in less than 10 seconds, it should be able to do 100,000s of points in less than a minute. Furthermore, for the summer of 2007, I will develop the proposed building detection technique and investigate methods of eliminating the need for an a-priori maximum building size parameter (associated with morphological filtering).

For the fall of 2007, I will develop a procedure that will automatically register the LiDAR data to an aerial image. After that, I will then create the proposed building extraction method. Following the completion of the building extraction method will be a publication documenting this work.

For the spring of 2008, I will develop the proposed reconstruction algorithm with a novel clustering technique. I will also develop a procedure for automatically mapping aerial images to the constructed models. I may have to develop an OpenGL program to render the neighborhood block size terrain subsets with the mapped aerial images in the event that Matlab fails to efficiently do this.

For the summer of 2008, I will publish the reconstruction algorithm and begin writing my dissertation. I will also defend my dissertation during this same semester (note the defense deadline for dissertations is the beginning of April).

**APPENDIX A:  
SEQUENTIAL GREEDY INSERTION STEPS**

### Step 1 - Initial Triangulation

Step 1a – Select the 4 outermost corner points of the LIDAR data (some points may be artificially created)

Step 1b – Perform Delaunay triangulation of selected 4 points (2 triangles formed) swapping the edges to obtain the optimal mesh

Step 1c – Mark the 4 points as used

Step 1d – For each of the two triangles formed, calculate the distance between the unused points and the plane formed by the triangle encompassing those unused points in x and y dimensions

Step 1d (i) – Cache the candidate point (point farthest away from triangle in z-direction) for each triangle formed

### Step 2 - Largest Deviation Point Insertion

Step 2a - Select the candidate point (the point with largest deviation from triangulated mesh).

Note: if this is the first iteration of the algorithm, all errors must be calculated as none are cached

Step 2b – Insert the Point into the Triangulated Mesh (mark it as used)

### Step 3 – Locate and Flip if Necessary

Step 3a – Locate the triangle within the triangulated mesh containing the recent inserted point

Step 3b – Split the located triangle into the necessary triangles containing the inserted point (based on the condition of insertion – [Figure 15](#)), [Figure 16](#), or [Figure 17](#))

Step 3c - Remove the original triangle (triangles are not allowed to overlap one another)

Step 3d – Recursively check each of the outer edges of the triangle containing the inserted point to see if flipping the edges will further optimize the existing triangulated mesh.

If a triangle edge is flipped, check the edges of both of those triangles and see if their adjacent triangle diagonals should be flipped (repeat until flipping will no longer further optimize the TIN according to local cost function).

Step 4– In the regions affected by insertion and flipping, recalculate the following parameters

Step 4a - The plane equations associated with the modified triangulations

Step 4b – Locate the triangles containing the unused points

Step 4c – Calculate the error between the unused point and the triangulated surface

Step 4d – For each triangle record the candidate value for the unused points (point with largest error deviation)

Step 5 - Return to step 2 and repeat if point budget or error approximation has not been met

If convergence in Step 5 was realized, finish inserting points and remove all triangles associated with artificial points (effectively removing those points from the triangulation)

**APPENDIX B:  
PLANAR COEFFICIENTS**

What follows is a derivation of calculations used to obtain planar coefficients and triangle normal vector. The planar equation defining the plane a given triangle forms is derived from the triangles' three vertices [Figure 9].

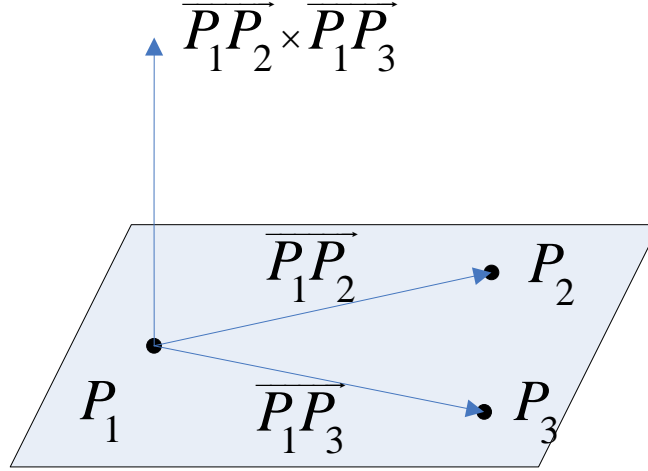


Figure 9: Normal Planar Vector Derived from Three Points

First, the two vectors  $\overline{P_1P_2}$  and  $\overline{P_1P_3}$ , are formed from points  $P_1$  and  $P_2$  and  $P_2$  and  $P_3$  respectively via the following equations:

$$\overline{P_1P_2} = (x_2 - x_1, y_2 - y_1, z_2 - z_1) = (x_{12}, y_{12}, z_{12}) \quad (1)$$

$$\overline{P_1P_3} = (x_3 - x_1, y_3 - y_1, z_3 - z_1) = (x_{13}, y_{13}, z_{13}) \quad (2)$$

The cross product of those equations defines the vector normal to the plane composed of the three vertices:

$$\overline{P_1P_2} \times \overline{P_1P_3} = (y_{12}z_{13} - y_{13}z_{12}, -x_{12}z_{13} + x_{13}z_{12}, x_{12}y_{13} - x_{13}y_{12}) \quad (3)$$

$$\overline{P_1P_2} \times \overline{P_1P_3} = (a, b, c) \quad (4)$$

Now that the normal vector is acquired, the coefficients ('a', 'b', 'c' and 'd') defining the plane in which the considered triangle's three vertices form, can be derived:

$$a \cdot (x - x_1) + b \cdot (y - y_1) + c \cdot (z - z_1) = 0 \quad (5)$$

$$a \cdot x + by + cz + (-ax_1 - by_1 - cz_1) = 0 \quad (6)$$

The 'd' coefficient is equal to the 4<sup>th</sup> term in equation (6). From equation (4) and equation (5), the values of 'a', 'b', 'c' are found to be the following:

$$a = y_{12}z_{13} - y_{13}z_{12} = (y_2 - y_1)(z_3 - z_1) - (y_3 - y_1)(z_2 - z_1) \quad (7)$$

$$b = -x_{12}z_{13} + x_{13}z_{12} = -(x_2 - x_1)(z_3 - z_1) + (x_3 - x_1)(z_2 - z_1) \quad (8)$$

$$c = x_{12}y_{13} - x_{13}y_{12} = (x_2 - x_1)(y_3 - y_1) - (x_3 - x_1)(y_2 - y_1) \quad (9)$$

$$d = -(ax_1 + by_1 + cz_1) \quad (10)$$

**APPENDIX C:  
DELAUNAY TRIANGULATION CIRCLE TEST**

After the insertion of a point, the edges of a triangle, with one of its vertex's being the candidate point, are checked to see if the adjacent triangle, sharing that edge, is compatible to have its diagonal flipped in accordance with Delaunay triangulation. This check is done via the circle test. The circle test aims to change the diagonal of a given two considered triangles to maximize the smaller interior angles in a given pair of triangles.

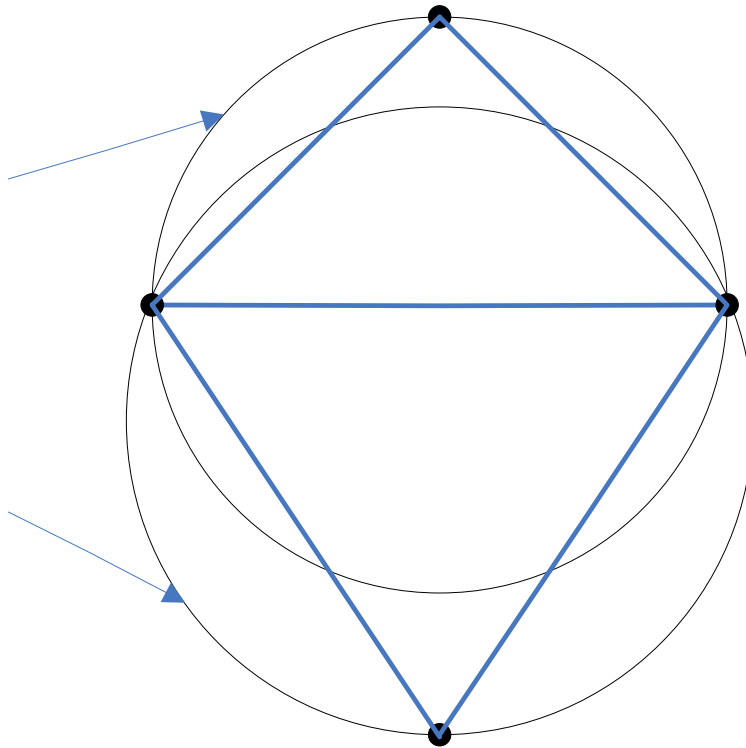


Figure 10 – Delaunay Circle Test

$CT_1$

For example, in [Figure 10] consider the two triangles, T1 (composed of vertices V1, V2, and V3) and T2 (composed of vertices V2, V3, and V4). Because vertex V4 lies outside of circle CT1 formed by the three points V1, V2, and V3 (conversely because vertex V1 lies outside of circle CT2 formed by the three points V2, V3, and V4), the configuration of the triangles exist such that their lesser internal angles are maximized.

In order to perform this test, the center of the circle, formed by the three vertices must be calculated. In [Figure 10], consider the lines formed by vertices V1, V2, and V1, V3, and then their perpendicular bisectors. Note that the three perpendicular bisectors of the sides of a triangle are concurrent at a point called the circumcenter (center of the circle). The slopes of the line segments formed by vertices V1, V2, and V1, V3 are defined as follows:

$CT_2$

$$V_1 = (x_1, y_1); V_2 = (x_2, y_2); V_3 = (x_3, y_3) \quad (11)$$

$$m_{12} = \frac{y_1 - y_2}{x_1 - x_2} \quad (12)$$

$$m_{13} = \frac{y_1 - y_3}{x_1 - x_3} \quad (13)$$

Therefore, the slopes of the perpendicular bisectors are defined as follows:

$$m_{12p} = \left( \frac{-1}{m_{12}} \right) \quad (14)$$

$$m_{13p} = \left( \frac{-1}{m_{13}} \right) \quad (15)$$

Using the point slope form of the equation of a line:

$$y - y_p = m \cdot (x - x_p) \quad (16)$$

The two perpendicular bisector line equations can be calculated as follows:

$$y - y_1 = m_{12p} \cdot (x - x_1) \quad (17)$$

$$y - y_3 = m_{13p} \cdot (x - x_3) \quad (18)$$

Distributing the slope and solving in terms of 'y', the above equations reduce to:

$$y = m_{12p} \cdot x - m_{12p} \cdot x_1 + y_1 \quad (19)$$

$$y = m_{13p} \cdot x - m_{13p} \cdot x_3 + y_3 \quad (20)$$

Setting the equations equation (19) and equation (20) equal to one another and solving for x (the intersection of the two perpendicular bisectors or the center of the circle in [\[Figure 10\]](#)):

$$x_C = \frac{m_{12} \cdot x_1 - m_{13} \cdot x_3 + y_3 - y_1}{m_{12} \cdot p - m_{13} \cdot p} \quad (21)$$

Plug (21) back into (19) to solve for the y-coordinate center of the circle:

$$y_C = m_{12} \cdot x_C - m_{12} \cdot x_1 + y_1 \quad (22)$$

If  $\|R_c\| - \|V\|_4 > 0$ , then the point is contained within the circle and flipping occurs. Else, if the aforementioned condition is not satisfied, then the point is outside the circle, the lesser of the two internal angles are already maximized for the considered pair of triangles and no flipping occurs.

**APPENDIX D:  
PERPENDICULAR DISTANCE FROM AN  
ARBITRARY POINT TO A GIVEN PLANE**

Calculating the distance between a given point and the plane formed by a given triangle is derived as follows.

The projection of one vector onto another is illustrated in [\[Figure 11\]](#) and described mathematically via the following equations:

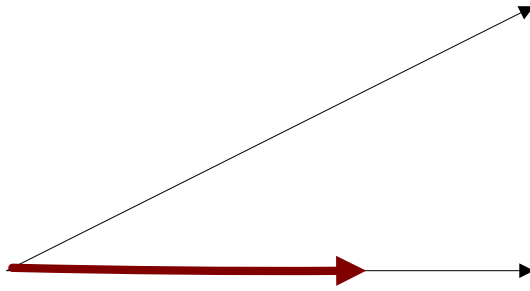


Figure 11 – Projection of a onto b

The orthogonal projection of the vector a onto b is defined as follows:

$$proj_a b = (a \cdot b)b \quad (23)$$

After normalizing the b vectors:

$$proj_a b = \left( a \cdot \frac{b}{\|b\|} \right) \left( \frac{b}{\|b\|} \right) \quad (24)$$

And then simplifying, the following equation results

$$proj_a b = \frac{a \cdot b}{\|b\|^2} b \quad (25)$$

Now consider point  $Q(x_1, y_1, z_1)$  located on plane F and point  $P_0(x_0, y_0, z_0)$  illustrated **b** accordingly in [\[Figure 12\]](#).

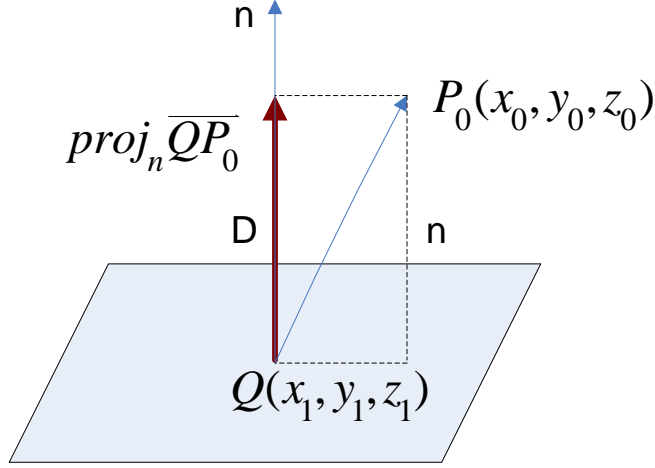


Figure 12 – Perpendicular Distance from Point to Plane

The distance between the two aforementioned points is defined as follows:

$$D = \left\| \text{proj}_n \overline{QP_0} \right\| = \frac{|\overline{QP_0} \cdot n|}{\|n\|} \quad (26)$$

The equation defining the vector from the plane to the point is defined as follows:

$$\overline{QP_0} = \langle x_0 - x_1, y_0 - y_1, z_0 - z_1 \rangle \quad (27)$$

The dot product of the above vector and the vector normal to the plane formed by the considered triangle's vertices is as follows:

$$\overline{QP_0} \cdot n = a \cdot (x_0 - x_1) + b \cdot (y_0 - y_1) + c \cdot (z_0 - z_1) \quad (28)$$

The magnitude of the normal vector is as follows:

$$\|n\| = \sqrt{a^2 + b^2 + c^2} \quad (29)$$

Note equation (27), equation (28), equation (29), which after plugging into equation (26), yields equation (30):

$$D = \frac{|a \cdot (x_0 - x_1) + b \cdot (y_0 - y_1) + c \cdot (z_0 - z_1)|}{\sqrt{a^2 + b^2 + c^2}} \quad (30)$$

Further simplifying the above equation, we can define the d coefficient as follows:

$$d = -ax - by - cz \quad (31)$$

Remember  $Q(x_1, y_1, z_1)$  lies within plane F and satisfies equation (26), therefore equation (31). Plugging equation (31) into equation (30) results in the following equation defining the distance D between unused points in the original LiDAR data and the triangulated surface approximation:

$$D = \frac{|a \cdot x_0 + b \cdot y_0 + c \cdot z_0 + d|}{\sqrt{a^2 + b^2 + c^2}} \quad (32)$$

The derivation of equation (32) is described in detail because this distance is equal to the perpendicular distance of a given candidate point, about to be inserted into the LIDAR based TIN, to the TIN itself. This distance is used in the Greedy Insertion Filtering technique.

**APPENDIX E:  
GREEDY INSERTION TRIANGULATION**

The algorithm selected to realize the triangulation of the irregular point spacings in the provided LIDAR data is Garland and Heckbert's sequential greedy insertion algorithm. In [20], Garland and Heckbert present both the sequential and parallel greedy insertion algorithms. The version of the greedy insertion algorithm, which only inserts a single point in each pass is called sequential greedy insertion, while the version of the algorithm in which inserts multiple points in each pass is called parallel greedy insertion. While the parallel version does cut down execution time, the savings realized come at the cost of the algorithm's performance in terms of accuracy; which is why the sequential version is selected.

The sequential greedy insertion algorithm simultaneously optimizes two adaptive optimization cost functions: (1) local Delaunay triangulation; (2) global point insertion. The algorithm starts by considering the quadrilateral formed by the outermost four points in terms of x and y or longitude and latitude spacing. Then an arbitrary triangulation is formed (two triangles are randomly formed from the 4 points).

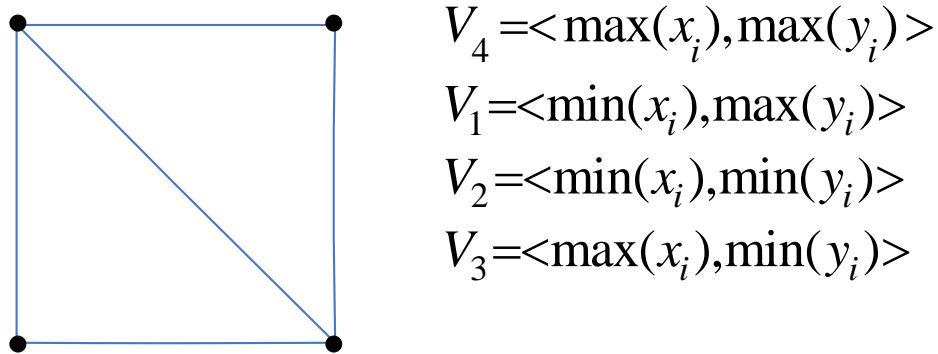


Figure 13: Initial Triangulation

That formation is then checked to see if flipping the diagonal will optimize the arbitrarily formed configuration to conform to Delaunay triangulation.

For all triangles, the distances between the triangles (planes) [Figure 14] and the points that they encompass (in x and y or longitude and latitude spacing) are calculated.

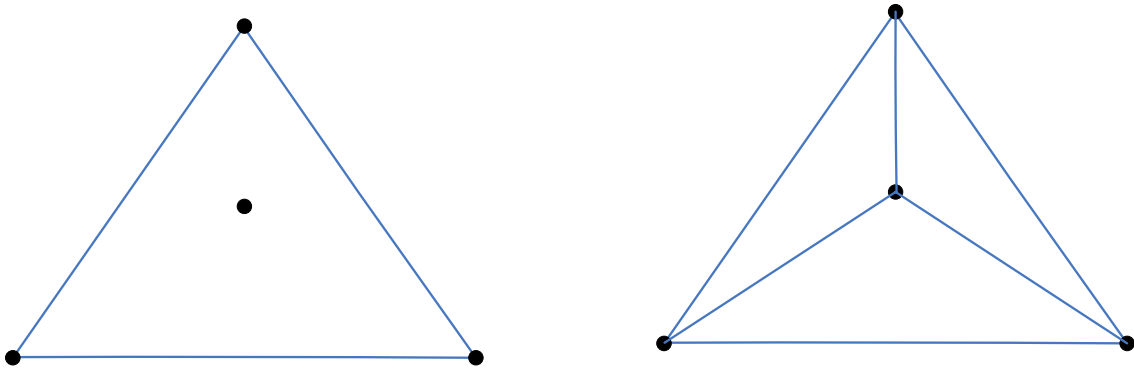


Figure 14: Distance between a given point and a plane

A detailed analysis showing the derivation of the calculations necessary to derive the planar coefficients describing the plane formed by the three vertices of a given triangle, which encompasses a given point not yet inserted, is presented in section B of the appendix. After all of the distances between the unused points and the existing triangulated surface are calculated, for each triangle, the unused point furthest from that triangle is cached into that triangles data structure.

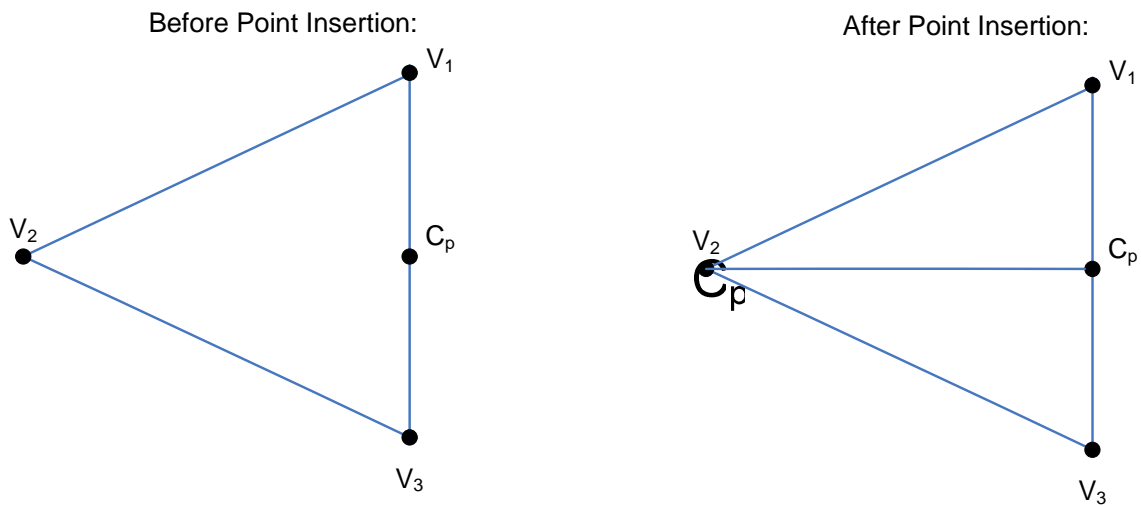
All of the LIDAR points that are considered, the point having the greatest distance from the TIN (labeled the candidate point) is the point inserted next (hence the name greedy insertion). Three cases can occur when inserting a given point: (1) the candidate point is inserted inside a triangle; (2) the candidate point is inserted at the edge of the outermost initial quadrilateral; and (3) the candidate point is inserted on a triangle edge.

The first point insertion case results in the formation of three triangles. The point is inserted and three lines are drawn from the point to the vertices of the encompassing triangle. This scenario is depicted in [Figure 15](#).



**Before Point Insertion:**  
Figure 15: Point Insertion (Case 1)

For the second point insertion case, the candidate point is inserted at the edge of the TIN, resulting in the formation of 2 new triangles, as depicted in [Figure 16](#).



V<sub>2</sub>

V<sub>3</sub>

Figure 16: Point Insertion (Case 2)

In the third point insertion case, the candidate point is inserted along the edge of a triangle. The algorithm is designed to delete the edge and then connect lines from the candidate point to the vertices of the two triangles which share the common edge in which the candidate point was inserted along. The third point insertion scenario is depicted in [Figure 17].

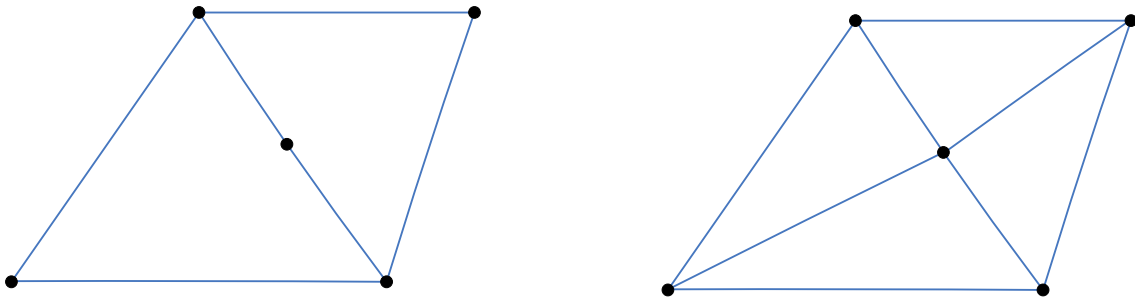


Figure 17: Point Insertion (Case 3)

After the insertion of the points, the edges of the triangles are checked for flipping. The edges are flipped to form a new diagonal if the flipping maximizes the lesser of the interior

**Before Point Insertion:**

angles of the triangles (Delaunay triangulation). If for two given triangles, their edges are flipped, then all of the adjacent triangles to those triangles are then checked to see if edge

flipping should be done with triangles adjacent to them. This process continues until it is determined that no adjacent triangle will further optimize the TIN via diagonal flipping in

accordance to Delaunay triangulation. This local optimization procedure is implemented to

combat the formation of slivers. A sliver is qualitatively defined as a triangle whose largest angle is 'relatively close' to 180 degrees. Therefore, triangle 'B' depicted in [Figure 18] is desired over triangle 'A'.

$V_4$

$V_1$

$C_p$

$V_2$

$V_3$

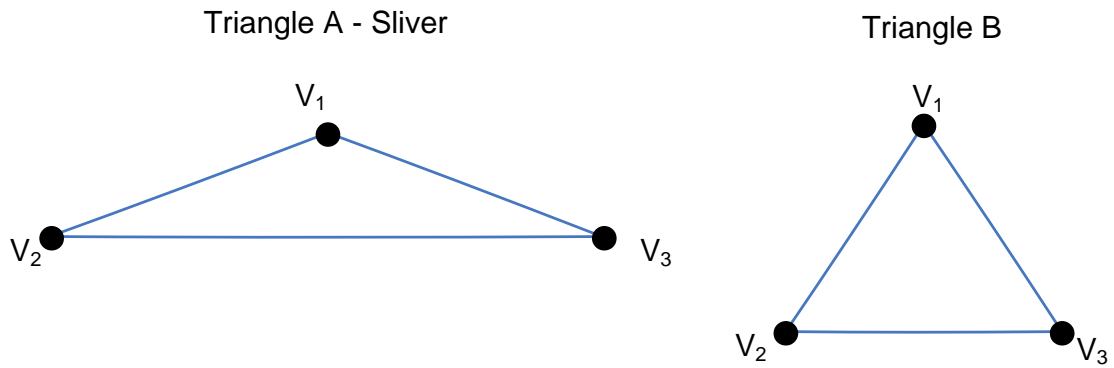


Figure 18: Sliver Example

All of the above procedures are depicted in the block diagram contained in [\[Figure 19\]](#).

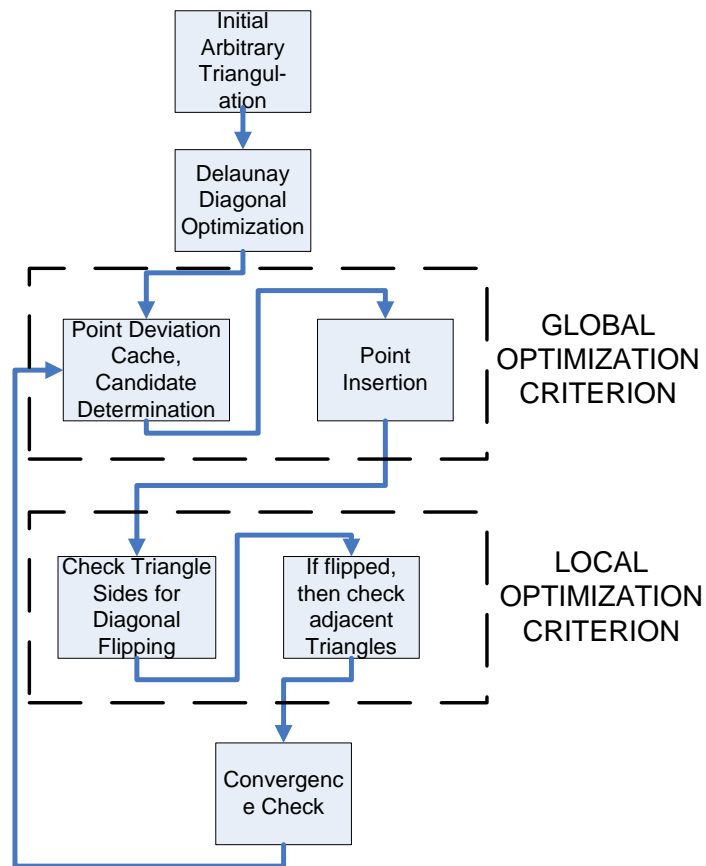


Figure 19: Greedy Insertion Block Diagram

**APPENDIX F:  
GREEDY INSERTION TRIANGULATION  
FILTERING MODIFICATION**

In order to group the coplanar triangles together, the normal vectors of the triangles generated from the greedy insertion algorithm are passed as inputs to the Fuzzy SART clustering algorithm. In order gain information from the magnitude of the normal vectors, the vector spanning from the origin to the triangle center is projected onto the normal vector. Therefore the orientation of the vector aligns with the orientation of the normal vector and the magnitude represents the distance from the origin to the plane encompassing the considered triangle.

The elevation coordinates of the LIDAR data are actually only accurate to a certain order of magnitude (in the order of centimeters). Making matters worse, the LIDAR data suffers from systematic errors and noise. Therefore, noise is existent in the data and presents difficulties for coplanar clustering based on the normal vectors of the triangles existent in the TIN generated from the raw LIDAR. An ideal set of coplanar triangles [Figure 20], actually exist as points jittering about that plane, as shown in [Figure 21]. The noise causes the LIDAR points to deviate from the ideal plane, thereby causing the normal vectors of the triangles to deviate from their ideal directions.

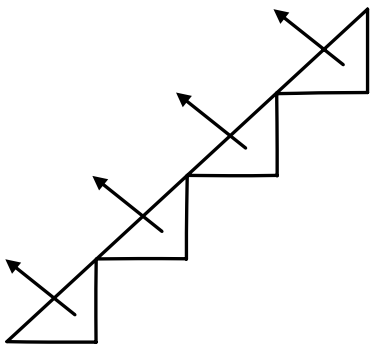


Figure 20: Ideal LIDAR Points

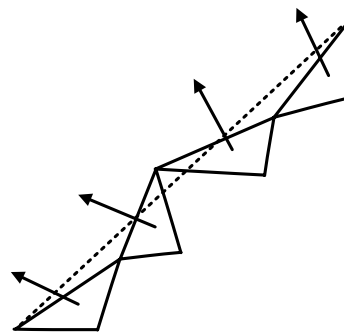


Figure 21: Actual LIDAR Points

One way to filter these errors would be to exploit the very nature of the triangulation algorithm selected. Sequential greedy insertion inserts the points farthest away from the initial

plane established. Therefore points along roof ridges, roof corners, and building edges are the points inserted first. The points inserted last are the points closest to an established plane, the points with the smallest errors. It is possible to simply program the sequential greedy insertion triangulation algorithm to only triangulate points above a certain error threshold. However, the insertion of fewer points leads to a less accurate TIN and furthermore, leads to fewer triangles sharing the same plane. Rather than not inserting the triangles, leading to fewer members of a given coplanar cluster, it would be advantageous to correct the inaccuracies of the points along the z or height dimension. Since the points, which jitter about the already established roof plane, are contained in a well defined plane, it is possible to remove the jitter or systematic error or noise by placing points below a certain threshold distance on the plane in which they are contained. While the longitude and latitude dimensions were preserved, the elevation dimension of a candidate point was modified if the candidate point met the following conditions: the perpendicular distance, defined in equation (33), of the candidate point was less than .2 meters ( $D_c \leq .2m$ ) from the containing triangulated plane; and the pitch of the roof, defined in equation (34) was less than 60 degrees ( $\theta \leq 60^\circ$ ).

$$D_c = \frac{|a \cdot (x_0 - x_1) + b \cdot (y_0 - y_1) + c \cdot (z_0 - z_1)|}{\sqrt{a^2 + b^2 + c^2}} \quad (33)$$

$$\theta = 90 - \sin^{-1} \left( \frac{D_c}{Z_c} \right) \quad (34)$$

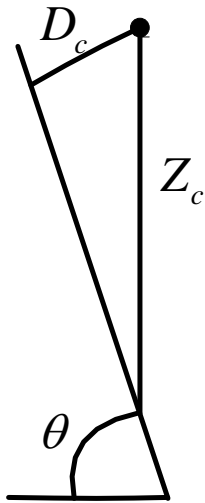


Figure 22 - Pitch of Roof Plane ( $\theta$ )

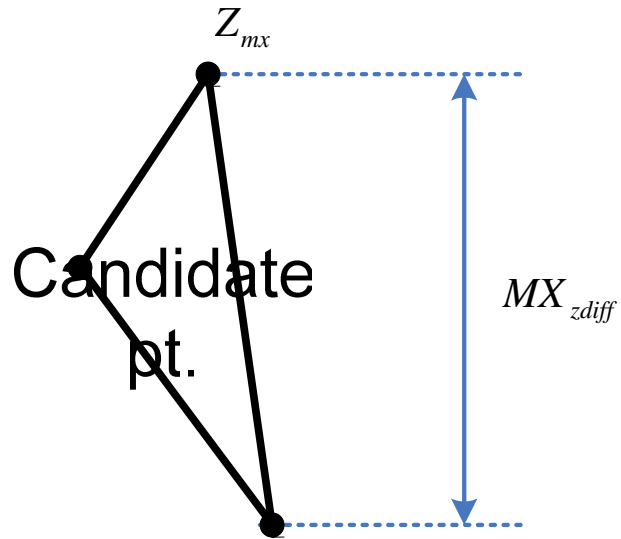


Figure 23 - Triangle Elevation Difference

## Roof Plane

Without the constraint imposed on the roof plane pitch ( $\theta \leq 60^\circ$ ), building edge points, which were not yet inserted/triangulated and less than .2 meters perpendicular distance from the building were being merged into the building's edge, thus distorting the building outline. The second constraint therefore confines points which only exist on a plane with a pitch ( $\theta \leq 60^\circ$ ) to become merged with that existent roof plane. Most of the building structures existent in the data set considered had roof planes with pitches less than 60 degrees.

This filtering technique was found to remove the noise depicted in [\[Figure 21\]](#) and therefore significantly improve normal vector triangulation clustering results. Merging these noisy points' elevation (z) dimension reinforced the presence of existing roof plane clusters resulting in an improved clustering performance by Fuzzy SART.

Only the spherical coordinates of the normal vectors of roof triangles were passed to the Fuzzy SART clustering algorithm. Triangles belonging to building walls and terrain were disregarded. In order to distinguish roof triangles from all other triangles the following measures

were implemented. For all triangles, the difference in elevation between the two vertices farthest from one another in a given triangle is calculated ( $MX_{zdiff}$  in [Figure 23]). All triangles having  $MX_{zdiff}$  greater than 2 meters were isolated. Then the average,  $Z_{avg}$ , of the z-dimension (elevation) of the highest vertex in a given triangle,  $Z_{mx}$  in [Figure 23], for all triangles with  $MX_{zdiff} \geq 2m$  was taken. All triangle centers must have an elevation greater than  $Z_{avg}$  in order to be considered as a candidate for a roof plane. The restriction of having the triangle differences being greater than 2 meters implements the assumption that all the buildings are greater than 2 meters or 6 and ½ feet.

## LIST OF REFERENCES

1. Anagnostopoulos, G. C.; “Novel Approaches in Adaptive Resonance Theory for Machine Learning.” *Doctoral Thesis, University of Central Florida, Orlando, Florida, 2001.*
2. Anagnostopoulos, G.C.; and Georgiopoulos, M.; “Ellipsoid ART and ARTMAP for Incremental Clustering Classification.” *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN '01)*, vol. 2, pp. 1221-1226, 2001
3. Anagnostopoulos, G.; and Georgiopoulos, M.; “Hypersphere ART and ARTMAP for unsupervised and supervised incremental learning,” *in Proc. IEEE-INNS-ENNS Int. Joint Conf. Neural Networks (IJCNN'00)*, vol. 6, Como, Italy, pp. 59–64., 2000
4. Arefi, H.; Hahn, M.; “A Morphological Reconstruction Algorithm For Separating Off-Terrain Points from Terrain Points in Laser Scanning Data.” *ISPRS Workshop Laser Scanning, 2005*
5. Baraldi, A.; and Parmiggiani, F.; “A neural network model for unsupervised categorization of multivalued input patterns: an application to satellite image clustering,” *IEEE Trans. Geosci. Remote Sensing*, vol. 33, no. 2, pp. 305-316, March 1995.
6. Baraldi, A.; Parmiggiani, F.; “A self-organizing neural network merging Kohonen's and ART models”, *Proceedings., IEEE International Conference on*, vol. 5, 27, pp. 2444 - 2449 vol.5, 1995
7. Baraldi, A.; Parmiggiani, F.; “Fuzzy combination of Kohonen's and ART neural network models to detect statistical regularities in a random sequence of multi-valued input patterns”, *Neural Networks, 1997., International Conference on*, vol. 1, 9-12, pp.281 – 286, June 1997
8. Barry, Joe; “Delaunay versus max-min solid angle triangulations for three dimensional mesh generation” *Internal Journal for Numerical Methods in Engineering*, vol. 31, pp. 987-997, 1991

9. Benediktsson, J. A.; Swain, P.H.; and Ersoy, O. K. ; “Neural network approaches versus statistical methods in classification of multisource remote sensing data,” *IEEE Trans. Geosci. Remote Sensing*, vol. 28, pp. 540 – 551, July 1990
10. Carpenter, G.A. & Grossberg, S. “A massively parallel architecture for a self-organizing neural pattern recognition machine.” *Computer Vision, Graphics, and Image Processing*, vol. 37, 54-115. 1987
11. Carpenter, G.A. & Grossberg, S. “Adaptive Resonance Theory.” *In M.A. Arbib (Ed.), The Handbook of Brain Theory and Neural Networks, Second Edition*, Cambridge, 2003
12. Carpenter, G.A. & Grossberg, S. “ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures.” *Neural Networks*, 3, 129-152, 1990
13. Carpenter, G.A.; Grossberg, S.; & Reynolds, J.H.; “ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network.” *Neural Networks*, vol. 4, pp. 565-588, 1991
14. Carpenter, G.A.; Grossberg, S.; Rosen, D.B.; “Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system”, *Neural Networks*, vol. 4, pp.759-771, 1991
15. Carpenter, G.A.; Grossberg, S.; & Rosen, D.B.; “ART 2-A: An adaptive resonance algorithm for rapid category learning and recognition.” *Neural Networks*, vol. 4, pp. 493-504, 1991
16. Chen, Liang Chien; Teo, Tee-Ann; Shao, Yi-Chen; Lai, Yen-Chung; Rau, Jiann-Yeo; “Fusion of LIDAR Data and Optical Imagery for Building Modeling.” *International Archives of Photogrammetry and Remote Sensing*, vol. 35, no. B4, pp. 732-737, 2004
17. Clode, S.P.; Kootsookos, P.J; and Rottensteiner, F. “Accurate Building Outlines from ALS Data.” *12<sup>th</sup> Australasian Remote Sensing and Photogrammetry Conference*, 2004
18. Filin, Sagi; “Elimination of Systematic Errors From Airborne Laser Scanning Data.” *Geoscience and Remote Sensing Symposium*, July 2005.

19. Fujii, Kensaku; and Arikawa, Tomohiko; "Urban Object Reconstruction Using Airborne Laser Elevation Image and Aerial Image." *IEEE Transactions on Geoscience and Remote Sensing*, vol. 40, no. 10, pp. 2234-2240, 2002
  
20. Garland, M.; Heckbert, P. S.; "Fast polygonal approximation of terrains and height fields." *Technical Report, Department of Computer Science, Carnegie Mellon University*, 1995.
  
21. Hofmann, A.D.; "Analysis of TIN-Structure Parameter Spaces In Airborne Laser Scanning Data for 3-D Building Model Generation." *Geo-Imagery Bridging Continents XXth ISPRS Congress*, 2004
  
22. Hu, Jinhui; You, Suya; Neumann, Ulrich; Park, Kyung Kook; "Building Modeling From LiDAR and Aerial Imagery." *Proceedings of ASPRS*, May 2004.
  
23. Huber, Martin; Schickler, Wolfgang; Hinz, Stefan; Baumgartner, Albert; "Fusion of LIDAR Data and Aerial Imagery for Automatic Reconstruction of Building Surfaces." *2nd Joint Workshop on Remote Sensing and Data Fusion over Urban Areas*, 2003
  
24. Huguet, B. Adriano; de Andrade, Marcos C.; Carceroni, Rodrigo L.; and de Araujo, Arnaldo. "Color-Based Watershed Segmentation of Low-Altitude Aerial Images." *Proceedings of the Computer Graphics and Image Processing, XVII Brazilian Symposium on*, pp. 138-145, 2004
  
25. Huguet, B. Adriano; Carceroni, Rodrigo L.; and de Araujo, Arnaldo. "Towards Automatic 3D Reconstruction of Urban Scenes From Low Altitude Aerial Images." *Proceedings of the 12th International Conference on Image Analysis and Processing*, pp. 254-259, 2003
  
26. Kangas, J.A.; Kohonen, T.; and Laaksonen, J. T.; "Variants of self-organizing maps," *IEE Trans. Neural Networks*, vol. 1 pp. 93-99, 1990.
  
27. Kidner, David B.; Ware, Mark J.; Sparkes, Andrew J.; Jones, Christopher B.; "Multiscale Terrain and Topographic Modeling with the Implicit TIN." *Transactions in GIS*, vol. 4, no. 4, pp 370-408, 2000
  
28. Kohonen, T.; "The self-organizing map", *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464-1480, Sept. 1990

29. Lammons, George; "After The Storm." *Military Geospatial Technology*, Volume 4, Issue 1, March 2006.
30. Lee, J; "Analyses of visibility sites on topographic surfaces." *International Journal of Geographic Information Systems*, vol. 5, pp. 413-29, 1991
31. Lattuada, Roberto; Raper, Jonathan; "Applications of 3D Delaunay triangulation algorithms in geoscientific modeling." *The Third International Conference/Workshop on Integrating GIS and Environmental Modeling CD-ROM*, January 1996
32. Liang-chien C. , Tee-ann T., Yi-chen S., Yen-chung L., Jiann-yeou R.; "Fusion of LIDAR Data and Optical Imagery for Building Modeling." *Geo-Imagery Bridging Continents XXth ISPRS Congress*, 12-23 July 2004
33. Mahalanobis, Abhijit; "Multidimensional Algorithms for Target Detection in LiDAR Imagery." University of Central Florida, Electrical and Computer Engineering Seminar Series. Orlando. 28 March 2007
34. Mass, H; and Vosselman, G. "Two Algorithms for Extracting Building Models from Raw Lser Altimetry Data." *ISPRS Journal of Photogrammetry & Remote Sensing*, vol. 54, pp. 153-155, 1999
35. Morgan, Michel; and Habib, Ayman; "3D TIN For Automatic Building Extraction from Airborne Laser Scanning Data." *Proceedings of the ASPRS "Gateway to the New Millennium"*, 2001
36. Morgan, Michel; Habib, Ayman; "Interpolation of LIDAR Data and Automatic Building Extraction." *ACSM-ASPRS2002 Annual Conference Proceedings*, 2002
37. Overby, Jens; Bodum, Lars; Kjems, Erik; and Ilsoe, Pee M; "Automatic 3D Building Reconstruction from Airborne Laser Scanning and Cadastral Data Using Hough Transform." *Geo-Imagery Bridging Continents 20<sup>th</sup> ISPRS Congress*, 2004
38. Rottensteiner, F.; and Briese, Ch.; "A New Method for Building Extraction in Urban Areas from High-Resolution LIDAR Data." *IAPRSIS*, vol. 34/3A, pp. 295-301, 2002

39. Rottensteiner, F.; and Briese, Ch.; "Automatic Generation of Building Models from LIDAR Data and the Integration of Aerial Images." *ISPRS*, vol. 34, 2003
40. Rottensteiner, Franz; Trinder, John; Clode, Simone; Kubik, Kurt; "Detecting Buildings and Roof Segments by Combining LIDAR Data and Multispectral Images." *Image and Vision Computing Proceedings*, 2003
41. Rottensteiner, Franz; Trinder, John; Clode, Simone; Kubik, Kurt; Lovell, Brian; "Building Detection by Dempster-Shafer Fusion of LIDAR Data and Multispectral Aerial Imagery." *Proceedings of the 17<sup>th</sup> International Conference on Pattern Recognition*, 2004
42. Schwalbe, Ellen, Mass, Hans-Gerd, Seidel, Frank; "3D Building Model Generation from Airborne Laser Scanner Data Using 2D GIS Data and Orthogonal Point Cloud Projections." *Proceedings of the ISPRS Workshop Laser Scanning*, 2005
43. Shorter, N. S. (2006). Heuristic 3D Reconstruction of Irregular Spaced LIDAR, Masters Thesis, University of Central Florida, Orlando Florida, 2006.
44. Shorter, N. S.; Kasparis, T. (2006). 3D Reconstruction of Irregular Spaced LIDAR, Proceedings of the 6th WSEAS International Conference on Systems Theory and Scientific Computation, Elounda, Greece, August 21-23 (pp. 19-24).
45. Shorter, N. S.; Kasparis, T. (2006). Fuzzy SART Clustering for 3D Reconstruction from Irregular LIDAR Data, WSEAS Transactions on Signal Processing, Vol. 2, No. 8 (pp. 1122 to 1129).
46. Sohn, G.; Dowman, I.; "Building Extraction Using LIDAR DEMS and IKONOS Images." *Proceedings of the ISPRS Working Group 3 workshop*, 2003
47. Straub, Bernd M.; Gerke, Markus; Koch, Andreas; "Automatic Extraction of Trees and Buildings from Image and Height Data in an Urban Environment." *International Workshop on Geo-Spatial Knowledge Processing for Natural Resource Management*, pp. 59-64, 2001
48. Sun, Bing Yu; Huang, De-Shang; Fang, Hai-Tao; "LiDAR Signal Denoising Using Least Square Support Vector Machine." *IEEE Signal Processing Letters*, Vol. 12, No. 2, February 2005.
49. Suveg, Ildiko; and Vosselman, George. "Automatic 3D Building Reconstruction." *Proceedings of SPIE*, vol. 4661, pp. 59 – 69, 2002

50. Vosselman, G.; "Slope Based Filtering of Laser Altimetry Data." *International Archives of Photogrammetry and Remote Sensing*, vol 33, part B3?2, pp. 935-942, 2000
  
51. Vu, Tuong Thuy; Matsoka, Matashi; Yamazaki, Fumio; "LiDAR based Change Detection of Buildings in Dense Urban Areas." *Geoscience and Remote Sensing Symposium*, 2004. IGARSS '04. Proceedings. 2004 IEEE International
  
52. Wang, Kai; Lo, Chor Pang; Brook, George; Arabnia, Hamid; "Comparison of existing triangulation methods for regularly and irregularly spaced height fields." *INT. J. Geographical Information Science*, vol. 15, no. 8, pp 743-762, 2001
  
53. Williamson, J; "Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps," *Neural Networks*, vol. 9, no. 5, pp. 881–897, 1996
  
54. Weed, Christopher A.; Crawford, Melba M.; Neuenschwander, Amy L.; Gutierrez, Roberto; "Classification of LIDAR Data Using a Lower Envelope Follower and Gradient-based Operator." *Geoscience and Remote Sensing Symposium*, vol. 3, pp. 1384-1386, 2002
  
55. Xu., R.; and Wunch, D.; II, "Survey of Clustering Algorithms", *IEEE Transactions on Neural Networks*, Vol. 16, No. 3, pp. 645-678., May 2005
  
56. Yu, Shirong; Wang, Weiran; "LiDAR Signal Denoising Based on Wavelet Domain Spatial Filtering." *International Conference on Radar*, October 2006
  
57. Zhang, Keqi; Cheng, Shu-Ching; Whitman; Dean, Shyu, Mei-Ling ;Yan, Jianhua; and Zhang, Chengcui. "A Progressive Morphological Filter For Removing Non-Ground Measurements from Airborne LIDAR Data." *IEEE Transactions on Geoscience and Remote Sensing*, vol 41, no. 4, pp. 872-882, 2003