ON FEATURE EXTRACTION FROM LARGE SCALE LINEAR LIDAR DATA

by

PARTHA PRATIM ACHARJEE

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2017

Acknowledgements

First of all, I would like to express my appreciation to Professor Venkat Devarajan for his mentorship throughout my whole Ph.D. career, which was not limited to academic guidance but as well as toward the personal development. He taught me to look at a problem from different directions, and also show me the beauty of the research process towards the solution.

I would also like to appreciate the help and advice I received from Dr. Alan Davis, Dr. Ioannis Schizas, Dr. Jonathan Bredow, Dr.Michael Manry, and Dr. Ramtin Madani. Their feedbacks and suggestions in the proposal defense and the final dissertation defense were very helpful.

I would like to thank Mr. Collin McCormick and Mr. Steven Nechero of the Natural Resources Conservation (NRCS) to provide the funding for this research over multiple years. They also provide good insight into the practical requirements, which increase the usefulness of this work in real world problem. NRCS also helped me by delivering the humongous amount of LiDAR dataset for our research. Ms. Charlotte Odom was very helpful to validate, train, and documenting the final tool developed from this dissertation. Our collaboration with ESRI was helpful to deliver the algorithm as software for industrial use. Help from Mr. Joseph McGlinchy during this collaboration is highly appreciated.

I would like to thank my lab mates especially Dr. Toscano, Dr. Alam, H. Yanyan, and Ahsan Habib for all of their thoughtful discussions and ideas.

Finally, I would like to appreciate the support I get from my parents, wife, and my brother. The long and stressful life at graduate school would not be possible with their support and encouragements.

July 31, 2017

Abstract

ON FEATURE EXTRACTION FROM LARGE SCALE LINEAR LIDAR DATA

Partha Acharjee, PhD

The University of Texas at Arlington, 2017

Supervising Professor: Venkat Devarajan

Airborne light detection and ranging (LiDAR) can generate co-registered elevation and intensity map over large terrain. The co-registered 3D map and intensity information can be used efficiently for different feature extraction application. In this dissertation, we developed two algorithms for feature extraction, and usages of features for practical applications. One of the developed algorithms can map still and flowing waterbody features, and another one can extract building feature and estimate solar potential on rooftops and facades.

Remote sensing capabilities, distinguishing characteristics of laser returns from water surface and specific data collection procedures provide LiDAR data an edge in this application domain. Furthermore, water surface mapping solutions must work on extremely large datasets, from a thousand square miles, to hundreds of thousands of square miles. National and state-wide map generation/upgradation and hydro-flattening of LiDAR data for many other applications are two leading needs of water surface mapping. These call for as much automation as possible. Researchers have developed many semi-automated algorithms using multiple semi-automated tools and human interventions. This reported work describes a consolidated algorithm and toolbox developed for large scale, automated water surface mapping. Geometric features such as flatness of water surface, higher elevation change in water-land interface and, optical properties such as

dropouts caused by specular reflection, bimodal intensity distributions were some of the linear LiDAR features exploited for water surface mapping. Large-scale data handling capabilities are incorporated by automated and intelligent windowing, by resolving boundary issues and integrating all results to a single output. This whole algorithm is developed as an ArcGIS toolbox using Python libraries. Testing and validation are performed on a large datasets to determine the effectiveness of the toolbox and results are presented.

Significant power demand is located in urban areas, where, theoretically, a large amount of building surface area is also available for solar panel installation. Therefore, property owners and power generation companies can benefit from a citywide solar potential map, which can provide available estimated annual solar energy at a given location. An efficient solar potential measurement is a prerequisite for an effective solar energy system in an urban area. In addition, the solar potential calculation from rooftops and building facades could open up a wide variety of options for solar panel installations. However, complex urban scenes make it hard to estimate the solar potential, partly because of shadows cast by the buildings. LiDAR-based 3D city models could possibly be the right technology for solar potential mapping. Although, most of the current LiDAR-based local solar potential assessment algorithms mainly address rooftop potential calculation, whereas building facades can contribute a significant amount of viable surface area for solar panel installation. In this paper, we introduce a new algorithm to calculate solar potential of both rooftop and building facades. Solar potential received by the rooftops and facades over the year are also investigated in the test area.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1 Airborne LiDAR history

Light Detection and Ranging (LiDAR) employs the same principle as radar but uses the near infra-red laser rather than a microwave signal. LiDAR uses a narrow laser beam to target different types of distant objects and the return laser beam is captured to measure the distance and physical features at a very high resolution [1]–[3]. First laser-based remote sensing technologies came into use in the 1960s. In 1962, one of the famous laser remote sensing experiments was carried out by the members of a team from Massachusetts Institute of Technology to measure the distance of the moon from the earth using laser pulses. They successfully detected the echoes of a pulsed optical radiation focused onto the surface of the moon[4]. Later, the accuracy of this experiment was improved by installing a retroreflector array on the surface of the moon on July 21, 1969, by the crew of Apollo 11, and two more retroreflector arrays were left by the crew members of Apollo 14 and Apollo 15. The distance of the moon was calculated by re-arranging the very basic equation of velocity definition,

$$Distance = Speed\ of\ light \times \frac{Total\ travel\ time}{2} \quad (1)$$

This concept opened up the idea of remote sensing using the laser beam, which was continued by NASA with airborne and spaceborne LiDAR development. Further scientific investigations on the laser at Stuttgart University demonstrated a laser profiler system with high geometric accuracy. However, in the mid-80s, a global positioning system/ inertial measurement unit (GPS/IMU) solution for sensor positioning was not reliable, which hindered the further development in this domain. Another growing technology at that time, the aerial photogrammetry, required an airborne GPS/IMU system, which stimulated a rapid development of these direct

georeferencing technologies. Airborne kinematic GPS solutions were invented as an extension of the ground-based GPS survey systems. In the meantime, the GPS satellite constellation bloomed enough to provide the coverage needed for widespread operations of the airborne GPS system. High-accuracy IMUs were developed for military missile guidance systems. With the arrival of high accuracy GPS/IMU system, by the mid-90s, commercially available LiDAR sensors were capable of 2,000 to 25,000 pulses per second, which were exclusively used for topographic mapping applications.

Before LiDAR was introduced to the mapping community, the science of photogrammetry (accurate mensuration from aerial photography) was used to generate the high-resolution map of different terrains, as well as for feature extraction. Low-resolution mapping was covered by a radar system, which showed different artifacts and limitations. Therefore, 3D mapping at high resolution and accuracy was still a domain to explore and invention. In that effort, LiDAR was introduced for 3D mapping, which provided a very accurate, direct and fast 3-D mapping solution [5]. Besides these technical edges, LiDAR is a potential mapping option in many geographical locations, where field surveying is not possible due to the low accessibility of the terrain. Additionally, co-registered distance and intensity information of the targeted object in LiDAR results in large savings in time and effort in registering elevation and intensity data obtained from disparate sources and sensors. This allows remote sensing researchers to more efficiently focus on the applications of such registered data. However, the huge co-registered 3-D point cloud introduced a new challenge in LiDAR data post-processing.

The state-of-the-art LiDAR systems are capable of 250,000 pulses per second, differentiate multiple returns from a single pulse and manage multiple pulses. 3D LiDAR point cloud brought new challenges such as irregularly spaced 3D dataset, large volume of dataset, data storage system, data visualization. In recent years, different algorithms and GIS software have been developed to handle the big data, which has made LiDAR an attractive 3-D mapping

solution. Although there are no public standards available yet for LiDAR data or derived products, different professional associations such as the International and American Societies for Photogrammetry and Remote Sensing (ISPRS and ASPRS) are working on the standardization of LiDAR data and derived products. ASPRS has developed a binary data format standard, called LiDAR Archive Standard (LAS), which has been accepted and widely used in the LiDAR community [6]. For visualization and processing, many proprietary and open source tools and libraries i.e. ArcGIS, ArcPy, MATLAB, PCL, LP360 are available now. All of these new developments help researchers to concentrate on developing new algorithms for different LiDAR applications.

In addition to the linear airborne LiDAR, new technologies such as Geiger mode and Single Photon LiDAR systems are emerging [7]. Hopefully, these emerging technologies will make 3D point cloud more available in near future.

1.2 LiDAR system components and data acquisition

Knowledge of LiDAR system components and data acquisition techniques is necessary to understand the nature of LiDAR data, which is also a pre-requisite to developed algorithm for feature extraction. A LiDAR system can be broadly categorized as an airborne or a terrestrial system. Terrestrial LiDAR, which is a ground-based mapping system, provides higher resolution and accuracy because of its short range of operation. In the terrestrial setup, scanners are mounted on ground-based devices, which mainly perform vertical scanning. On the other hand, an airborne LiDAR can cover a large area from an aircraft in a very short time. Airborne LiDAR sensors are mounted on an aircraft and downward scanning is performed. In airborne case, the scanner is a dynamic system, which makes the subsequent refinement of the raw return data to a useful co-registered 3D point-cloud very challenging. Therefore, there is also a georeferencing system to

track the sensor movement and orientation. A georeferencing system, a LiDAR data acquisition unit and a data processing unit work simultaneously to gather high-resolution 3D point cloud of the earth surface. The essential components of an airborne LiDAR system therefore are

a) An aircraft for sensor mounting

b) A laser scanner and receiver

c) A differential global positioning system (DGPS) for location tracking

d) An inertial measurement unit (IMU) to calculate aircraft position and orientation

e) Data storage and a processing unit

Current LiDAR systems may have supplementary components for co-collected ortho-imagery and hyperspectral images. As an example of a commercially available system, a *LiteMapper* scanner from *3D Laser Mapping* is shown in Figure 1 [8]. Current LiDAR systems have become high-speed scanning, compact, lightweight and power-efficient [9].

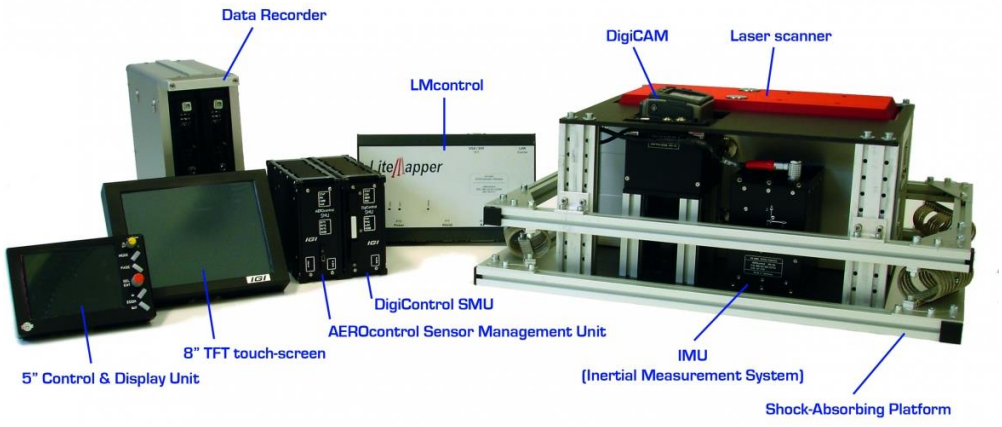

Figure 1: Components of a *LiteMapper* airborne LiDAR system.

The laser scanner and receivers are mounted on an aircraft in an airborne LiDAR system. A GPS system provides the location of the aircraft. Additionally, the IMU system monitors the

orientation of the aircraft, which helps to calculate the laser scanner orientation. The laser scanner continuously scans over $\pm\theta$ degrees about the nadir point, while the aircraft is moving forward. This angle is called scan angle, which is typically ± 20 degree. The scanning scheme of an airborne LiDAR system is shown in Figure 2.

The location of the aircraft is known from the GPS, the orientation of the aircraft is known from the IMU, the orientation of the laser scanner is known from the scan angle, and the range of each target point is known from the travel time of the return pulse. Using all of these variables, the latitude, longitude, and altitude of the targeted point can be calculated. Thus, 3D locations of all of the targeted points provide a 3D map of the landscape.

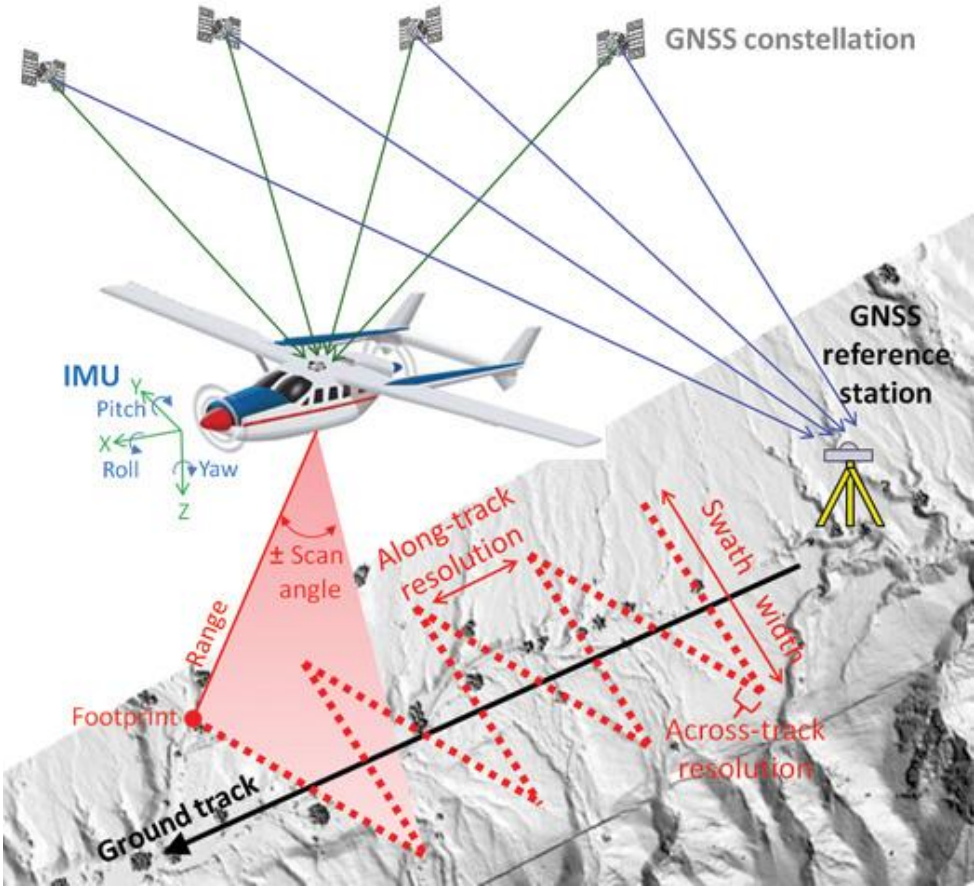

Figure 2: Airborne LiDAR scanning (Image-Notes magazine)

A LiDAR equation can be derived by integrating all of these parameters to estimate the location of the target point. With proper calibration, the exact coordinates of the target points can be estimated by,

$$\vec{X}_G = \vec{X}_0 + R_{yaw,pitch,roll}\vec{P}_G + R_{yaw,pitch,roll}R_{\Delta\omega,\Delta\phi,\Delta\kappa}R_{\alpha,\beta}\begin{bmatrix}0\\0\\-\rho\end{bmatrix} \quad (2)$$

where (Figure 3),

$\vec{X}_G$ is the position of laser point on the ground

$\vec{X}_0$ is the vector from the origin of the reference to aircraft

$R_{yaw,pitch,roll}$ is the rotation matrix relating the ground and the IMU coordinate system

$\vec{P}_G$ is the lever-arm which is the spatial offset between the laser unit and the IMU

$R_{\Delta\omega,\Delta\phi,\Delta\kappa}$ is the rotation matrix defined by the boresight angle $\Delta\omega, \Delta\phi, \Delta\kappa$ which are rotation offsets between the IMU and the laser unit coordinate

$R_{\alpha,\beta}$ is rotation matrix relating the laser unit to the laser beam coordinate where α is the encoder angle along the flight path and β is the encoder angle perpendicular to the flight path.

$\vec{\rho}$ is the laser range whose magnitude is equivalent to the distance from the laser firing point to its footprint.

The coordinates of the object points are derived with respect to the ground coordinate system. The GPS system provides the $\vec{X}_0$ vector to calculate the location of the aircraft. The IMU provides the orientation of the aircraft through $R_{yaw,pitch,roll}$ matrix, which relates the ground coordinate system and the IMU coordinate system. $\vec{P}_G$ is the lever-arm offset to calculate the location of the laser scanning system from the IMU system. The $R_{\Delta\omega,\Delta\phi,\Delta\kappa}$ matrix relates the laser

scanning coordinate system with the IMU coordinate system. The $R_{\alpha,\beta}$ matrix takes care of scan angle of the laser scanner, and $\vec{\rho}$ is calculated from the travel time of the return pulse. Through all of these transformations, the location of the object point is calculated as the $\vec{X}_G$ vector.

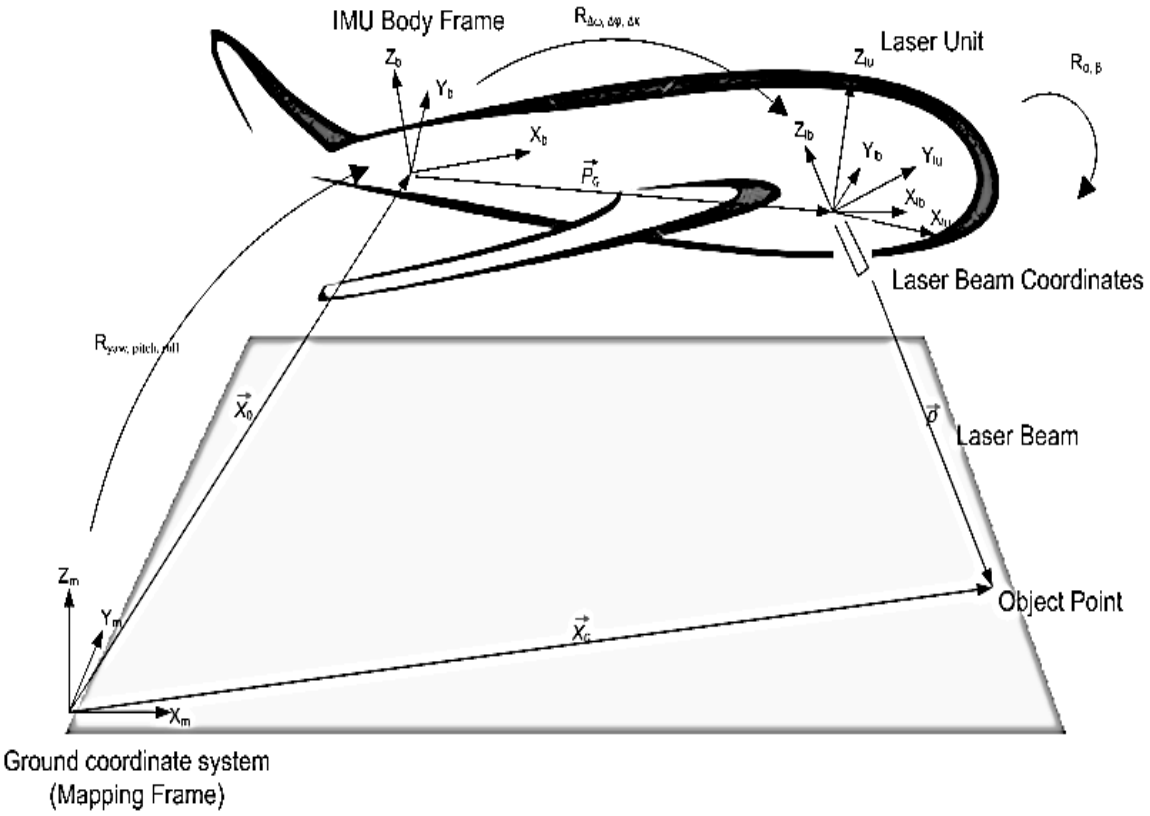

Figure 3: All parameters of LiDAR equation [10]

The derived coordinates of the targeted points (X, Y, Z), the intensity of the returned laser pulse (I), the number of returns from a pulse, scanning angle and related metadata can be saved as a LAS file for further processing. The aircraft generally fly in an aerial pattern to collect data from a large area. This scanning pattern may vary according to different applications. The data collected in a single pattern is called a swath. Adjacent swaths are overlapped for continuous terrain mapping. This overlapping data collection helps in calibration, strip adjustment and bundle

adjustment of multiple swaths. Generally, multiple swaths are saved in a single LAS file, and large projects may have thousands of LAS files. Swath, LAS file boundary, and point cloud will be shown in the following section.

1.3 LiDAR data characteristics and visualization

Before show a visualization of a 3D point cloud, a few characteristics of LiDAR data should be discussed. The main advantages of airborne LiDAR over other remote sensing mapping technologies are its co-registered elevation and intensity data, higher accuracy, denser resolution and its ability to map the earth below the vegetation.

LiDAR intensity is collected for every point, which represents the strength of the return pulse. It is related with the reflectivity of the surface struck by the laser beam. Reflectivity is the function of the laser wavelength, which is the near infrared zone and generally fixed for a project. In general, other factors such as weather and flight elevation are unchanged during a data collection phase. Therefore, the intensity is a relative measure, not a quantifiable measurement. For any specific object, it will be in a certain range for a project but the range may differ in another project. By considering its relativeness, intensity can still be used in feature detection and extraction. Typically, LiDAR data has a higher vertical accuracy (5-15cm RMSE) than the horizontal accuracy (40cm RMSE). LiDAR 3D point cloud density is varied based on the project requirement. Small feature extraction application, power line detection, may need 20 points per square meter.

LiDAR also has the capability to map under vegetation because of the multiple return property of the laser [11]. The multi-return phenomenon is described in Figure 4. A portion of the laser beam can be passed through the leaves, when the remaining part may be reflected as a return. This reflection and refraction can happen at multiple levels. Therefore, a single laser pulse can be returned from multiple levels of vegetation. Finally, the last return may come from the ground

surface. The total number of returns and return number of each received signal are stored in the LAS file. For example, the returns are stored as the 1st return of 4 returns, 2nd return of 4 returns, and so on. Each 3D point is listed with its respective information in the LAS files. From the LiDAR 3D point cloud, a very accurate digital elevation model (DEM), a digital terrain model (DTM), a triangulated irregular network (TIN) etc. can be derived [12]–[15].

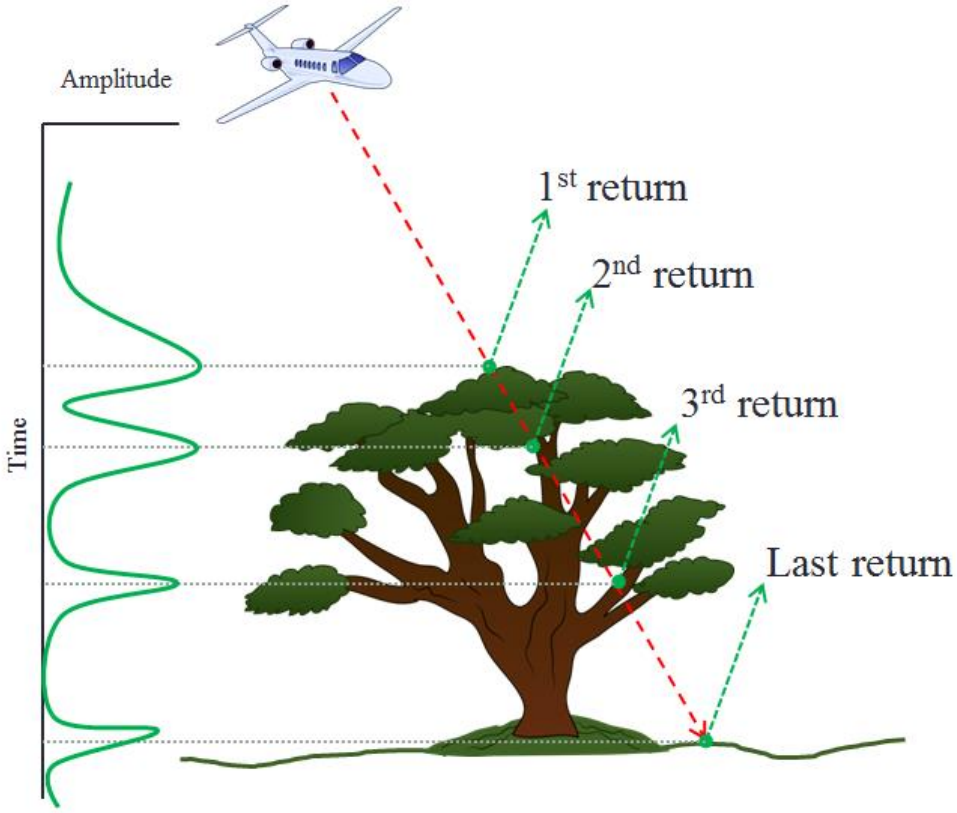

Figure 4: Multiple returns from vegetation

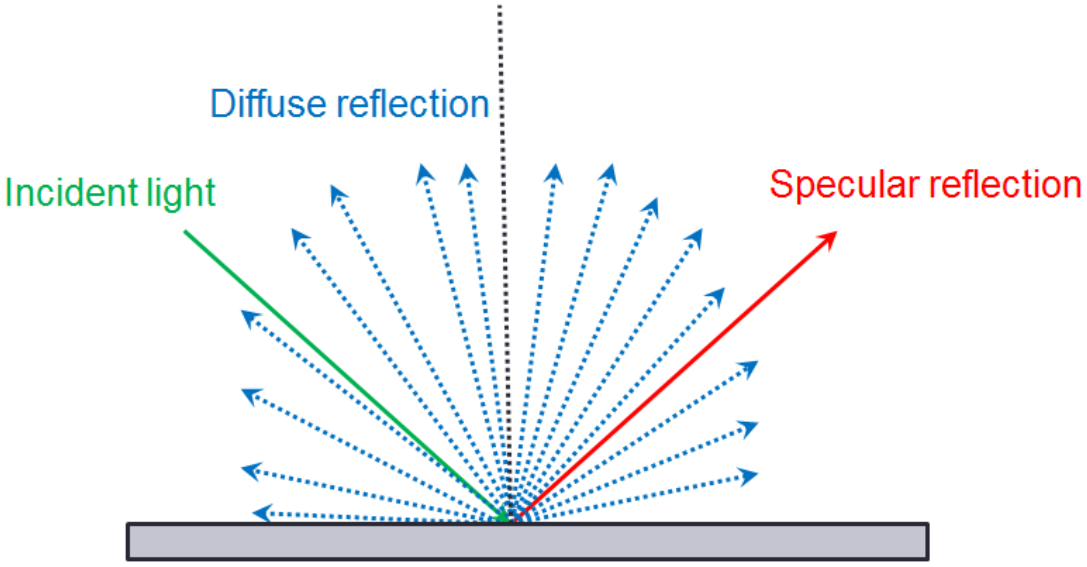

Figure 5: Different types of reflection of a laser pulse.

Typically, the LiDAR receiver does not collect the specular reflections of the incident laser pulses. In fact, it mostly receives diffuse reflection. Therefore, the receiver does not have to be in the position where it can receive the specular reflection. Receiving the specular reflection changes the intensity value significantly, especially from water surfaces. The LiDAR intensity property from water surfaces will be discussed in details in a later section.

Now, few LiDAR sample data are presented in Figure 6 for better understanding some of the LiDAR properties mentioned thus far.

The storing, visualization and processing of a LiDAR 3D point cloud demand significant computation power, large storage, and specialized software tools. In recent years, different tools have been developed for LiDAR data processing and visualization. In Figure 6, the top view of a 3D point cloud is shown using LP360 (a product from GeoCue) in ArcGIS environment. The point cloud is overlaid on satellite images, and LAS files boundaries are shown by red boxes. From the figure, it can be seen that the data points are irregularly spaced and sparse. The irregular spacing

happens because not all objects provide returns. The point cloud is color-coded by the elevation value, higher elevation is shown in red color and lower elevation is shown in blue color. Each LAS file consists of three swaths, which are overlapped in two regions (two strips in each las files in Figure 6). Point density is higher at the overlapping region because both swaths contribute points in that zone. It should be also noted that in a few areas data density is very low because of unavailability of LiDAR region from that region.

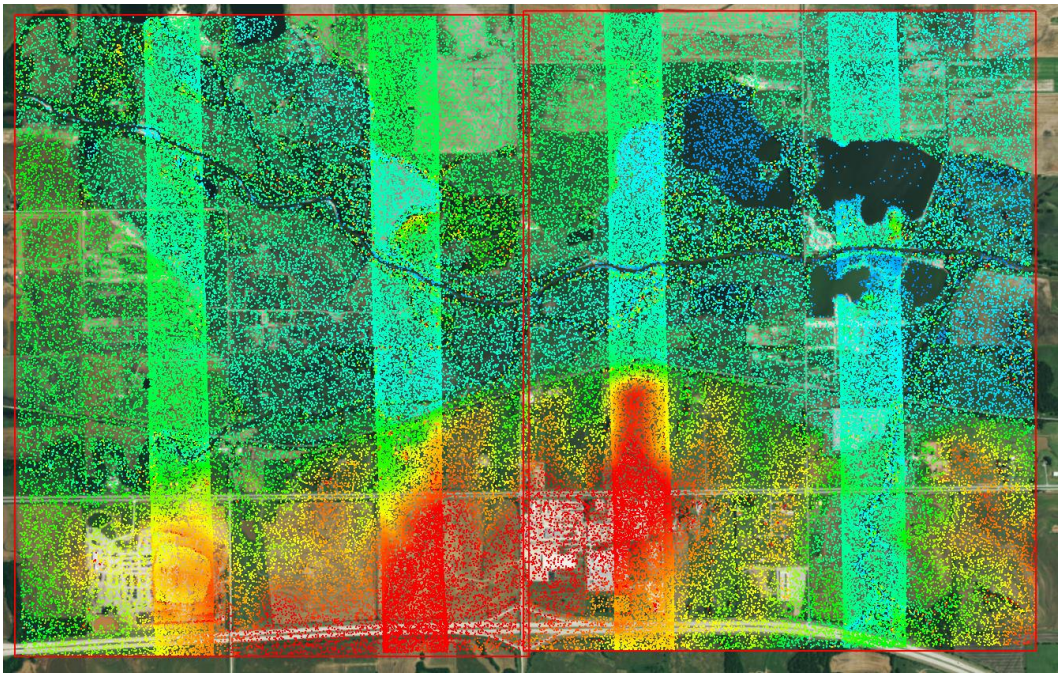

Figure 6: 3D LiDAR point cloud from the top

The same 3D point cloud is shown from different viewpoints for better perception in Figure 7. It is clear that the 3D point cloud provides a 3D model of the scene, although the points are sparse and irregularly spaced. For a better visualization, yet another form of powerful visualization method, known as a triangulated irregular network (TIN), is commonly used. A TIN is obtained using the Delaunay Triangulation algorithm, in which a network of points is connected by edges in three dimensions, to form a triangle. Rendering of the triangular facets delivers the three-dimensional visualization. This TIN based visualization gives a quick 3D model of the

whole landscape. Facets of the TIN are color-coded based on the average elevation, intensity or other LiDAR parameters.

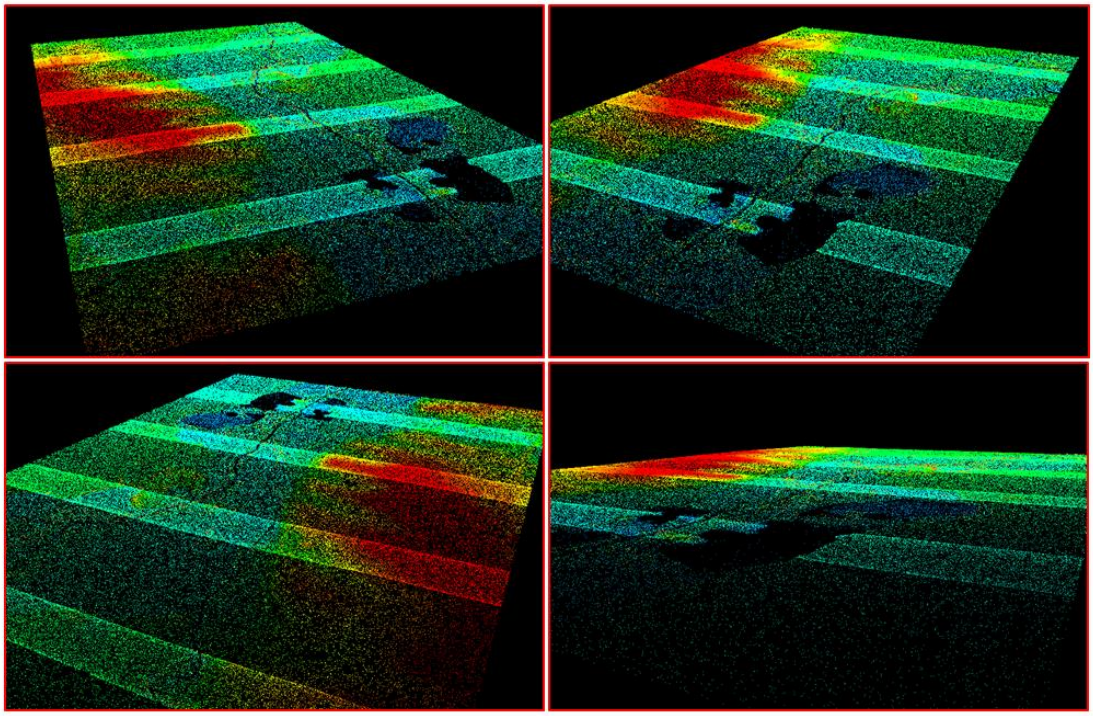

Figure 7: LiDAR point cloud from four different angles of view.

A top view of the TIN visualization using the same dataset is shown in Figure 8, where the elevation information is used to determine the triangles facets color. From the figure, it is clear that the TIN visualization renders a better 3D model than a sparse point cloud. A few places where no LiDAR return was available are also filled with TIN from neighboring points. Four different TIN representations of the same area from multiple viewpoints are shown in Figure 9. The higher elevation of the land surface and the vegetation is visible in Figure 9.

Similarly, other LiDAR parameters can be also shown as a 3D point cloud as well as a TIN surface. In Figure 10, the intensity is shown using a TIN representation for the same area.

These visualizations and corresponding overlaid satellite image are powerful resources for manual feature extraction and, validation of any automated detection.

A few more 3D point cloud visualizations are shown in the following figures to get a more comprehensive picture of possible LiDAR data. In Figure 11, a 3D point cloud of New York City is shown. All points are color coded based on their elevation. It is clear that building, vegetation, and grounds are visually distinguishable based on color. In Figure 12, a cross-sectional view of the vegetation is shown to elaborate on the multi-return phenomenon in LiDAR data. It is seen that multiple LiDAR returns are available from different levels of the vegetation. Finally, in Figure 13, it can be seen that the number of LiDAR returns per unit area is lower for water surfaces.

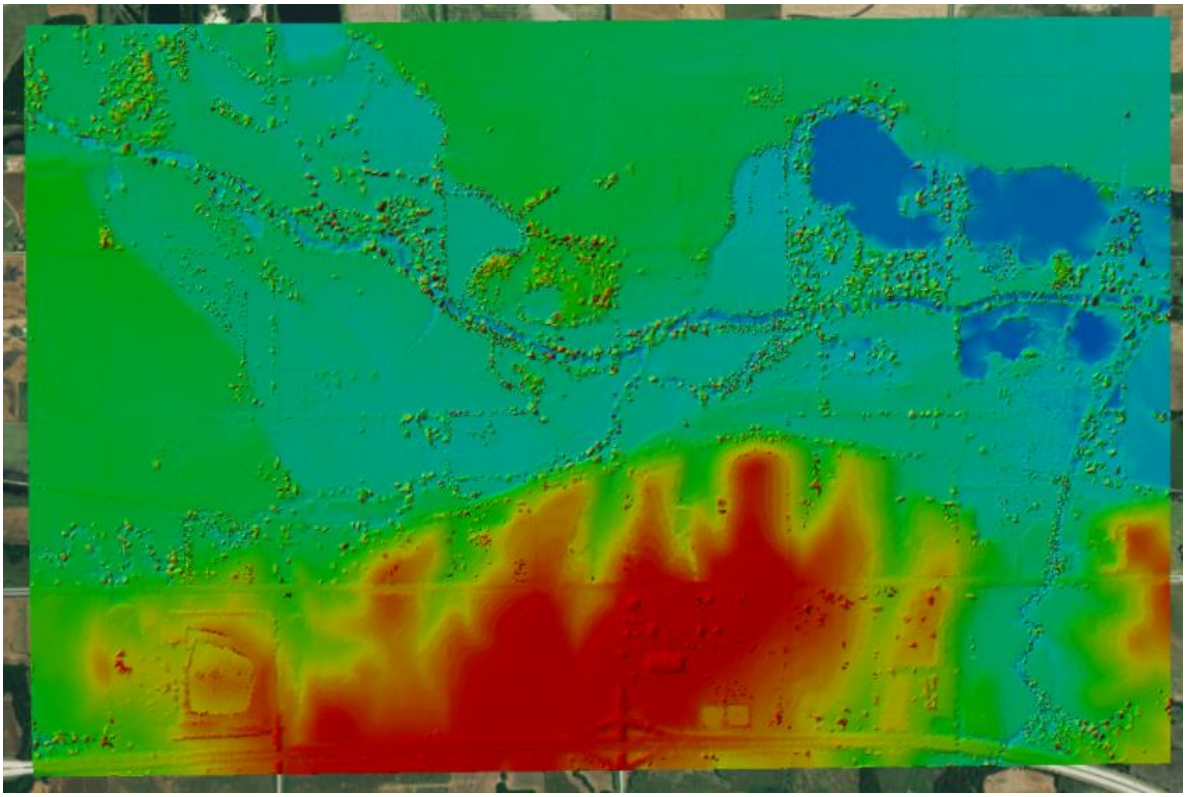

Figure 8: TIN visualization of LiDAR data, color-coded by elevation

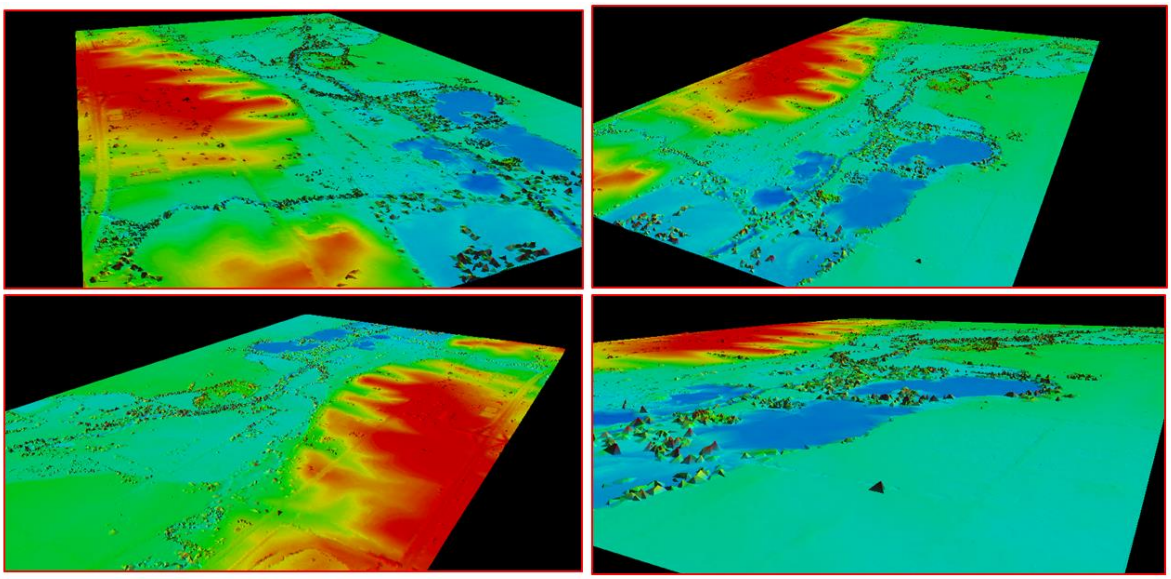

Figure 9: TIN representation of the 3D point cloud from four different angles

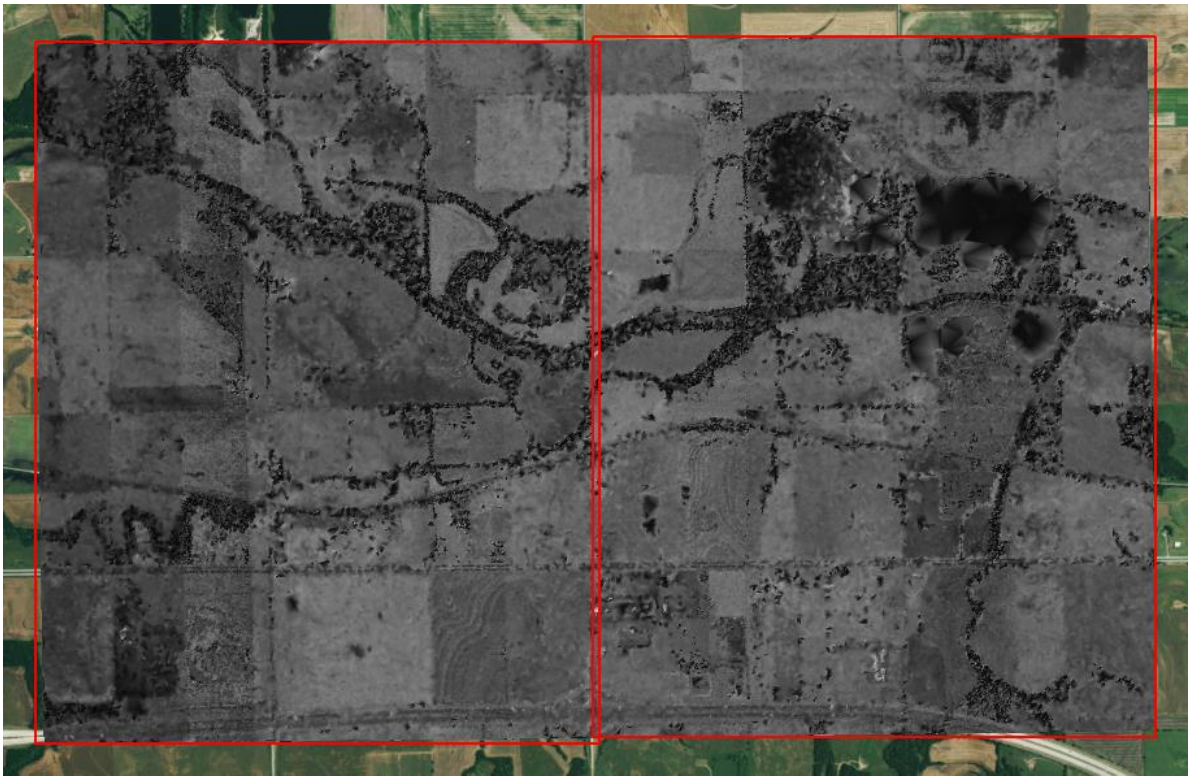

Figure 10: Intensity of LiDAR data in TIN representation

Besides visualization, few statistics (or metadata) of a typical LiDAR project will provide more insight into the LiDAR data. In Figure 14, statistics of a few LAS files are shown. In this example, each LAS file contains around 10 million of points and average point spacing is around 0.5 units. Size, shape, and number of LAS files are different for different projects. In Figure 15, detail statistics of a typical LAS file is shown. The LAS file contains the coordinate system information, unit used for X/Y/Z ranges etc. More than three-quarter of returns are typically first returns. The raw LiDAR data can be processed to extract all sorts of information. One very popular process, primarily manual or semi-automated, is the classification of individual returns as ground, unassigned, noise, water etc. It is important to be noted that this classification information is a derived property using manual or automated algorithms.

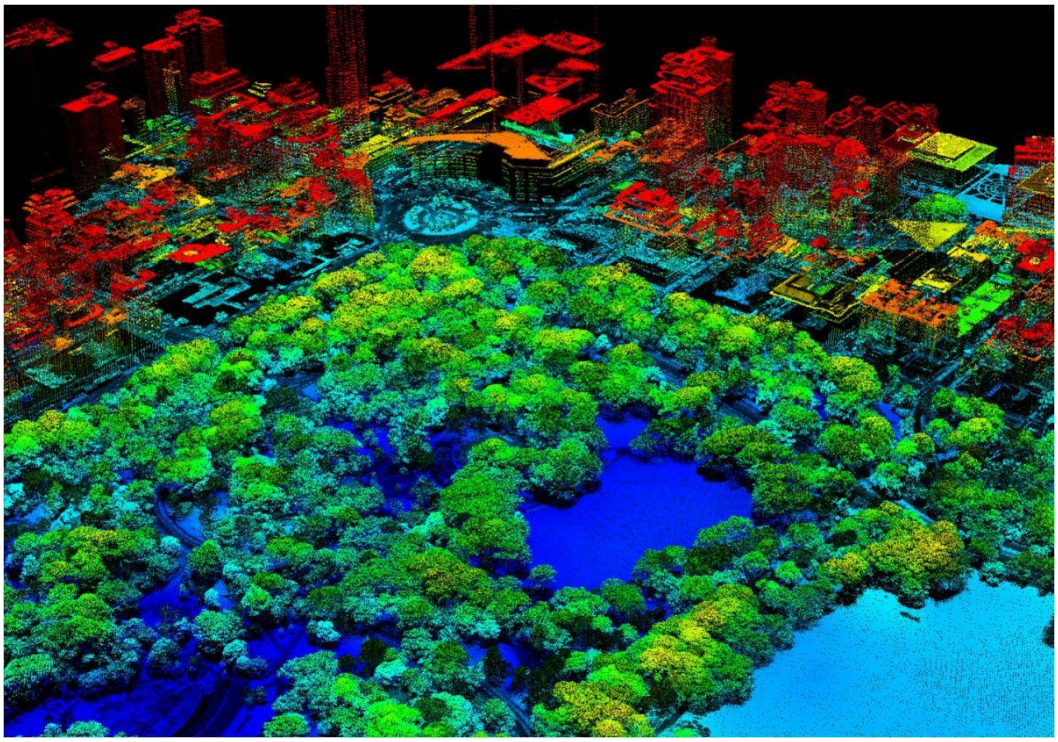

Figure 11: A 3D point cloud of NYC (Credit: Jarlath O'Neil-Dunne; UVM-SAL)

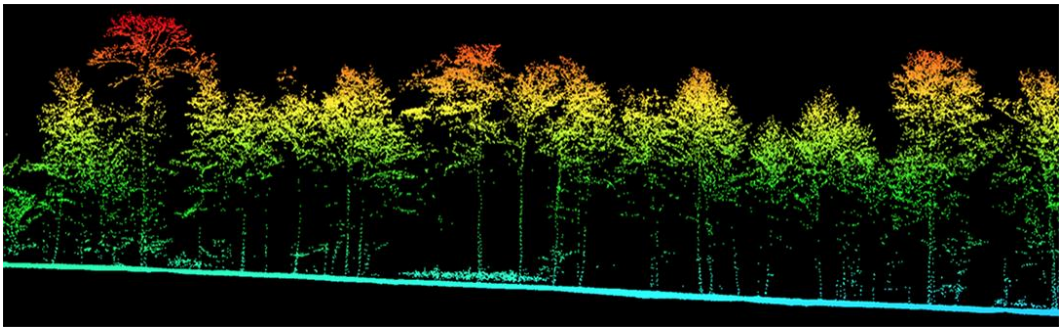

Figure 12: A cross-sectional view of the vegetation (Credit: Professor Susan Trumbore; UCI)

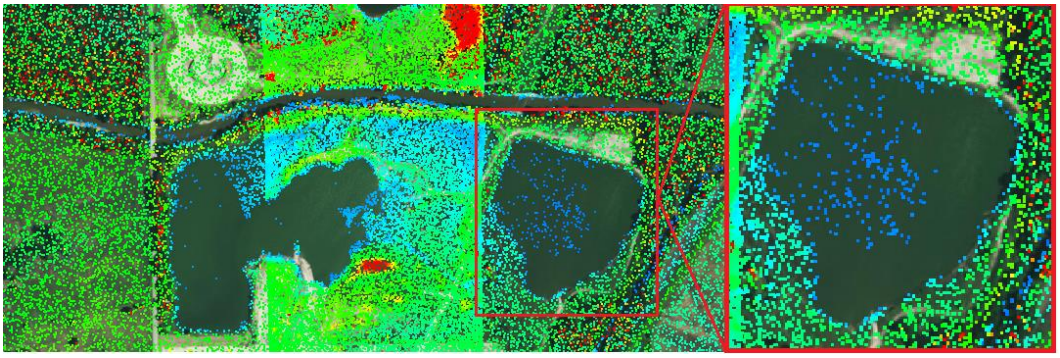

Figure 13: Low point density of LiDAR return from water surfaces

As LiDAR data brings a lot of information to the table, the data size tends to be large. Therefore, ground features or other useful information derived from the point cloud have more practical significance than the raw point cloud itself. However, manual feature extraction is very time-consuming, expensive and prone to error. This is even quite impossible for large-scale application. Therefore, automated feature detection algorithms are desirable for large scale dataset.

LiDAR point cloud has many practical applications that comprise an active research area. In this thesis, we describe two LiDAR-derived applications. First, we present a new automated algorithm for water surface mapping from a LiDAR point cloud. Second, we will show an elaborate algorithm to estimate the amount of solar potential of large areas using LiDAR 3D point clouds.

LAS File	Version	Point Count	Point Spacing	Z Min	Z Max
15_06723897.las	1.2	10,996,291	0.450	195.860	310.160
15_06733891.las	1.2	8,818,427	0.505	203.700	333.290
15_06733892.las	1.2	8,856,381	0.504	203.640	299.830
15_06733894.las	1.2	8,787,662	0.506	206.950	315.450
15_06733895.las	1.2	8,787,838	0.506	208.430	304.850
15_06733897.las	1.2	8,713,516	0.508	212.330	272.210
15_06753891.las	1.2	8,604,302	0.511	208.390	286.410
15_06753892.las	1.2	8,726,059	0.508	209.570	405.240
15_06753894.las	1.2	8,757,558	0.507	209.280	286.640
15_06753895.las	1.2	8,851,655	0.504	209.540	270.400
15_06753897.las	1.2	8,775,414	0.506	209.770	269.750
15_06763891.las	1.2	8,497,015	0.515	206.330	222.830
15_06763892.las	1.2	8,662,823	0.509	207.610	237.100
15_06763894.las	1.2	8,599,680	0.510	208.620	291.400
15_06763895.las	1.2	8,775,723	0.505	208.460	311.770
15_06763897.las	1.2	11,074,222	0.450	208.950	313.410
15_06783891.las	1.2	8,939,326	0.501	206.290	302.420
15_06783892.las	1.2	10,198,350	0.467	207.740	345.240
15_06783894.las	1.2	9,845,157	0.476	208.000	305.240

Figure 14: Statistics of a few LAS files from a typical LiDAR project.

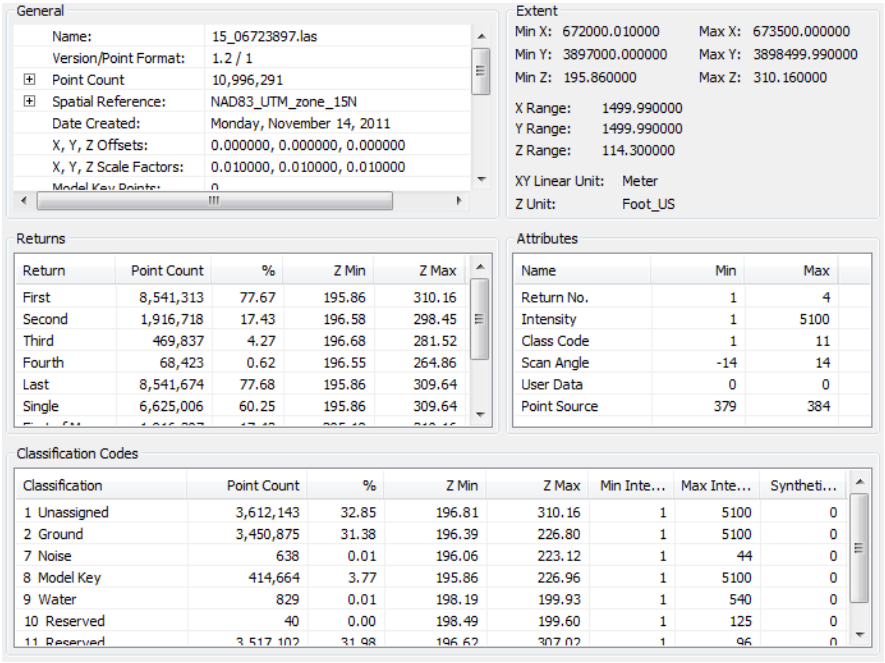

Figure 15: Statistics of a typical LiDAR LAS file.

Chapter 2

Still and flowing waterbody mapping

2.1 Introduction

A waterbody is defined as any physiographical feature containing water, which includes a pond, lake, reservoir, river, sea etc. Mapping water features play a vital role in planning any sustainability program of aquatic ecosystems and other environmental studies [16], [17]. Mapping water body boundaries is a very important and demanding requirement in hydrology and environmental science [18]–[20]. The USGS and the NRCS have been conducting an up to date nationwide waterbody mapping effort because of its importance to planning nature conservation services.

Traditional water-body boundary extraction approaches require manual examination, estimation, and annotation from high-resolution imagery. On-site surveys are usually limited, due to practical reasons, to only large sized water-bodies in a small selected area. On the other hand, seasonal variation of water level requires recent mapping in many applications, which is cumbersome with the on-site survey. This difficulty has invigorated remote sensing technologies which have brought many automated waterbody mapping approaches [21]. In contrast to field survey, remote sensing provides less expensive, less time consuming and more accurate set of software tools to efficiently map waterbodies. Satellite images, multi-spectral imaging, and LiDAR mapping are a few potential sensor outputs that can solve this problem effectively. However, use of multiple sensor outputs requires proper registration of these disparate data obtained over different times and conditions. This registration process, which at its core is a data preparation step, ends up becoming a major technical hurdle towards the actual exploitation of the data for useful application. In many ways, LiDAR has been a major breakthrough in that both elevation and intensity are available for every X, Y already registered, directly from the vendor.

Thus, researchers can actually focus on developing sophisticated algorithms to actually solve a feature extraction. This notion, then, was the motivation behind this dissertation.

Availability of LiDAR data has been increasing surprisingly in recent years because of new developments in the sensor industry, availability of increasing computation power, and developments in 3D point cloud processing tool sets. For instance, the United States Geological Survey (USGS) is conducting the 3D Elevation Program (3DEP) program to respond to the growing needs for high-quality topographic data and for a wide range of other three-dimensional representations of the nation's natural and constructed features. The primary goal of 3DEP is to systematically collect high-quality LiDAR data over the United States, Hawaii, and the U.S. territories, with data acquired over an 8-year period. Therefore, it is obvious that humongous amount of high-quality LiDAR data are going to be available in the near future. Hydro-flattening makes LiDAR-derived digital elevation model (DEM) accurate and consistent over water surfaces. Thus, mapping water body boundaries are also necessary to hydro-flatten LiDAR dataset.

As mentioned earlier, an airborne LiDAR system measures backscattering reflected light. The arrival time and the energy of the reflected light provide co-registered information about a target point’s elevation and reflectance respectively. The relative reflectance information can help to distinguish different objects with the same geometric feature. It has been observed that water shows high absorption and specular reflection in the near-infrared range, which provides additional information to aid in waterbody mapping from LiDAR return. However, simultaneously keeping track of the elevation and the intensity to draw manual boundary is very cumbersome and expensive. Therefore, automated or at least semi-automated detection or feature extraction algorithms are desired [22], [23].

Now, we are going to review the current methods to detect boundaries of water bodies from LiDAR. Most of the literature covers hydro-breaking generation in *coastal areas*, whereas only a few results were reported for river and standing or still waterbodies detection. Early model-

based approaches used historic shorelines and examined cross-shore LiDAR elevation profiles to detect a sudden change of elevation [24], [25]. Model-based approaches need nominal values of the magnitude of elevation change from some other sources or previous knowledge about the terrain. A few works tried to use additional information along with LiDAR data. Corrected LiDAR products such as DSM, DEM were used as primary information sources [26]; in some cases, ortho-images and hyperspectral images were also incorporated [27]–[31]. However, ortho-images and corrected LiDAR DEM are not always available. Furthermore, inland waterbody mapping is also necessary to hydro-flatten LiDAR data to generate accurate DEM.

LiDAR returns from waterbodies have specific geometric properties. The flatness of water surfaces is the most common geometric property. Therefore, in early works, the flatness property was mostly explored to detect water surface. The difference between the minimum and the maximum elevations within a 5m grid was reported as a useful derived property called vegetation height model (VHM) [32]. VHM, slope, the roughness of the surfaces are a few widely used geometrical properties. Besides the elevation information, it was also assumed by many researchers that the intensity of water surface return is quite low. Water surfaces show high absorption rate for near infrared region light. However, this assumption is not always true, especially for specular reflection return [33]. Actually, specular reflection from a waterbody occasionally causes reasonably high-intensity return. Therefore, intensity-derived features can be used as supporting features along with geometric features. Additionally, a smaller 2D point density is another common feature of water surface returns.

In [34], all of the above-mentioned features were assigned proportional weights. Data-driven intensity corrections and post-processing steps were also part of this approach. Waterbody detection based on this approach reported approximately 90% accuracy.

As intensity plays an important role in water surface mapping, model-driven and data-driven intensity corrections were proposed in [35]. After performing intensity correction and

dropout modeling, a seeded region growing based algorithm was proposed in a companion paper [33] based on a few features derived from flatness and low-intensity properties of LiDAR returns. In this paper, an overall 95% accuracy was reported in selected study areas. However, the method required knowledge of the sensor position, GPS time stamps, and scan angles in order to model the LiDAR drop-outs.

Co-registered RGB ortho-images were incorporated as additional information to derive an unsupervised mean-shift classifier in [36]. However, as mentioned earlier, concurrent image capture with LiDAR data collection is not always possible and, registering a satellite image with the corresponding LiDAR point cloud can be an extremely challenging and error-prone task.

Additional information such as historical borders may lead to better classification. In [37], a support vector method (SVM) based classification method was proposed, which used data density based features and required historical borders as references. Additionally, LiDAR-derived geometrical features were integrated with full-waveform LiDAR attributes to substitute ortho-images.

A few techniques were developed to extract geomorphic features from the corresponding digital elevation model (DEM). These techniques were designed to detect both natural and artificial channels (ditches) as well as differentiate between drainage networks and engineered structures, such as roads and bridges [38], [39]. However, the technique of looking for a center-line and of deriving the boundaries of the shore from that line is not applicable for inland closed water bodies.

In recent years, two new algorithms, large-scale automated standing waterbody extraction (LASWE) and multi-scale LASWE, were proposed for inland standing water body detection, which was demonstrated on large scale standing water bodies [40], [41]. First, flat land and water surfaces were segmented by detecting peaks in the elevation histogram. Thereafter, flat lands were removed by an elevation-based SVM classifier and intensity based maximum likelihood classifier.

It should be pointed out that the papers discussed above deal with two broad problems: a) shoreline/coastal-line extraction and b) standing water bodies and river extraction, the subject of this dissertation.

In our proposed algorithm, geometric and optical properties of a waterbody were combined to extract different water features. By definition, a seed is the starting point of a feature, which eventually covers the whole feature by following a set of designated rules. Seed based methods only analyze potential areas around selected seeds, which make it suitable for large-scale implementation. A new large dynamic range edge detection method called angular filtering was also proposed to extract sharp local change of elevation [42]. Quantitative results of the automated WB extraction are presented for three datasets, for which the WB boundaries were also manually extracted; in other words, where ground truth is available. Qualitative performance of the automated extraction is also presented for visual inspection for a very large-scale dataset as detected water body polygons overlaid on RGB images. Superior accuracy and significantly low computational time are key advantages of the proposed method, which enable very large scale practical applications. Once validated, our proposed algorithm does not require ortho-images, historical boundaries, DEMs or classified LiDAR point clouds.

2.2 The Multi-pronged Logic of Our Approach

In the following, we present the sequence of logic that was used to develop our algorithm. Conclusions reached from a detailed analysis of previous work were liberally used and then several new techniques were developed to eliminate or minimize the disadvantages of the previous approaches. Here were some preliminary assumptions and assessments to get us started:

- We wanted to entirely stay away from any multi-sensor fusion approaches and strictly use airborne LiDAR.

- It is evident that LiDAR elevation and intensity can be used to map features from LiDAR. Before proceeding to the proposed method, elevation and intensity map needed to be explored to assess key features.
- The raw LiDAR point cloud is irregularly spaced and required higher processing time if directly used for feature extraction. As the proposed algorithm is focused on large scale application, derived and regularized image-like input is more desirable. This process is known as rasterization, which is a considered as a pre-processing step. Both elevation and intensity can be rasterized from the point cloud data.

2.2.1 Rasterization of 3D point cloud

First, the extent of the project was calculated, and the whole scene was divided by multiple grids. The area of each grid can vary based on application. In this application, we used a $2m \times 2m$ grid size. Each grid may contain multiple returns or no returns at all. For multiple returns, the property of the grid was calculated from the property of the returns in that grid. Here, 'property' refers to the LiDAR parameter we need to rasterize, which can be Elevation, Intensity, Class, and Number of returns. Now, the property of the grid can be assigned in various ways. Such as average, median, inverse distance weighted (IDW), mode, mean, median etc. An example of a point cloud, a regular grid, and the corresponding LiDAR returns are shown in Figure 16. It is also shown that the elevation of the grid is assigned from the corresponding returns.

IDW is a frequently used method to assign elevation of the grid from its returns. The returns far from the center of the grid get smaller weights than the closer returns. The representation can be shown as follows,

$$Z_b = \frac{\sum_{n=0}^{N} Z_n / D_n}{\sum_{n}^{N} 1/D_n} \quad (3)$$

where, Z_b is the elevation of the grid, Z_n is the elevation of n-th returns inside that grid, and D_n is the distance of the n-th point from the center of the grid. Elevation raster based on IDW of a typical LiDAR project is shown in Figure 17. Besides IDW, the minimum elevation of the points is also used frequently for ground filtering application. Median and average intensity are also frequently used for intensity classification. The number of points inside a grid provides an idea of LiDAR point density, which is also a useful feature in many applications.

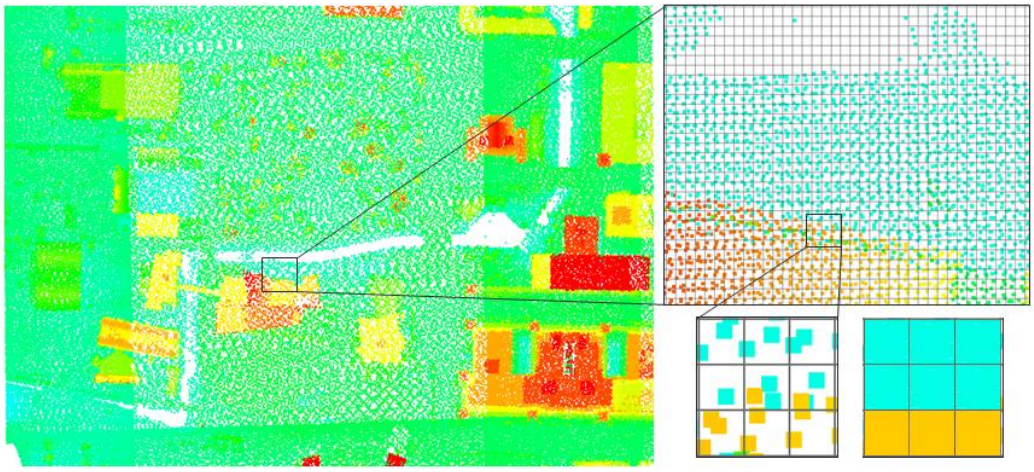

Figure 16: Irregular point cloud, grid on the point cloud, and multiple returns inside the grids.

In the process of rasterization, it is possible that many grids may not have any LiDAR return at all. This situation can happen when LiDAR return is unavailable for a large part of the scene. The laser beam may be fully absorbed or specular reflected and could not reach to the receiver. The unavailability of laser return from a specific part of the scene is known as a drop-out. This drop-out can be treated in various ways based on application. One way is to identify the drop out areas and use that information for algorithm development, as we have done in this dissertation. In this application, the drop-out areas were identified and saved as a drop-out raster. However, if the intent is strictly to convert an irregular grid to a regular grid, a useful approach is to interpolate the property of the grid from nearby neighbor grids. This nearby neighbor can be defined as the closest N neighbor, i.e., a neighbor inside a predefined block etc. The property of the grid can be

assigned from the median, average or IDW value of the neighbor grids property. The definition of IDW is same as the Equation 3. Here, Z_n is the elevation on n-th valid neighbor, D_n is the distance of the neighbor from the drop out grid, and Z_b is the calculated elevation of the drop out grid. The IDW interpolation scheme is shown in Figure 18. The intensity raster of a typical LiDAR project is shown in Figure 19. The high-intensity grids are shown in red and the low-intensity grids in blue color. The drop-out areas, are shown in white color.

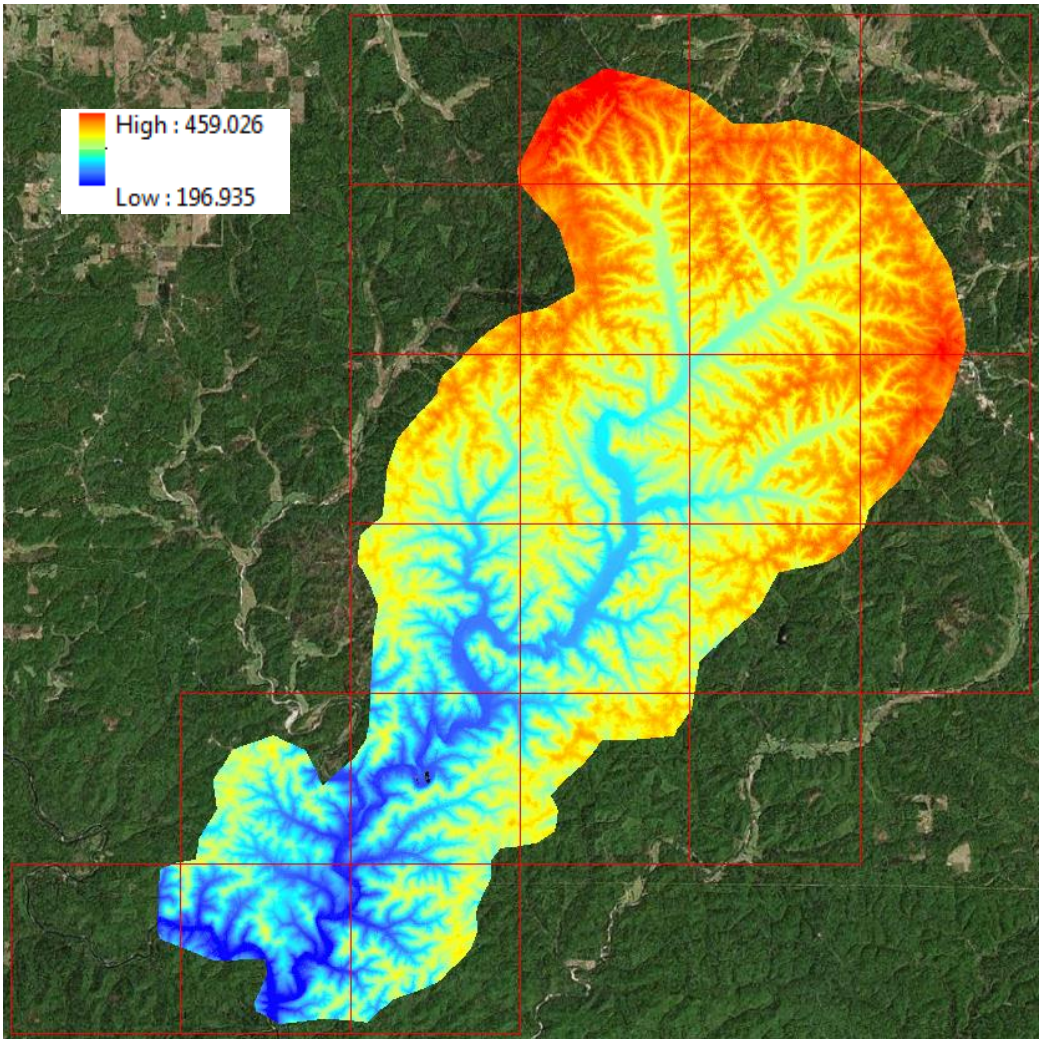

Figure 17: Elevation raster of a typical LiDAR project.

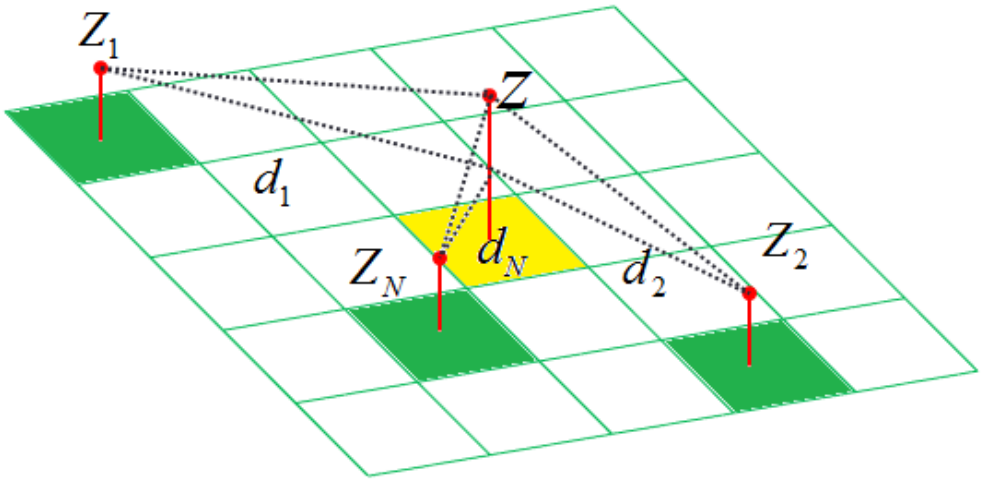

Figure 18: Estimating values of drop-out (Yellow) from neighbors (Green) value using IDW.

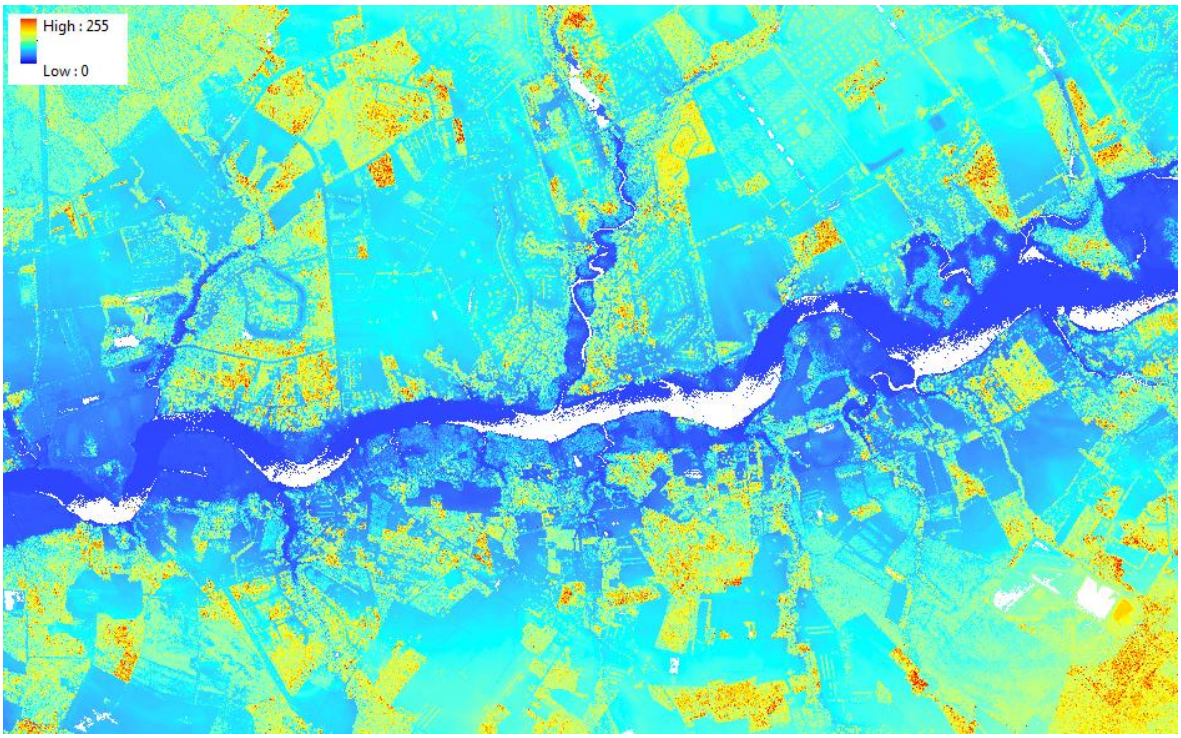

Figure 19: Intensity-raster with drop out region (White).

In summary, rasterization provides regularly spaced image-like data using a different parameter of the LiDAR return. Value for no return grid can be calculated by interpolation from the nearest neighbors or a left as it is. In any case processing the rasterized data is much faster than dealing with an irregular 3D point cloud. In this application, we derived four rasters: elevation, intensity, drop-out, and vegetation.

2.2.2 Hypothesis: Flatness of water surfaces

From the derived raster of many LiDAR projects, we extensively examined the data properties to identify key properties of a water body. This resulted in three hypotheses to identify water surfaces. Now, these hypotheses will be elaborated upon before describing the details of the algorithm. For visualization, sample waterbodies in Figure 20 were used.

Figure 20: Aerial view of two sample waterbodies.

Local elevation change appeared to be the most useful geometric property for different feature extraction schemes from rasterized LiDAR data. Elevation change has a large dynamic range from few inches to hundreds of meters depending on the terrain. Generally, elevation gradient is low for natural features such as fields, rolling hills etc. and high for man-made features

such as building, tower, dam, bridge etc. Therefore, distinguishing different levels of low elevation change is more useful than distinguishing high elevation changes, when natural features detection is the main concern, such as in rural areas. It is given that a water surface is comparatively flatter than other features. Furthermore, a water-to-land interface has a higher chance of local elevation. To capture the local elevation changes, we developed a new idea called angular filtering, which takes elevation raster as input and provide a map of an elevation change measure.

The following is the logic and the steps involved in our Angular filtering idea. A change of elevation was calculated for each pixel from the elevation raster. A $N \times N$ block was formed around each target pixel. For a target LiDAR pixel, *2* neighboring tiers were selected for angle calculation as shown in Figure 21. The middle red pixel is the designated target pixel; the yellow and the green pixel rings are the first and the second tier neighbors respectively. Elevation differences of all neighboring pixels from the target pixel were calculated. The range of this change can be large. In order to represent this range optimally, we developed the angular filtering technique. The elevation change is converted to change of angle, which then compresses the range in a manner similar to a non-linear quantization approach. That is, the elevation changes were non-linearly mapped to an angle θ_i using the inverse tangent function shown in Equation 4. The maximum angle change θ_{max} was calculated from all corresponding θ_i.

$$\begin{aligned} \theta_i &= \tan^{-1}\left\{\frac{Z_i - Z}{d_i}\right\} \\ \theta_{max} &= \max\{|\theta_i|\}\ \forall i \end{aligned} \quad (4)$$

where, Z_i is the elevation of the i^{th} neighboring pixel in a given tier, d_i is the Euclidian distance between the i^{th} pixel and the target pixel, θ_i is the corresponding angle and Z is the elevation of the target pixel.

The slope angle θ_{max} (0°-90°) was assigned as the angular filter output for the target pixel. At the end of the angular filtering process, elevation associated with each pixel was replaced by the corresponding θ_{max} for that pixel. As shown in Figure 21 (left), the inverse tangent function efficiently compresses very large elevation changes to manageable angular changes. For instance, lower elevation changes map to larger angle changes and vice versa. θ_{max} closer to 90° indicate that the local region is rough, and θ_{max} closer to 0° indicate smooth terrain. Thus, the angular filtering can exploit elevation change information more elegantly than other popular edge detection methods in image processing, where the range of pixel amplitudes is usually limited to the range of 0 to 255. Angular filtering outputs for the two sample areas are shown in Figure 22.

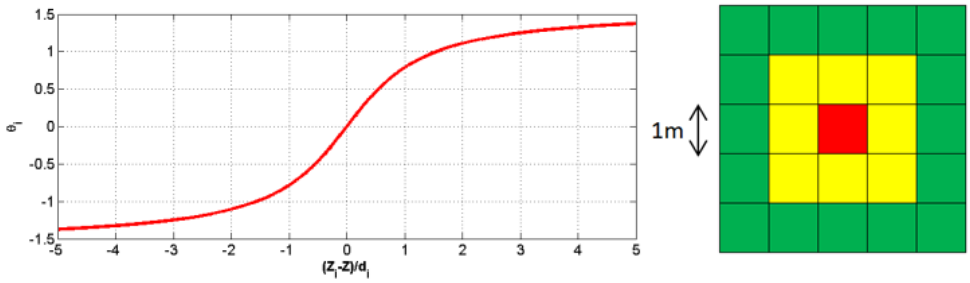

Figure 21: (Left) Elevation difference to angle mapping. (Right) The targeted block (red), first and second tier neighbor for angular filtering (yellow and green).

As stated before, a water-land interface shows noticeable elevation gradient in comparison to the water surface. Therefore, it can be expected that elevation gradient measured as the angular filtering value $|\theta|$ will be higher around the water-land interface. The elevation gradient is shown in Figure 22 for two sample water-bodies. Red and blue indicate lower and higher angles respectively. It can be seen from Figure 22 that the water-land interface has higher angular filtering value, which implies that the water surface is comparatively flatter than the water-land interface.

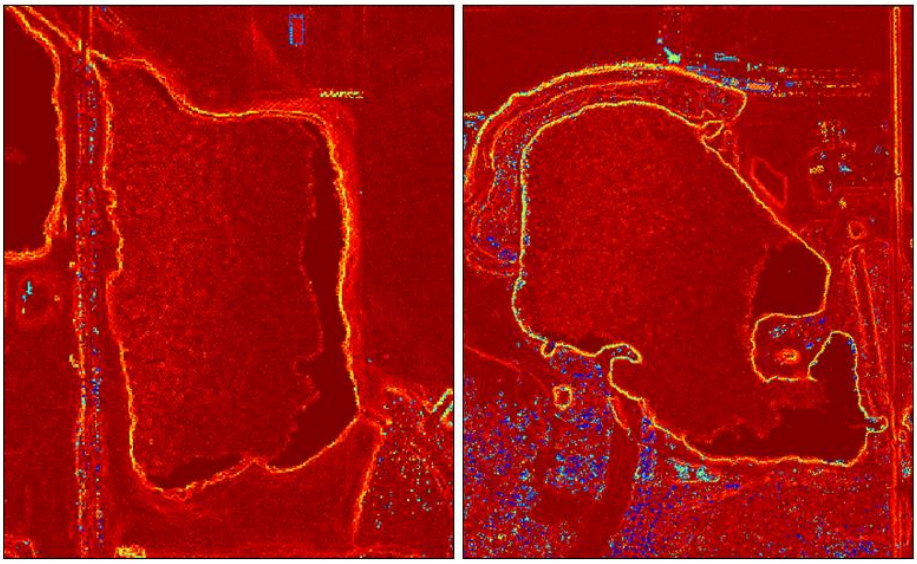

Figure 22: Maximum angular filtering of two sample water-bodies. Blue =90 and Red=0.

2.2.3 Hypothesis: Drop-outs due to no LiDAR return

A water surface exhibits different characteristics than other land and man-made features. As described before, a LiDAR receiver collects backscattering laser returns to measure the distance and the reflectance of the illuminated object. Specular reflection is prominent from a water surface because of its mirror-like properties, which leads to low return density and drop-outs. A drop-out is caused by no LiDAR pulse returns to the receiver because of either total specular reflection or total absorption. The water surface shows higher absorption rate for near-infrared lasers. In addition, a smaller number of returns to the receiver causes a lower LiDAR data density. Both of these properties were successfully utilized by different researchers to extract different water features.

At high scan angles, water surface exhibits higher specular reflection. Because of a higher chance of specular reflection and absorption from water surfaces, it can be assumed that at least a small part of the water surface will always have drop-outs. Drop-outs on the same sample

waterbodies are shown in Figure 23. LiDAR drop-out areas are shown in orange on the RGB ortho-images for the two sample waterbodies. It is clear that a small part of the whole waterbody has drop outs. Those drop-out regions can be used as initial search region to detect the corresponding water-bodies.

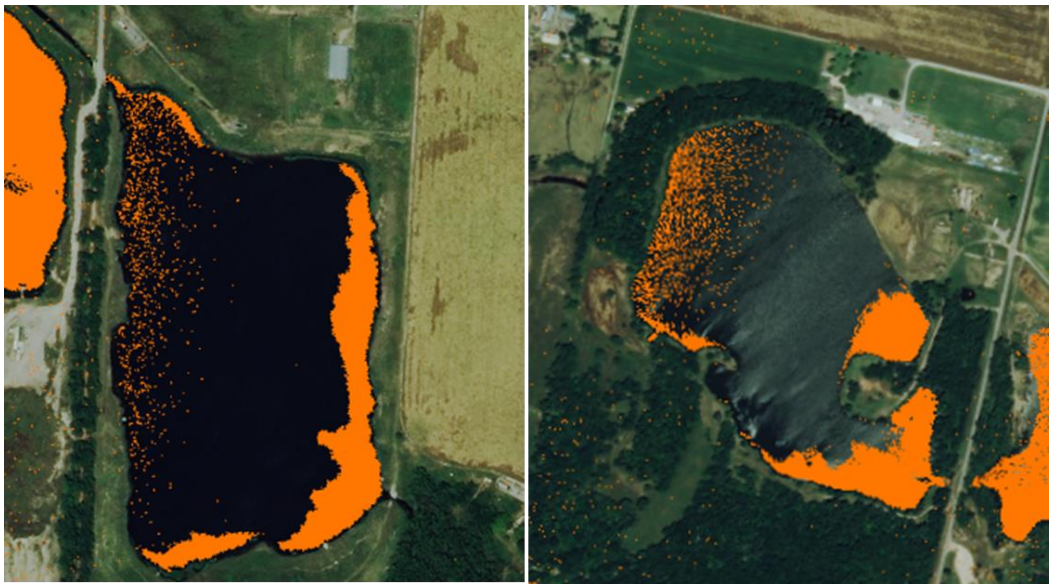

Figure 23: Drop-out areas (orange) are overlaid on RGB images for two samples waterbodies.

2.2.4 Hypothesis: Bi-modal intensity distribution

A water surface shows another important property. LiDAR returns are significantly absorbed by water, which leads to a very low-intensity return. But, specular reflection may be received by the scanner, when the scan angle is small e.g. less than 5 degrees. These specularly reflected receptions have very high-intensity value. In many situations, waves on water surfaces may be created in such a way, that the wave surface normal is aligned with the laser beam. In those cases, high intensity due to specular reflection is also observed. The whole situation is illustrated in Figure 24. At point a, when the scan angle is very small, the specular reflection is received by the receiver. At point b, the water surface is flat and the scan angle is high, which leads to missing the specular reflection beam. At point c, the laser beam perpendicularly hit the

water surface because of small waves, which leads to receiving the specular reflection. At point b, the laser return may have a small value if the diffuse reflection is received. Otherwise, it will be a drop out region on the water surface. Therefore, LiDAR returns from water surfaces are more likely to have very low (only absorption) or very high intensity (specular). The intensity images for two sample water-bodies are shown in Figure 25. Blue to red indicates lower-to-higher intensity. It is clear from Figure 25 that waterbodies mainly have a very low intensity and only a small part may show very high intensity due to specular reflection. Thus, very low intensity can be combined with dropouts as a prominent feature to generate the initial search region. Although LiDAR intensity is not always reliable, intensity distribution can carry useful information along with the geometric properties.

The probability density functions (PDF) of the data can be examined for a better understanding of the intensity distribution. From the PDF of a water surface intensity distribution, it can be observed that water-bodies have peaked at very low and very high ranges. This property is utilized in the Kolmogorov-Smirnov [43] test to differentiate PDFs of water and land surfaces. In Figure 26, the PDF and the cumulative distribution function (CDF) are shown for the same sample water-bodies and two large areas with both land and water features. Blue lines indicate plots for water-bodies and red lines indicate plots for land features. It is clear from Figure 26, that the water surface intensity is more likely to have an early PDF peak than a land surface. Therefore, the CDFs from a water feature and a land feature show a significant difference. This difference can be measured by the Kolmogorov-Smirnov test to distinguish between the corresponding PDFs of water and land features.

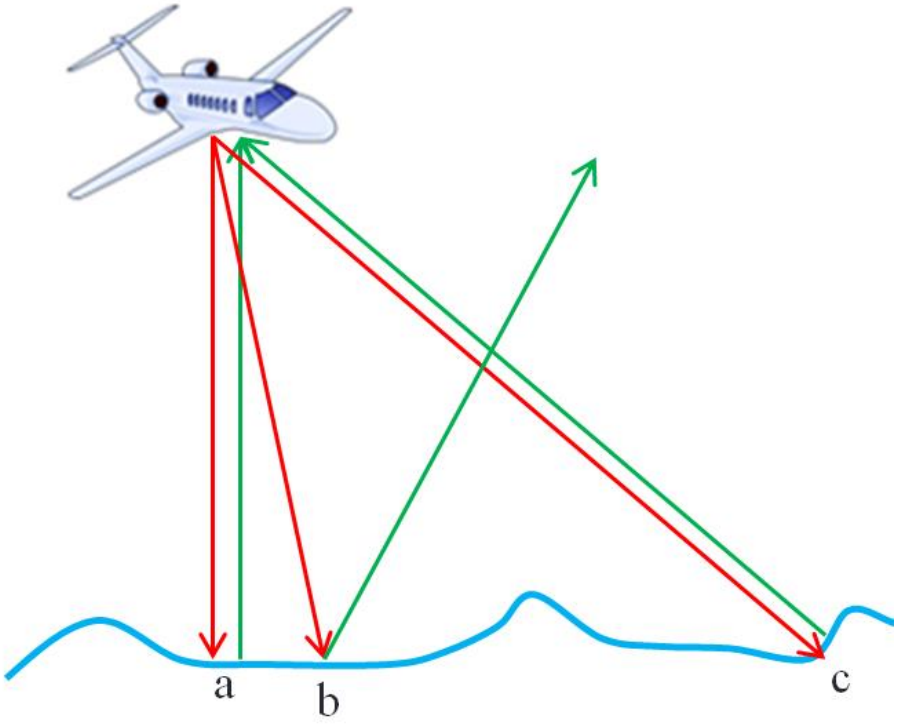

Figure 24: Simplified illustration of different scenarios of specular reflections.

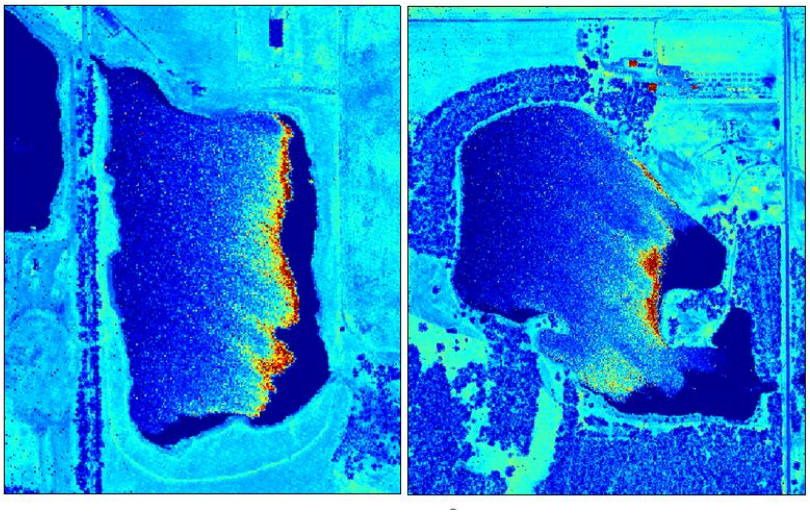

Figure 25: Intensity map of two samples waterbodies. Blue (low intensity), Red (high intensity).

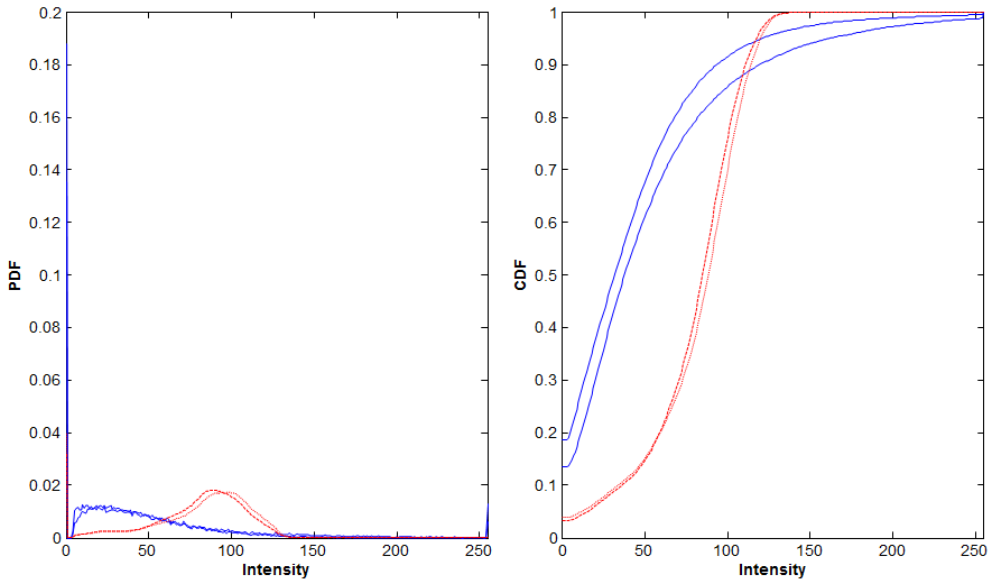

Figure 26: Blue lines: PDF and CDF from water surfaces. Red lines: PDF and CDF from the land feature.

2.3 Elements of our algorithm based on the hypotheses

It may be remembered that an inland waterbody boundary detection algorithm for very large areas was the objective of this proposal. LiDAR return properties from water surfaces were utilized to develop a computationally fast automated algorithm. As described in the previous section, three hypotheses were considered to develop this algorithm. a) At least a small portion of any waterbody has LiDAR return drop-outs due to specular reflection. b) LiDAR return intensity is more likely to be very low (high absorption) or very high (specular return). c) Local change of elevation is higher in the water-land interface than on the water surface. These three key characteristics were exploited to address the detection of inland waterbodies. The whole process flow of the algorithm is shown in Figure 27. Detailed steps of each working block are described below.

2.3.1 Rasterization phase

We already discussed that working with point cloud costs both computational power and time. On the other hand, rasterization quantizes the irregular 3D point cloud to a regular 2D gridded image-like data, which can be readily processed using the available image processing software tools. From a 3D point cloud, four different rasters were produced for 2m×2m pixels. This small block size ensures a low quantization error.

For an elevation raster, the minimum and the maximum elevation of the corresponding 3D-points was set as the elevation of the entire raster. Therefore, two different rasters were calculated— minimum elevation raster, maximum elevation raster. The minimum elevation of the 3D point cloud is commonly used for DEM generation. Both the minimum and maximum elevation raster were used to estimate the vegetation location. Additionally, for those grids which have no LiDAR returns, elevations were estimated from a surrounding raster using the inverse

distance weighted (IDW) interpolation method. This whole process is performed using *lasdataset to raster* function of ArcPy package.

Intensity raster was generated from the average intensity of the 3D point cloud. The intensity of a specific raster is calculated from the average intensity of the corresponding 3D points inside that raster. Here, the average operator was used for the robustness of the generated raster. For the intensity raster, the intensity of drop out regions was not estimated from interpolation. As we considered that drop out regions is a part of the water surface, the intensity is set to zero for all drop out grids. This rasterization process was also performed using *lasdataset to raster* function from ArcPy package.

The drop out raster was created from the initial intensity raster, before assigning zero values to the drop-out region. The ArcPy package assigns a large negative value, where no returns are available. A binary drop-out raster was generated from those large negative values, in which the raster value is set to "1" if it has any LiDAR return, otherwise to "0".

2.3.2 Processing phase

The elevation, the intensity, and the drop-out raster were the inputs to the processing phase. From these three rasters, three more rasters were processed in this phase; the vegetation raster, the angular filter raster, and the seed raster.

Vegetation height model: Knowing the vegetation location is important to remove overhanging vegetation regions from the waterbody boundary. The vegetation height model (VHM) can be derived from the elevation difference between the minimum and the maximum elevation raster [32]. LiDAR receiver receives multiple returns from different levels of vegetation. Therefore, the minimum elevation raster provides the ground or lower part of the vegetation, where the maximum elevation raster provides the top part of the vegetation. If the

height difference is larger than the defined tree threshold-h, then that raster is considered as vegetation raster. The vegetation raster can be formally defined as follows:

$$VHM(i,j) = \begin{cases} 1 & if \quad Z_{max}(i,j) - Z_{min}(i,j) > h \\ 0 & otherwise \end{cases} \quad for \ all \ i \ and \ j$$

The VHM raster may have a few sparse non-vegetation grids inside vegetation because of discontinuity of canopy or returns from the same height level. Those small sparse discontinuities were closed by a morphological closing operation. This operation takes a structural element B, a 5×5 square, and performs the erosion of the dilation of an image A. The closing operation on VHM can be formally defined as follows,

$$VHM \bullet B = (VHM \oplus B) \ominus B$$

Here, $\oplus$ and $\ominus$ are dilation and erosion operations respectively.

Angular Filtering: To calculate the surface geometric property the angular filtering method was formulated. The details steps of the angular filtering were provided in an earlier section. The angle value θ is high, when the local surface around the target pixel is rough. On the other hand, smaller θ value indicate smooth local surface. The angular filtering raster was calculated for the minimum elevation raster.

Seeds generation: From the second hypothesis, we assumed that at least a small part of the water surface is a drop-out region. Therefore, the dropout regions were identified to start the boundary search. Unfortunately, land areas also have a few sparse drop-out regions. Those drop-out regions are generally sparse. The sparse drop-out regions are eliminated by the morphological opening operation. The opening operation on the drop-out raster can be formally defined as follows,

$$DropOut \bullet B = (DropOut \ominus B) \oplus B$$

Here, $\oplus$ and $\ominus$ are dilation and erosion operations respectively. B is the structural element, which is a 5×5 square.

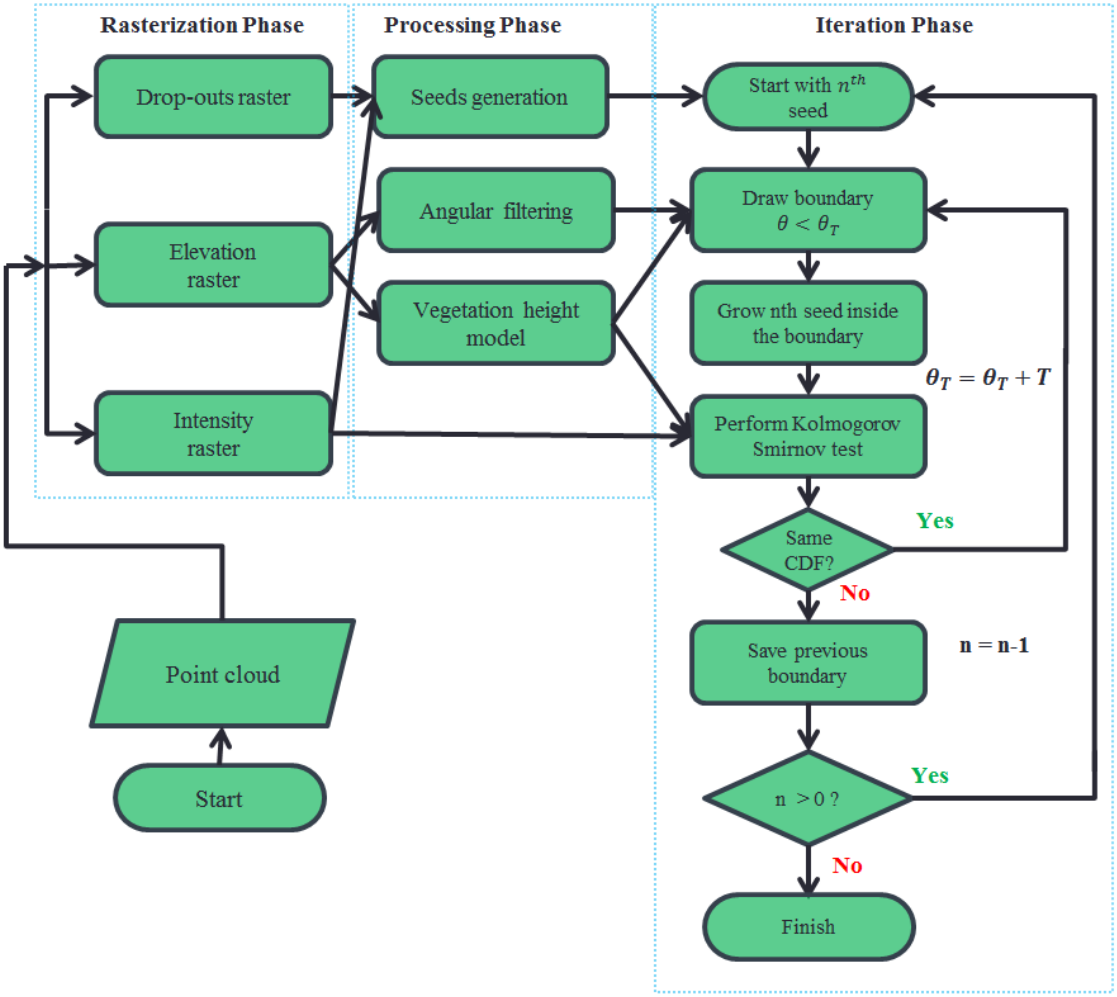

Figure 27: Algorithmic flow-chart of the proposed waterbody boundary mapping method.

After removing sparse drop-outs, there may still be small areas, which are not a seed or may be multiple seeds from a single waterbody. These small seeds were removed by a connected component analysis. Only those drop-out grids were considered as seed, which has linkage with a

certain number of seeds. A connected component is only considered as a seed, if a minimum number of drop-outs are connected. This minimum number can be varied by application. We considered that to be a seed, a minimum of 100 drop-out grids should be connected. This value can be defined by user based on project, and minimum waterbody size to be detected. An example of a connected component analysis is shown in Figure 28. In this figure, minimum element size is considered as 10. Therefore, only three components 1, 2, and 3 were selected (green), and other (red) are eliminated from the scene. Each connected component was ranked according to the number of drop-outs in it, and treated as initial search regions for water surfaces. Therefore, the higher ranked seed indicates larger waterbody around its vicinity. A single waterbody may have multiple seeds from different parts of the drop-out region; all of those were combined in the iterative phase. The waterbody searching phase starts from the largest seed to map the largest waterbody first; this may contain some smaller seeds from the same waterbody. Eventually, all seeds are considered.

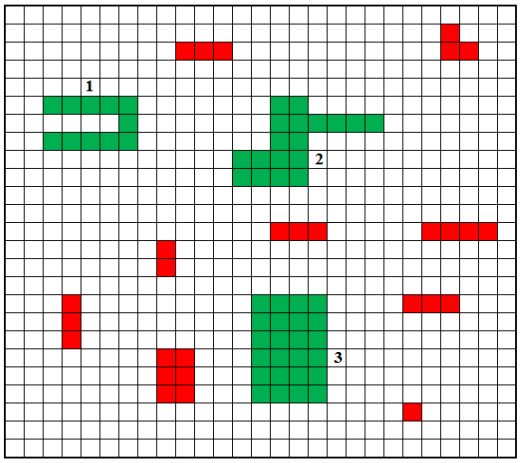

Figure 28: Connected component (Green), sparse elements (Red)

2.3.3 Iteration Phase

The iteration phase started with seeds generated from the drop-out raster. A seed was hypothesized to be a small portion of a water surface. The area around that seed was included in

the hypothetical waterbody, if the elevation and the intensity feature matched with the seed. Now, recall that a still waterbody is locally flat compared to its water-land interface. On the other hand, a large waterbody such as a river has decreasing elevation profile, although, in a local neighborhood, it can also be assumed to be flat. Therefore, a seed can be grown to a larger region by monitoring and adding surrounding pixels with low angular filtering value and low intensity. Note that after a water-land interface is crossed, subsequent angular filtering could result in a low value too, if the surrounding land surface is also planar. Therefore, it is critical that the water-land interface is correctly detected.

Water-bodies with vegetation on the banks were also enclosed by the vegetation raster. This enabled the detection of water-vegetation interface.

Thus, the overall approach of the iteration phase is to start with a minimum angular difference as the criterion to add new "water" pixels to the seed. Thereafter, the threshold was increased gradually to add more and more "water" pixels to the growing region. When the real water-land interface is crossed, a large amount of land surface or vegetation would be added to the growing region. This situation is equivalent to a virtual "flooding" of the area surrounding the actual waterbody. This virtual flooding was detected from the corresponding LiDAR intensity of the land and vegetation. The flooding is then reversed to yield the correct waterbody.

The proposed algorithm was also designed for the rivers by considering it locally flat, although, the elevation profile is gradually changing. If there are gaps for rocks, boats and data noises, those were ignored by considering the whole waterbody as a continuous object. In the case of waterfalls, different seeds detect each section of the river. If it is mostly a dry river, then only the water will be detected as separate ponds. The detailed iteration steps are described below (Figure 27).

At the beginning of the iteration step, the largest seed was selected for the first iteration. Thereafter, the same procedure was followed for all the other seeds. For any seed, first a small

angular filtering threshold θ_T was selected. Pixels connected with the seed with that minimum θ_T and containing no vegetation were added to the growing region. If the newly added pixels were water, they should have very low or very high intensity. Thus, after adding new pixels, the CDF shape should be unchanged or only slightly changed. In case of the flooding scenario, a large number of land pixels will be added, which will significantly change the shape of the water surface CDF. The Kolmogorov-Smirnov (KS) test was performed to check the similarity of the CDFs. Details of the KS test is given at the end of this section. On each iteration, θ_T was increased with a step size of T degrees and new pixels were added. After adding new pixels, if no significant change of CDF occurred, those pixels were considered to be water pixels. In case the virtual flooding starts adding land pixels, the CDFs would differ significantly and the Kolmogorov-Smirnov hypothesis test would fail. At that point, the iteration was stopped and the previous water boundary was retrieved and saved, in essence by reversing the flood and going back to the normal sized waterbody. Thereafter, a new iteration was initiated using the next seed. If the region grown from any previous seed included the currently selected seed then the current seed was skipped. This case was observed frequently because the same water-body has multiple seeds from different parts of its surface, for example, when there was a bridge over a lake. The whole iteration phase was continued until all the seeds were processed.

The iteration phase for a single seed is illustrated in Figure 29. The detection process started with the larger seed from the right side of the water body, which is indicated in dark red. During each iteration, the angular filtering threshold was increased by a small value and a new region was added to the seed. Therefore, CDFs were also changed in all iterations. The added region and the corresponding CDFs are shown with a unique color for each iteration. It is clear from Figure 29 that the shapes of the CDFs were similar before flooding. At flooding, all blue pixels were added, and the CDF significantly changed its shape. This change of CDF was detected

by the KS-test. Therefore, the boundaries of the water surface from the previous iteration were considered to enclose the true water body.

For the sake of validation of our algorithm, as one possible method, the water boundary detected after the final iteration was compared with the corresponding registered satellite image and manual detection as shown in Figure 30. It should be noted that once the algorithm is validated with a few satellite image photogrammetrically registered to the LAS file, it doesn't need such registered satellite images over and over again for detecting other water bodies for large areas. This is an important advantage of this algorithm.

As we discussed above, the KS-test was utilized to compare CDFs from iterations. This non-parametric test can be used to compare a one-dimensional, continuous distribution with a reference. In this case, CDF from the previous iteration was considered as the reference distribution. KS-test empirically quantifies the distance between the reference and current CDF. It is important to note that adding more low-intensity pixel with the CDF may increase the distance with the reference CDF, but in this case, the new CDF will rise earlier. Therefore, if the CDF distance is increased because of current CDF raise earlier than the reference CDF, then it's still valid waterbody CDF. This fact was handled by performing one-sided KS test, which is described in details in next paragraph.

Let's consider, we have n and m samples from current and previous iterations respectively. The CDFs are $F_{current,\ n}$ and $F_{previous,\ m}$ respectively. We are trying to identify if these two sample come from the same distribution or not. In this case, the KS distance will be,

$$D_{cdf} = \sup_{I} \left(F_{previous,\ m}(I) - F_{current,\ n}(I)\right)$$

The distance D_{cdf} is the largest distance for any intensity value. The distributions of these two samples were considered different if D_{cdf} is larger than a threshold set by KS statistics. This condition is set as follows,

$$D_{cdf} > c(\alpha)\sqrt{\frac{n+m}{n*m}}$$

Where, $c(\alpha)$ is a multiplier based on the confidence level α.

α	0.10	0.05	0.025	0.01	0.005	0.001
$c(\alpha)$	1.22	1.36	1.48	1.63	1.73	1.95

The D_{cdf} is negative if the current CDF rises earlier than the previous CDF. Therefore, the KS test is still valid, if more low intensity water pixels are added with the previous water region. This is known as one-sided KS test.

Inside the waterbody, there may have been a few isolated gaps due to islands, boats, ships as well as random noise. Smaller gaps with similar elevation as the surrounding water surfaces were removed as random noise and floating objects. Only islands greater than a user-defined size (in our case 200 square meters) inside any water body were detected as non-water-body objects. Therefore, the detected waterbodies were rectified using another connected component analysis.

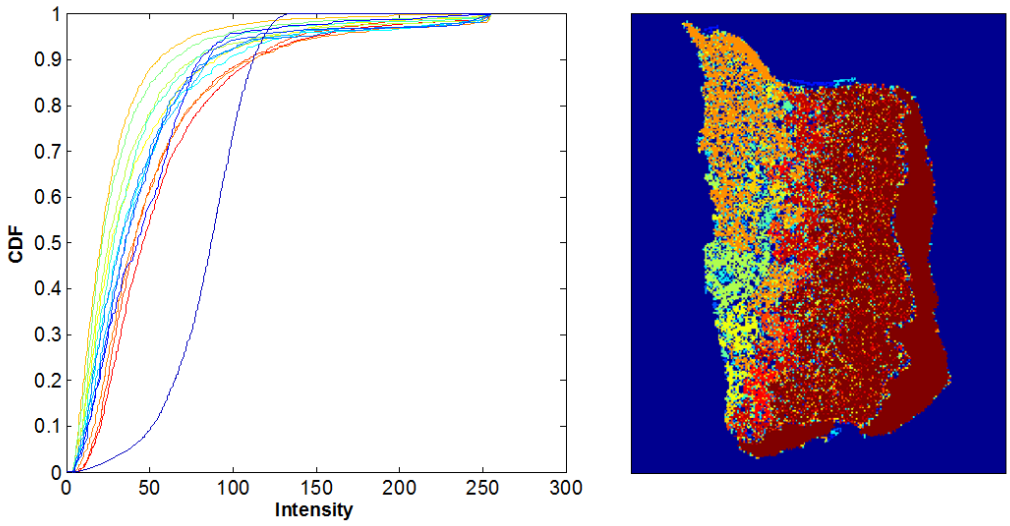

Figure 29: Change of intensity CDF in every iteration (Left). Added regions in iterations are shown in different colors (Right).

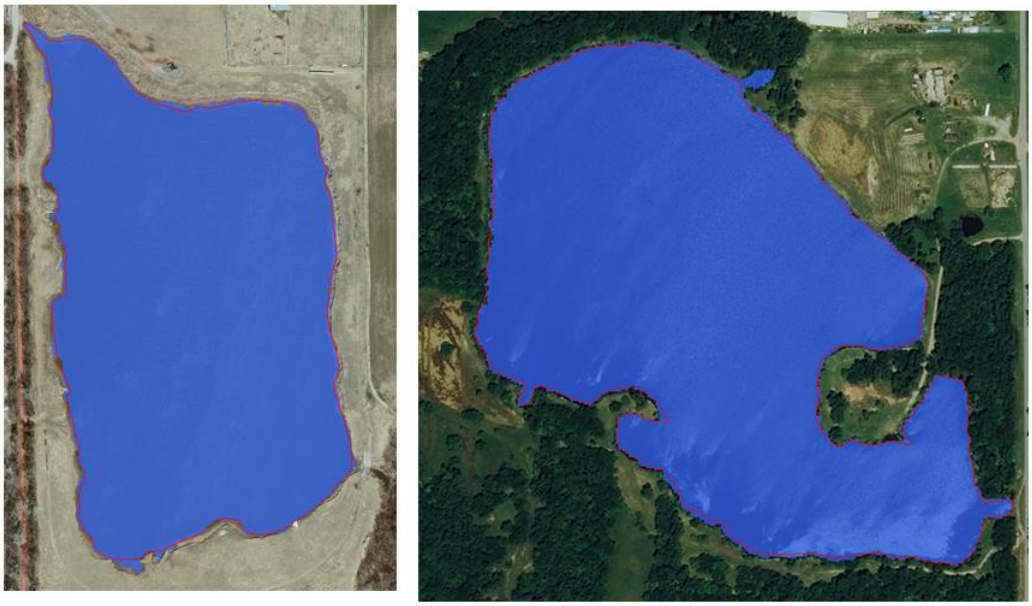

Figure 30: Detected waterbody (blue polygon) with aerial images and manual detection (red lines).

2.4 Large-scale implementation

The previous section describes the core algorithm from a theoretical perspective. In theory, we can consider a single raster for the whole project. But in practice, computation time, memory limitation, and project size all should be considered. In this section, we will discuss how this algorithm can be implemented for large scale project. We also introduce our Python-based implementation for large-scale LiDAR project. The steps of the large-scale implementation are shown in Figure 31 and will be discussed in detail in subsequent sections.

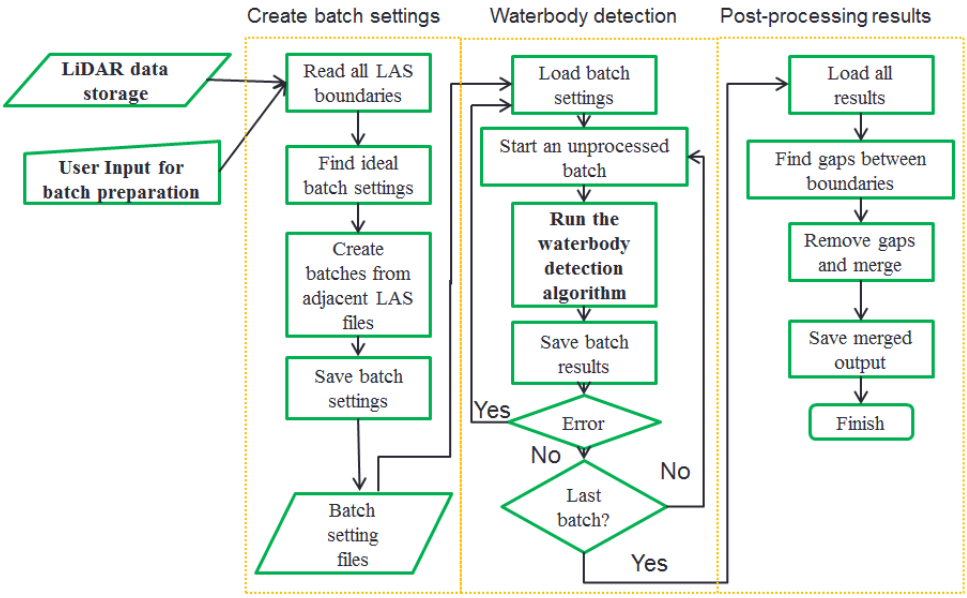

Figure 31: Large-scale implementation workflow.

Create batch settings: LiDAR data from large LiDAR projects are generally stored as LAS files in single or multiple directories. A single project may have multiple terabytes of LAS files. Generally, most of the LAS files cover a square shaped region but the area of the region varies. Therefore, the batch processing tool has to decide the right number of LAS files for a batch. Another problem is the locations of LAS files are unknown before processing them. A few

file naming conventions were attempted to help identify LAS files from the nearby areas but this is not true for all projects. Therefore, in out batch-setting phase, all LAS files extents were read. From LAS files geolocations, a grid-like neighboring relation was created for all LAS files. The user can provide an estimation of the dimension of a single batch. The algorithm defined the best batch setting from the LAS files size, user input, and the memory constraint of the machine. Batches were created by considering all of these parameters. The log of this operation is saved in a file for diagnosis, and the batch-setting was saved in a .pckl file.

The user interface for batch creation is shown in Figure 32. First four inputs are filed location and project name. The user can set the run method in *define run method* input. All LAS file can be run in as a single batch, or individual LAS files can be treated as a separate batch. These two are the fastest. No geolocation is needed to set the experiment in these two modes. The user can use the file name to create batches if the project supports any naming convention. Otherwise, the user can choose the automatic batch creation from files geolocation. In the *batch setting* input, the user can indicate the location of the row and the column number in the LAS files name, if naming a convention was used, and also can define the expected batch dimension. For example, if [1, 5, 6, 8, 5, 5] is given in batch setting then it means 1-5 letters of file name has the row number, 6-8 letters has the column number, and the expected batch size is 5-by-5. In the case of geolocation based batch creation, only 5×5 LAS files will consider, but row and column indications will be ignored.

The log of the batch creation is shown in Figure 33. We see that 45 batches were created from 661 LAS files. The log was saved in a text file, and the batch setting was saved in a .pckl file. The whole process took 21 minutes 43 seconds on an Intel i7, 16GB RAM, Windows OS machine.

Waterbody detection: Now that, the batches are ready, the original algorithm can run on each batch without any CPU memory related issues. In the second phase, the batch-setting .pckl

file was loaded to the original algorithm, which is shown in Figure 34. This tool runs each batch separately and saved the output in a temporary .tiff file. In the case of any unexpected freeze of the run, the batch setting file can be loaded again, and then the processing would resume from the same place. When all batches were processed, the post-processing steps started.

Post-processing: After the waterbody detection phase, detected waterbody maps from all batches were available. Now, it is necessary to merge all results to a single output. A single waterbody can occupy multiple batches, and detected as separate waterbody. A sample LiDAR project, LAS files boundaries, and batch boundaries are shown in Figure 35. From the figure, it is clear that a long river can occupy many batches, therefore should be treated as a single waterbody.

At the batch boundary, a single straight line shows up because of a boundary constraint over each batch. This straight line separates two parts of a waterbody from the adjacent batch. One example of waterbody occupying two adjacent batches is shown in Figure 36. To remove this line, a search was performed around the batch boundary. If a single line separates two waterbodies from adjacent batches, then that line was removed, and the two waterbodies were merged as a single one. The outcome of this unification process is shown in Figure 36.

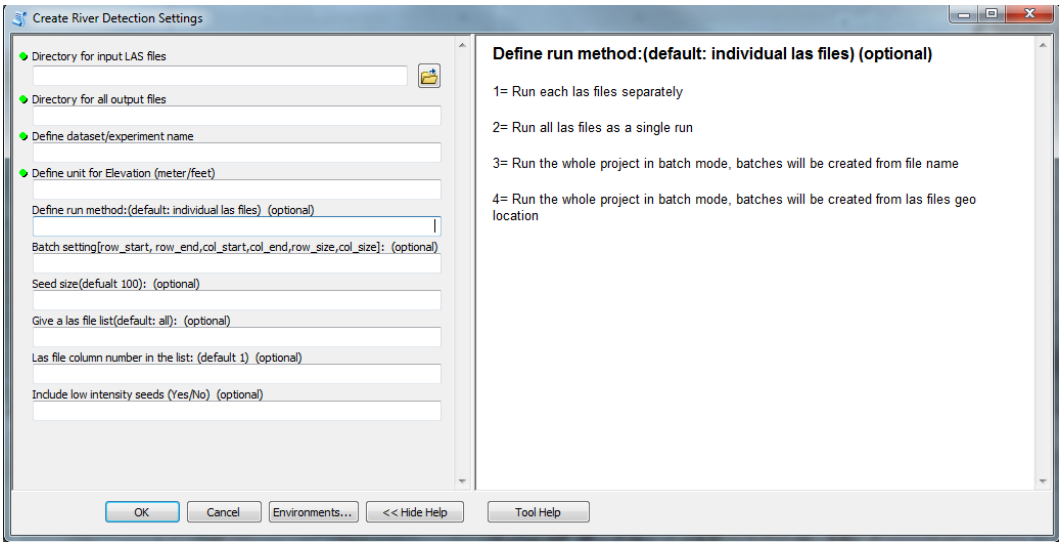

Figure 32: Create batch-setting user interface.

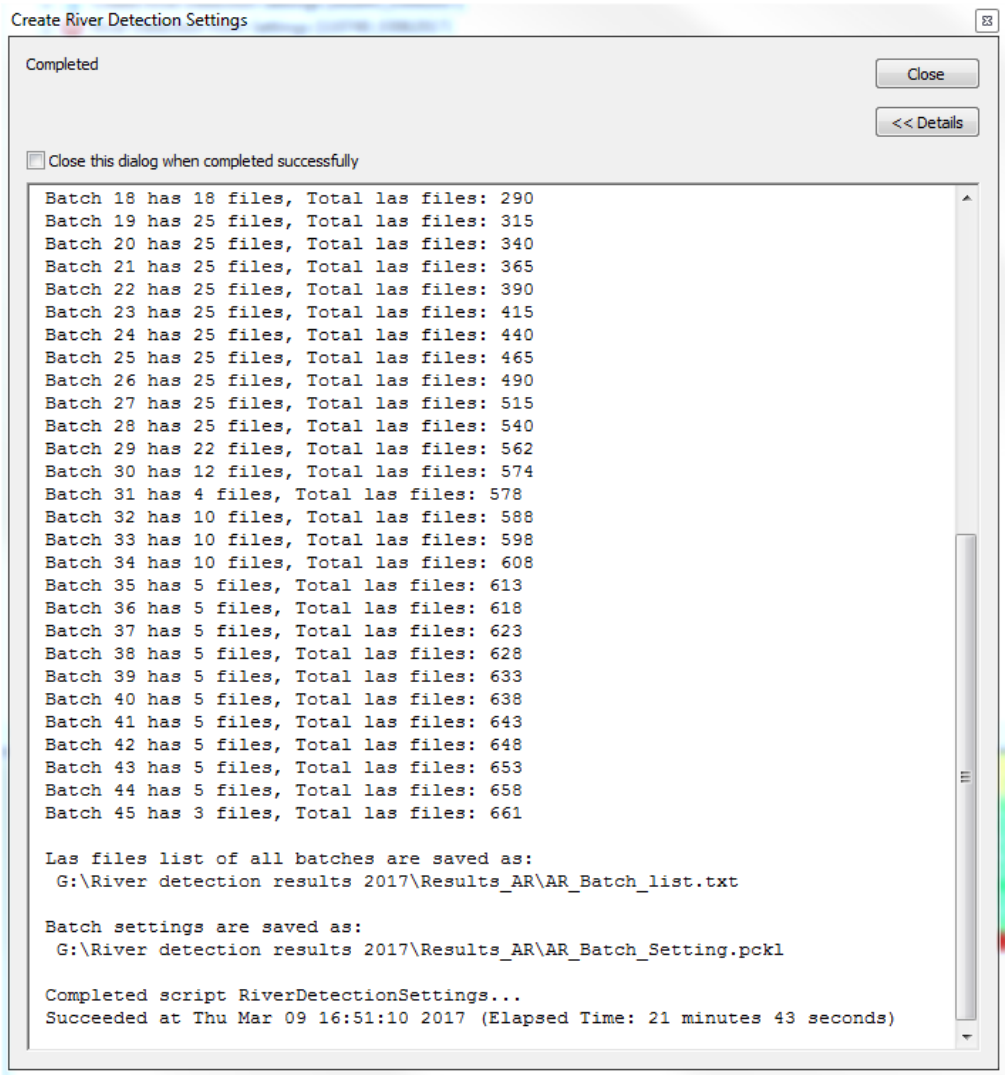

Figure 33: Batch setting logs.

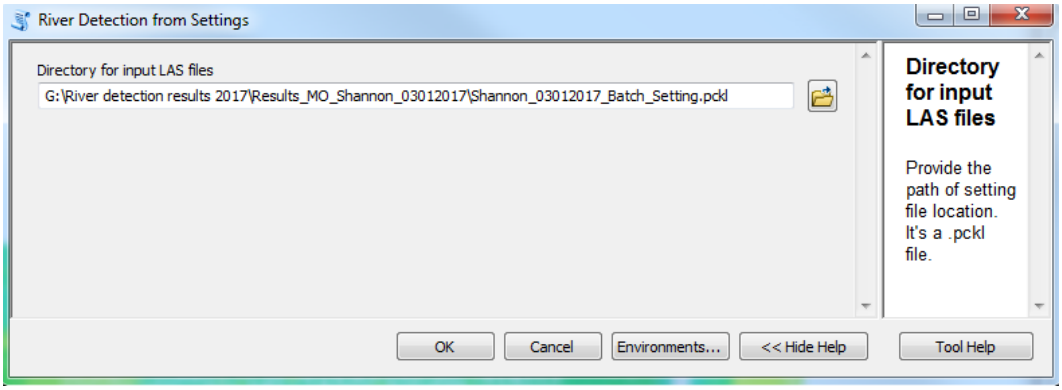

Figure 34: Load setting for processing.

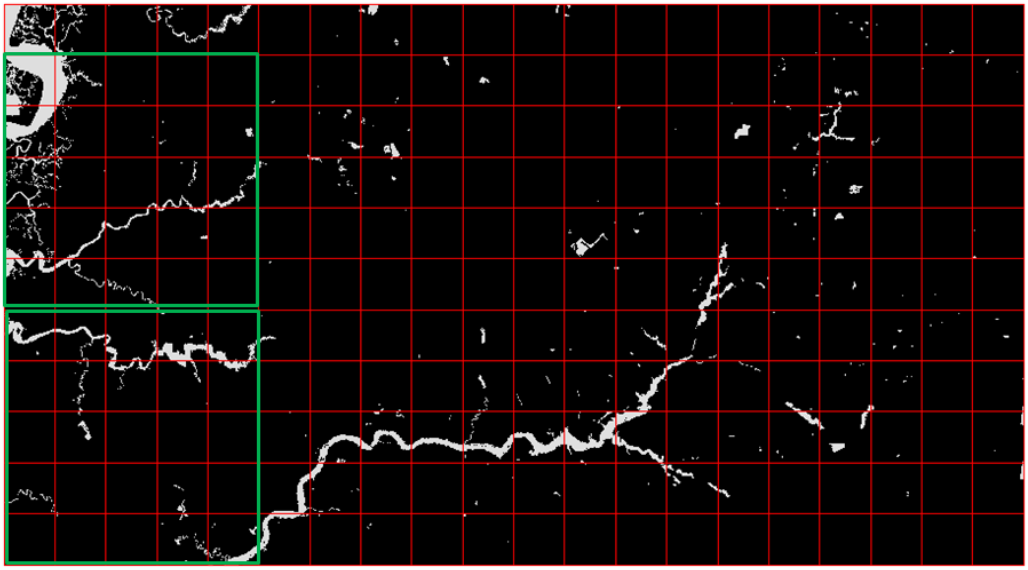

Figure 35: Batch boundaries, Las files boundaries, and detected waterbodies.

Figure 36: Waterbodies in multiple batches. Merged waterbodies after post processing.

2.5 Results and Validation of the Algorithm

Quantitative results are presented for three datasets from Arkansas, Nebraska, and Oklahoma, as described in the following dataset subsection. For these datasets, the reference waterbody boundaries were manually derived from the satellite image and LiDAR data by a private contractor. It may be noted that obtaining accurate ground truth of the water boundaries is challenging for a few reasons. First, the water level is variable over seasons. Second, the manual detection is tiresome and time-consuming and, prone to error.

2.5.1 LiDAR dataset

Four different datasets were analyzed to present the performance of the proposed method. Three smaller areas were used for small-scale quantitative analysis and a large scale dataset was used for qualitative performance analysis. For three small areas, ground truth boundaries of true water-bodies were manually detected, using a satellite image and the corresponding LiDAR data. Area-01 contains a small lake with a narrow channel and a small island. Area-02 has a large lake with multiple islands and submerged vegetation near the boundary. Finally, area-03 contains

multiple waterbodies in different sizes and shapes. The sizes of area-01, area-02, and area-03 are $9\ km^2$, $20\ km^2$ and $40\ km^2$ respectively. Quantitative and qualitative comparisons are shown for these two datasets.

The large datasets came from Maryland, USA. Maryland-river dataset covering an area of $500\ km^2$ contains the city of Salisbury and a part of the Wicomico River. This dataset demonstrates the performance of river detection in a rural as well as in an urban scene. The Natural Resources Conservation Service (NRCS) of Ft. Worth, Texas provided all of these datasets. Manually detected hydro-breaklines were also provided by NRCS for a comparative performance analysis of the automated tool. The nominal pulse spacing for the LiDAR data is less than 0.7 meters. LiDAR data was collected with a Riegl LMS-Q680i full waveform LiDAR sensor, which collected an intensity value, GPS week time, flight-line and echo number attributes for each discrete pulse.

2.5.2 Quantitative measure

The overall accuracy (ACC), specificity (SPC) and sensitivity (SEN) were derived as quantitative performance measures from a confusion matrix. In Figure 37, relationships among the quantitative measure are shown. Automated detection of waterbodies was represented as the red polygon derived from the detected boundaries. On the other hand, manual detection was represented with green polygons generated from the manually created breaklines. An overlapping operation was performed to detect *True Positive (TP)* area between these polygons. A union operation was performed and subtracted from the total area to calculate the *True Negative (TN)* area. The overlapping area was subtracted from the manual detection and automated detection to calculate the *False Negative (FN)* and the *False Positive (FP)* areas respectively. Calculation of all quantitative measures are formulated as follows,

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specifity = \frac{TN}{TN + FP}$$

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

Simple coefficient values of a performance measure may not fully provide the extent of the correctness of waterbody detection. Additionally, larger visual inspection may also be necessary for performance validation. Therefore, detected water boundaries were overlaid on registered satellite images to also obtain a qualitative performance measure.

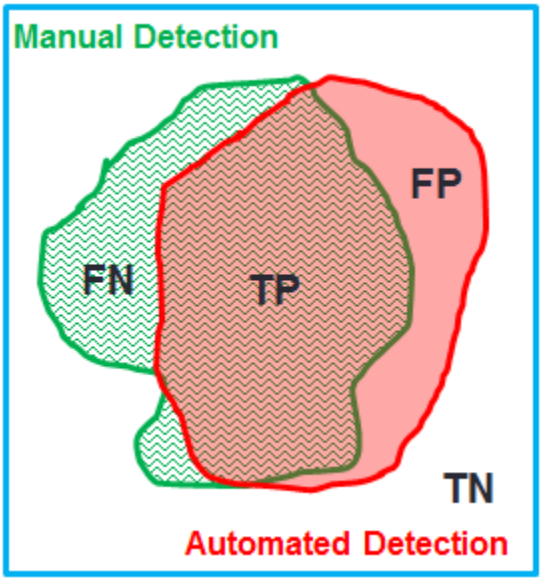

Figure 37: Quantitative measures from manual and automated detection.

2.5.3 Small LiDAR projects results

In this section, the quantitative results for small scale project are shown. Small scale projects were used for quantitative results because the detail and valid manual detection were possible only on this scale. For large scale application, a visual inspection will be provided in the later section.

First, in Figure 38, manually detected border and automatically detected waterbody polygons from the Area-01 dataset are overlaid on the corresponding satellite image. From the figure, it is clear that the lake and small channel from the lake are detected. The automatically detected waterbody boundaries visually matched with the manually detected break-lines, except for a few cases of submerged vegetated areas near the shore. The small island and a small waterbody in that island were also correctly detected. From the overlapping areas, it was observed that 0.043 km^2 of water surface was correctly detected; 0.002 km^2 missed and 0.002 km^2 were misclassified as water-surfaces in a 9 km^2 study area, resulting in an overall accuracy of 99.96 %, sensitivity of 95.16 %, and specificity of 99.97 %. As an approximate comparison between the levels of effort and the funding needed to performing the manual work is, thousands of hours of manual labor, that includes the classification, drawing the actual boundaries, vectorizing them etc. versus running our automated algorithm using the original LiDAR data. While we have not performed an accurate comparison a ball park comparison would be a 1000 to 1 in savings. The accuracies are comparable or better for the automated approach.

In Figure 39, manually detected borders and automatically detected waterbody polygons from the Area-02 dataset are overlaid on the corresponding satellite image. From the overlapping areas, it was observed that 7.02 km^2 of water surface was correctly detected; pixels corresponding to 0.077 km^2 were missed and 0.24 km^2 were misclassified as water-surfaces in the 20.345 km^2 study area. Therefore, the overall accuracy was 98.45 %, sensitivity was 96.71 % and specificity was 99.41 %.

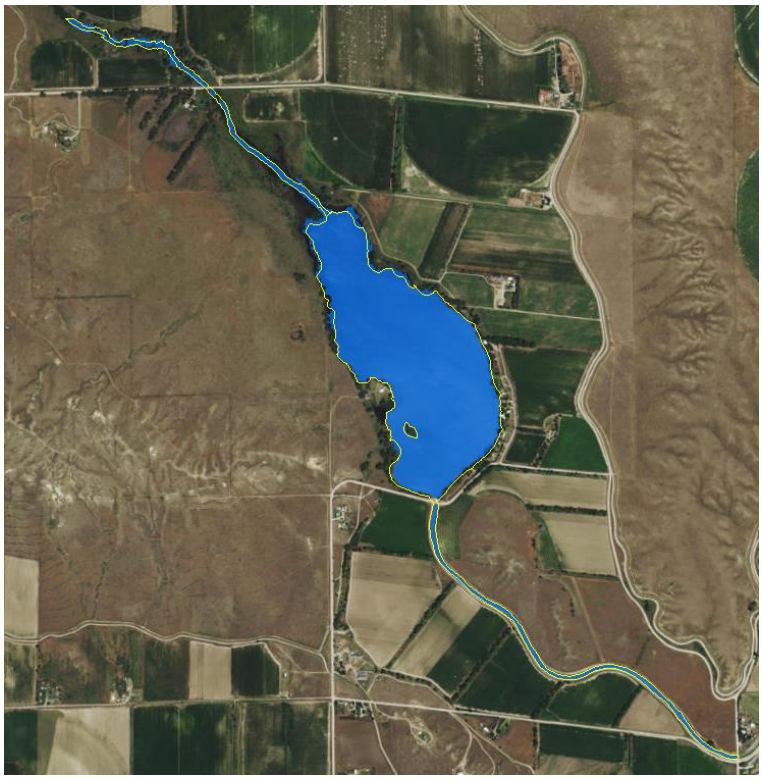

Figure 38: Detected water-body overlaid on satellite image for the Area-01 dataset. Manually detected border (Yellow lines) and automatically detected water-body (Blue semi-transparent polygon).

In Figure 40, manually detected border and automatically detected waterbody polygons from the Area-03 dataset are overlaid on the corresponding satellite image. As can be seen, this area is the largest of the three and, encloses more than one waterbody. It can be seen that the automatically detected borders of all seven water-bodies matched with the manual detection. The algorithm is able to detect smaller waterbodies too. In this case, a minimum size was given as a constraint to detect only larger waterbodies, for which manual detection was available from a contractor. From the overlapping areas, it was observed that 0.845 $km^2$of water surface was

detected correctly; 0.012 km^2 were missed and 0.01 km^2 were misclassified as water-surfaces in a 39.06 km^2 study area, resulting in an overall accuracy of 99.94 %, sensitivity of 98.60 %, and specificity of 99.97 %.

Figure 39: Detected water-body overlaid on satellite image for the Area-02 dataset. Manually detected border (Yellow lines) and automatically detected water-body (Blue semi-transparent polygon).

Quantitative results of these three datasets are summarized in Table 1.

Area	Size (sq.km)	Waterbody area (sq.km)	Accuracy (%)	Sensitivity (%)	Specificity (%)
01	9.00	0.05	99.96	95.16	99.97
02	20.35	7.09	98.45	96.71	99.41
03	39.06	0.85	99.94	98.60	99.97

Once the algorithm was validated using three relatively small geographic areas, it was applied over a significantly larger area. Detection performance for this large-scale dataset is shown in Figure 41.

Figure 40: Detected water-body overlaid on satellite image for the Area-03 dataset. Manually detected border (Yellow lines) and automatically detected water-body (Blue semi-transparent polygon).

2.5.4 Large LiDAR projects results

The Qualitative results for three large datasets are shown in this section. Unfortunately, for this dataset, similar to the most available datasets, valid manual detection is not available. Therefore, the performance of detection was assessed by visual inspection of the corresponding satellite images. The Salisbury area dataset is a 500 sq.km area, which covers urban areas with few rivers. The Manokin River dataset is a 125sq.km area dataset, which mainly covers wide rivers and river channels. The Ste. Genevieve dataset from Missouri is a 1500 sq.km area, which covers small waterbodies and a few lakes.

The Salisbury dataset contains a few rivers in both rural and urban areas. Additionally, there are many inland standing waterbodies too. In Figure 41, automatically detected water-bodies are shown in blue polygons and overlaid on the corresponding registered satellite image. From a visual inspection, it is clear that all the major waterbodies were correctly detected. The whole area is separated in five different areas (A, B, C, D, and E) to show the details result.

In Figure 42, zoomed version of area A is shown with satellite images. From the figure, it is clear that the main review and small branches are detected in the final result. In Figure 43, the enlarged version of Area B and Area C are shown. A few man-made small waterbodies, including a waterbody surrounded by vegetation are detected in this figure. We also see that small ponds and bridges were detected correctly in this part. Two more bridges were correctly detected in the area shown in in Figure 44. Long river branches and rivers were detected in Figure 45 area.

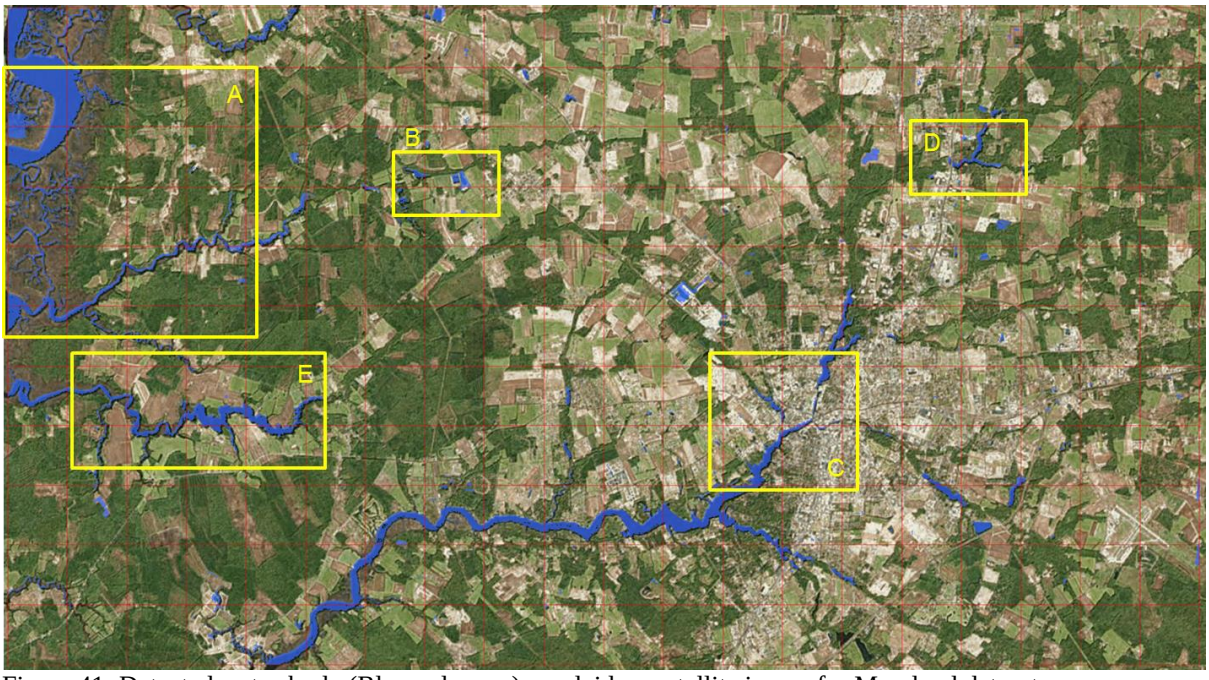

Figure 41: Detected water-body (Blue polygons) overlaid on satellite image for Maryland dataset.

Five interesting areas are marked and enlarged for better visualization in later figures.

Figure 42: Enlarged version of Area A of the previous figure.

Figure 43: Enlarged version of Area B and C of Salisbury dataset.

Figure 44: Enlarged version of Area D of Salisbury dataset.

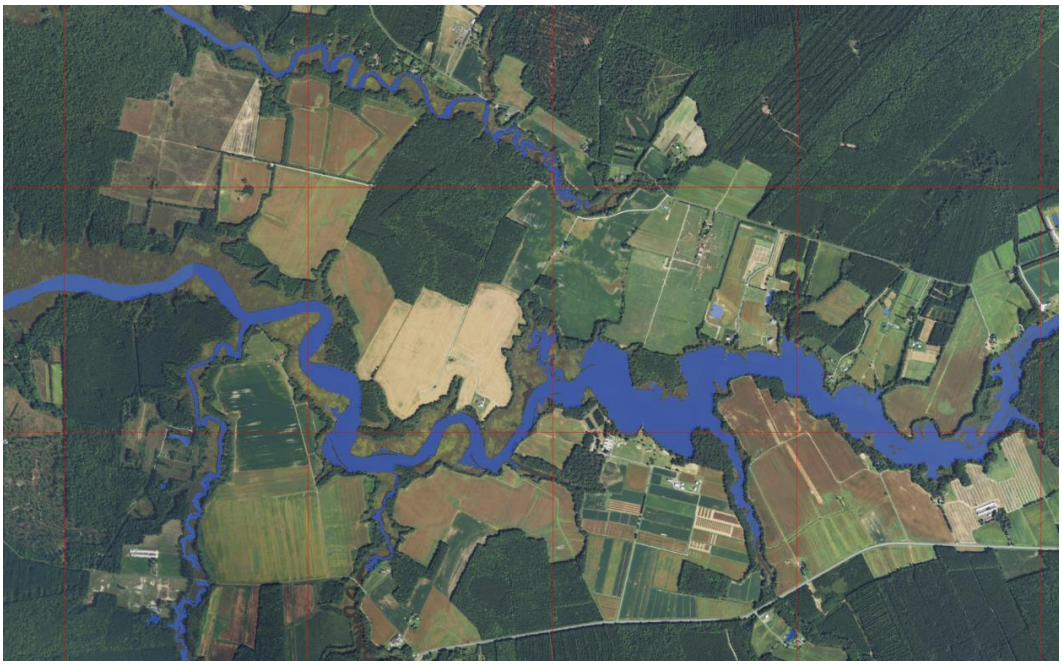

Figure 45: Enlarged version of Area E in Salisbury dataset.

In Figure 46, the overall detection and approximate manual detection are shown for Manokin River dataset. The automated detection is shown in the blue polygon, and the approximate manual detection is shown in yellow lines. It is clear that the manual detection and the automated detection are similar. The enlarged versions of a few parts of this area are shown in subsequent figures.

In Figure 47, a wide river and its small branches are correctly detected. In most of the cases, manual detection only cover a certain river width. Therefore, a few small branches are missing in manual detection but are correctly detected in our automated detection. This can be confirmed from the right side satellite image in Figure 47. In Figure 48, we see that the manual detection only covers the main river channel, but ignores the watershed areas. In our automated detection, the watershed areas were also detected. The watershed region may have low depth but still fulfill the local flatness and intensity criteria. In Figure 49, we can see that small winding river, watershed, and bridges were correctly detected. In Figure 50, a small winding river and

breaches were mapped. A flat field was detected as waterbody in manual detection but correctly ignored in automated detection.

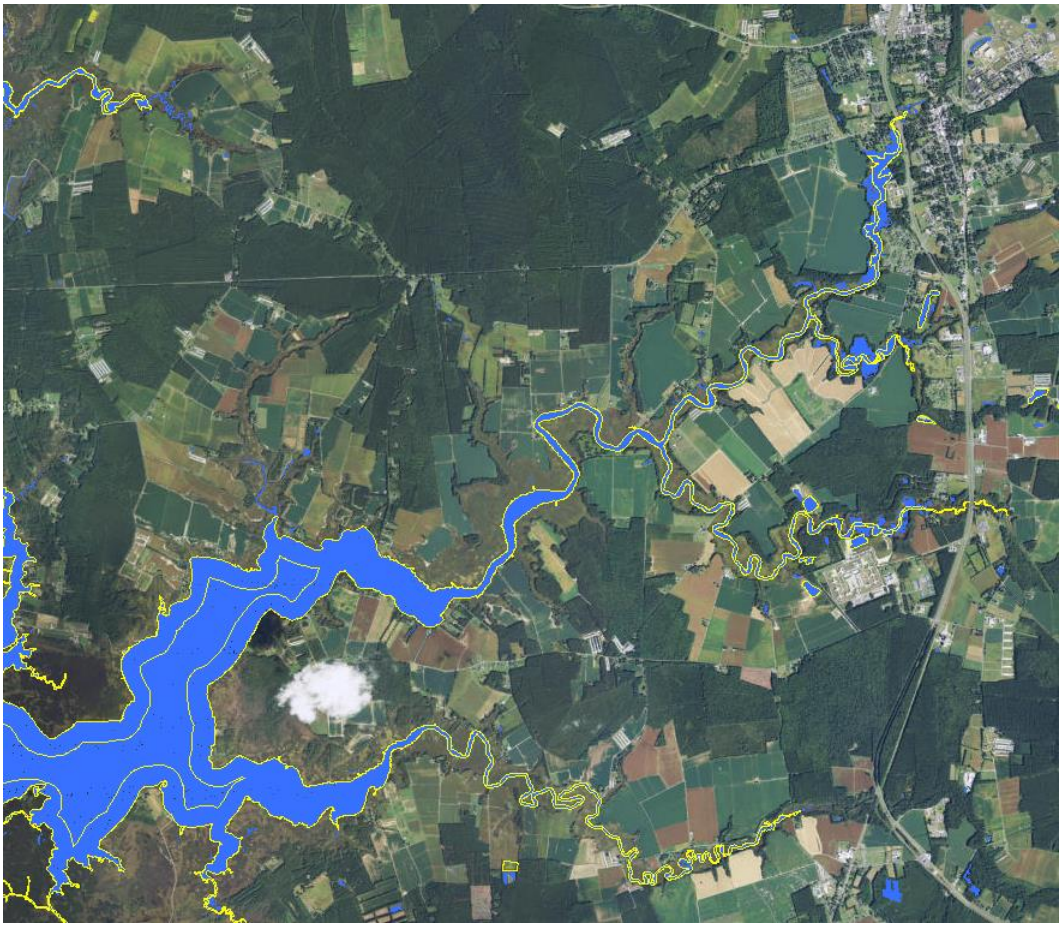

Figure 46: Manokin River dataset. Detected rivers (blue polygon) Approximate manual detection (yellow lines).

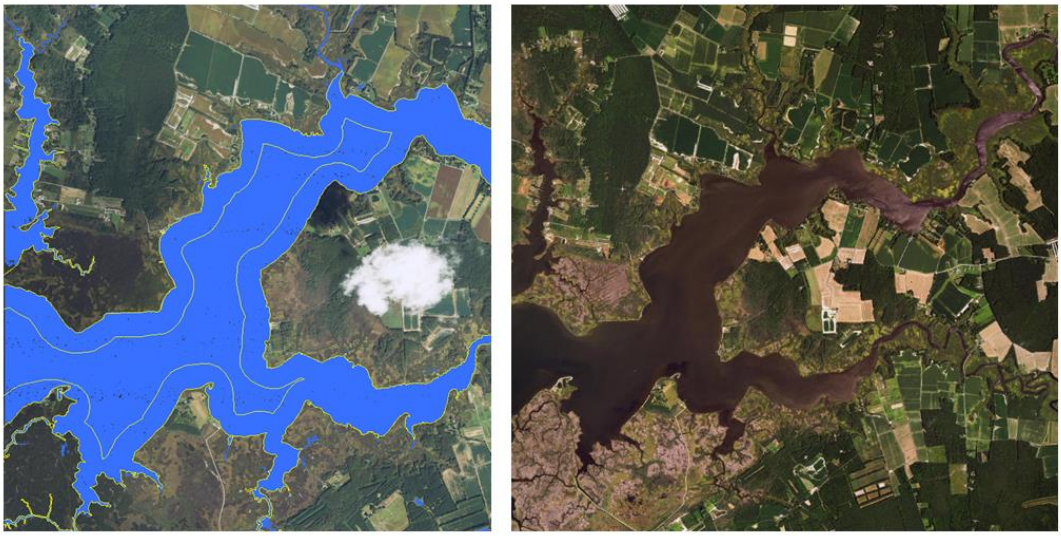

Figure 47: Enlarged area 01 of Manokin dataset.

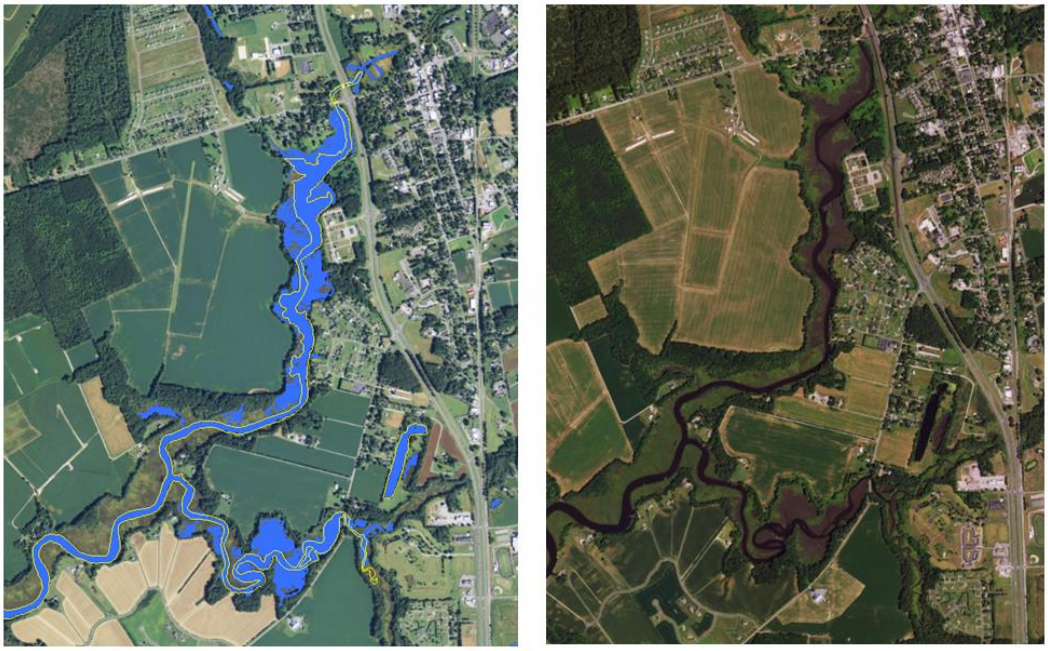

Figure 48: Enlarged area 02 of Manokin dataset.

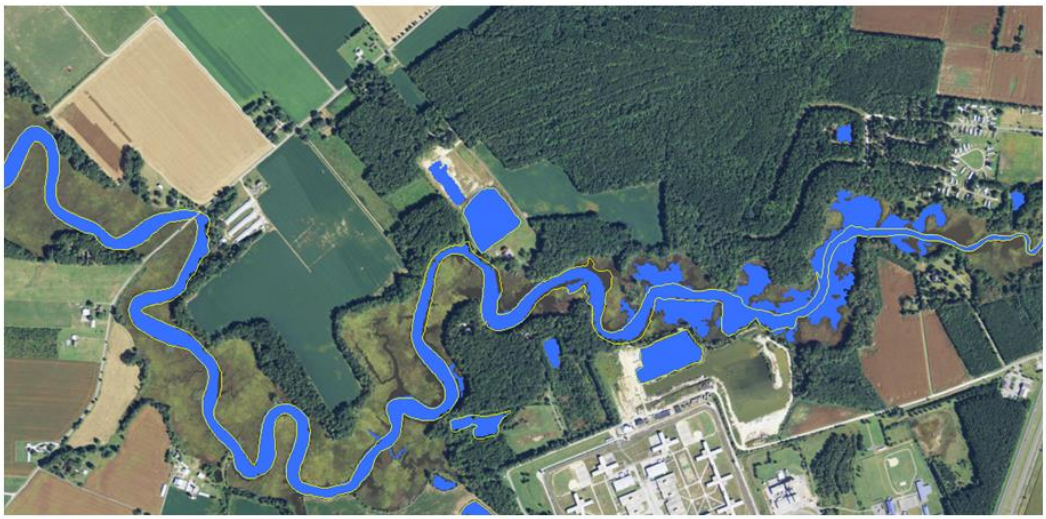

Figure 49: Enlarged area 03 of Manokin dataset.

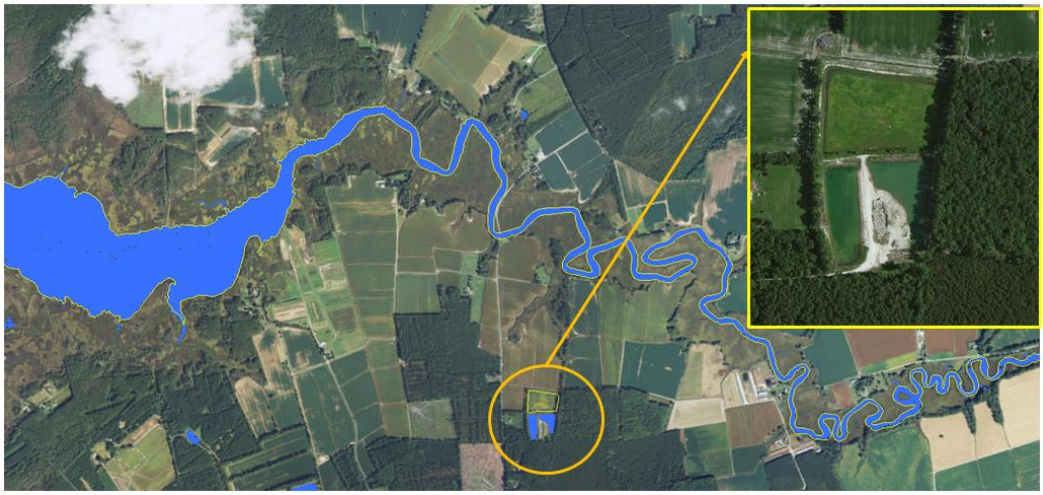

Figure 50: Enlarged area 04 of Manokin dataset.

In Figure 51, automated detections are shown with LAS file boundaries for Ste. Genevieve, Missouri dataset. This project covers around 1500 sq.km area. Automated geolocation based batch creation was used to create batches. Each batch took around 4 LAS files in a single run. The visualization is not clear in a single picture because of the huge size of the project. Therefore, six more zoomed in versions of key areas are shown in subsequent figures. From Figure 52 to Figure 57, it is clear that small size waterbodies, small river channels, and small lakes were correctly detected. It is important to note that this kind of detail mapping of small waterbodies was very tiresome and hugely expensive if performed manually.

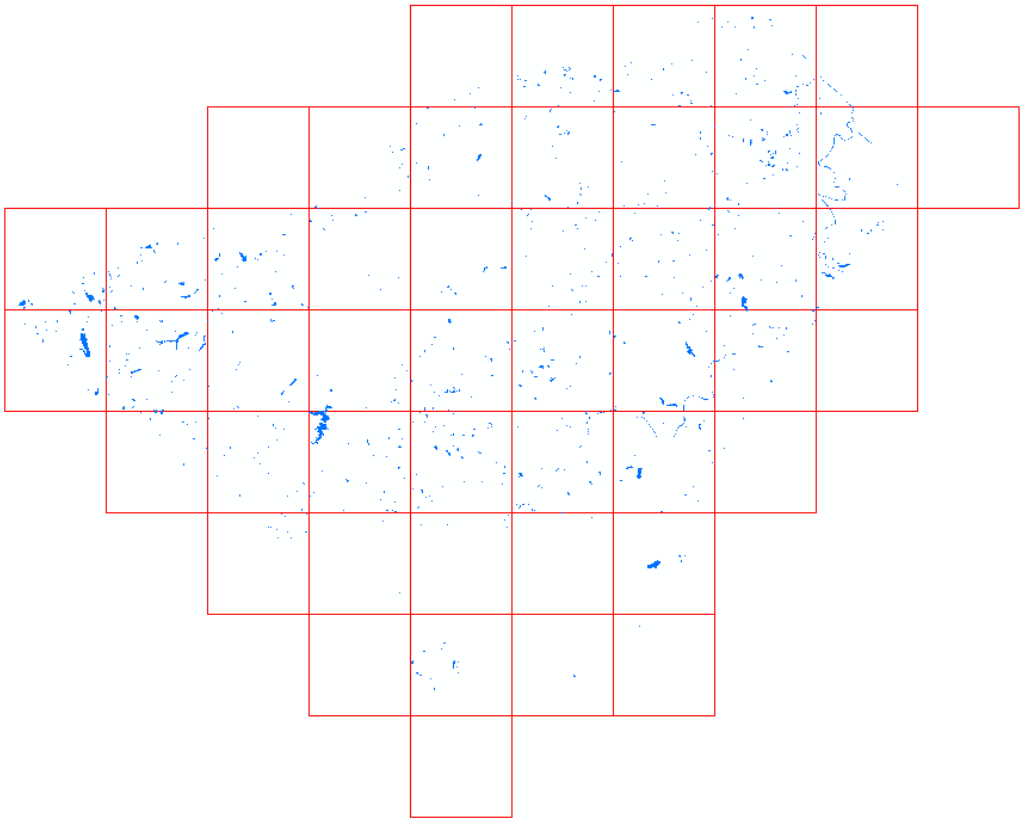

Figure 51: Automated waterbody detection (blue polygon) of Ste. Genevieve dataset with LAS files boundaries

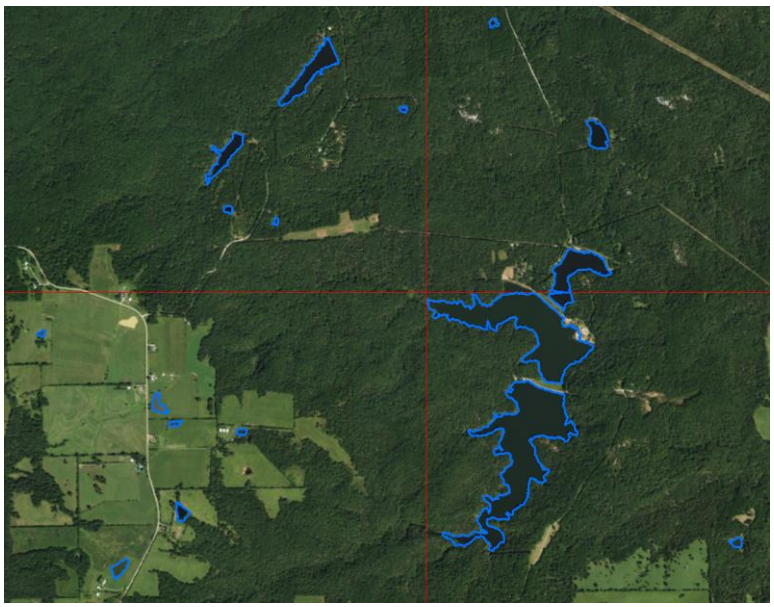

Figure 52: Area 01 enlarged version of Ste. Genevieve dataset

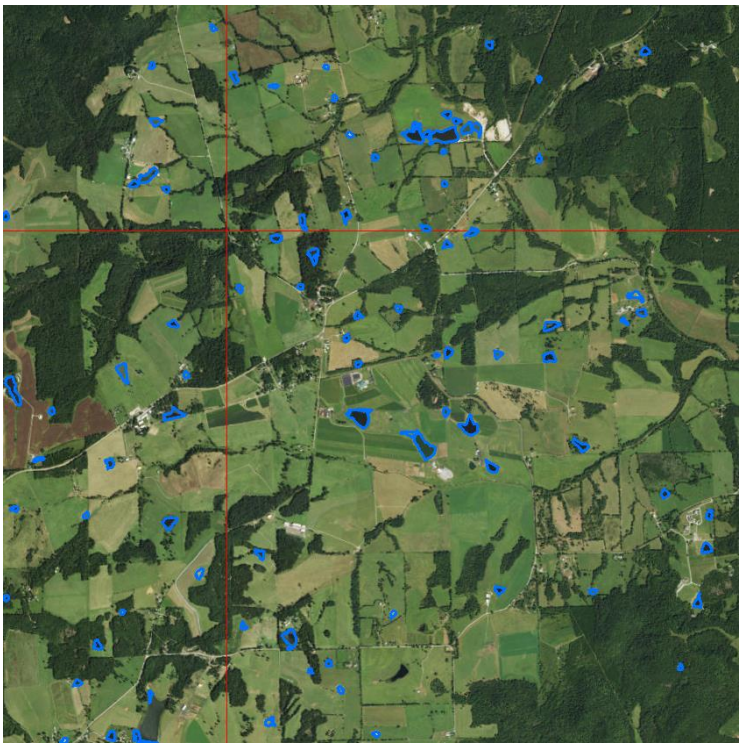

Figure 53: Area 02 enlarged version of Ste. Genevieve dataset

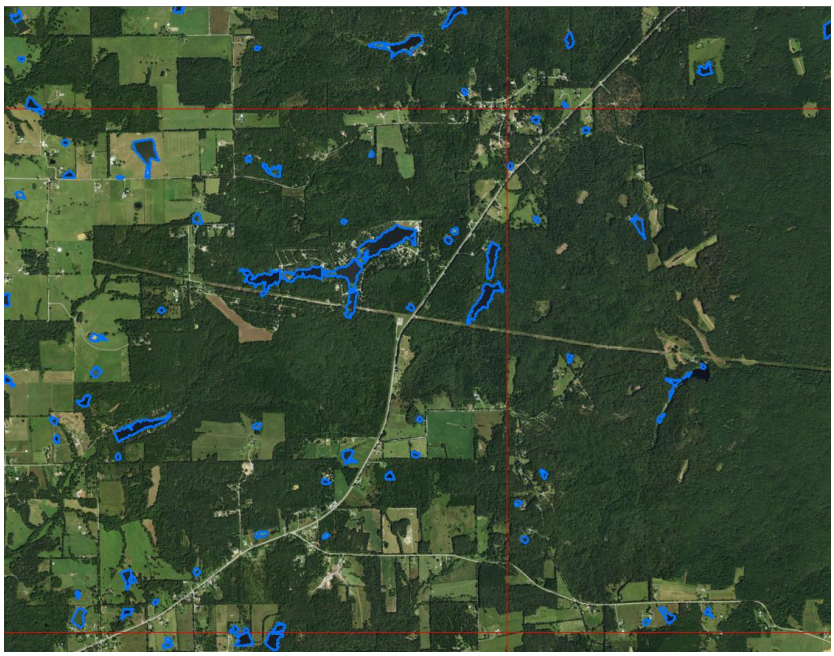

Figure 54: Area 03 enlarged version of Ste. Genevieve dataset

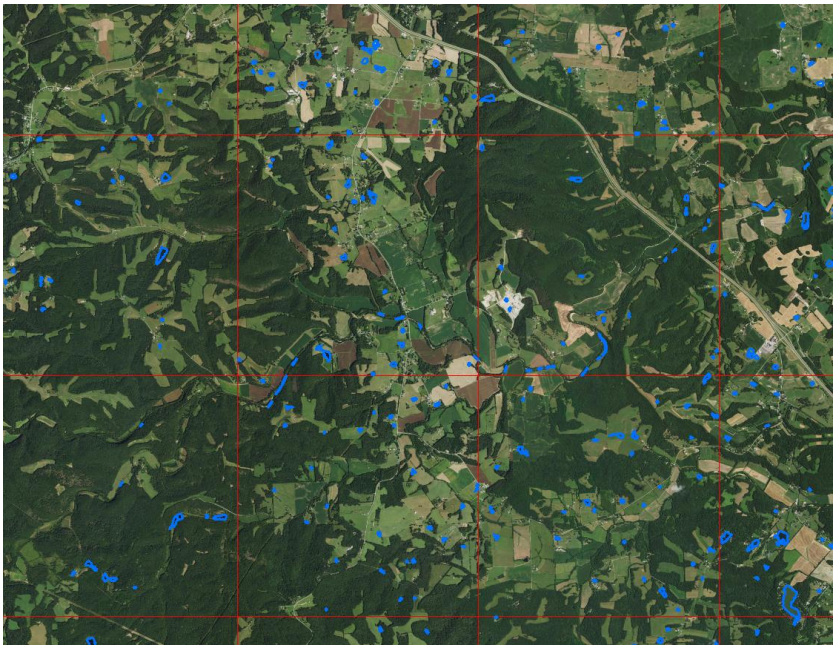

Figure 55: Area 04 enlarged version of Ste. Genevieve dataset

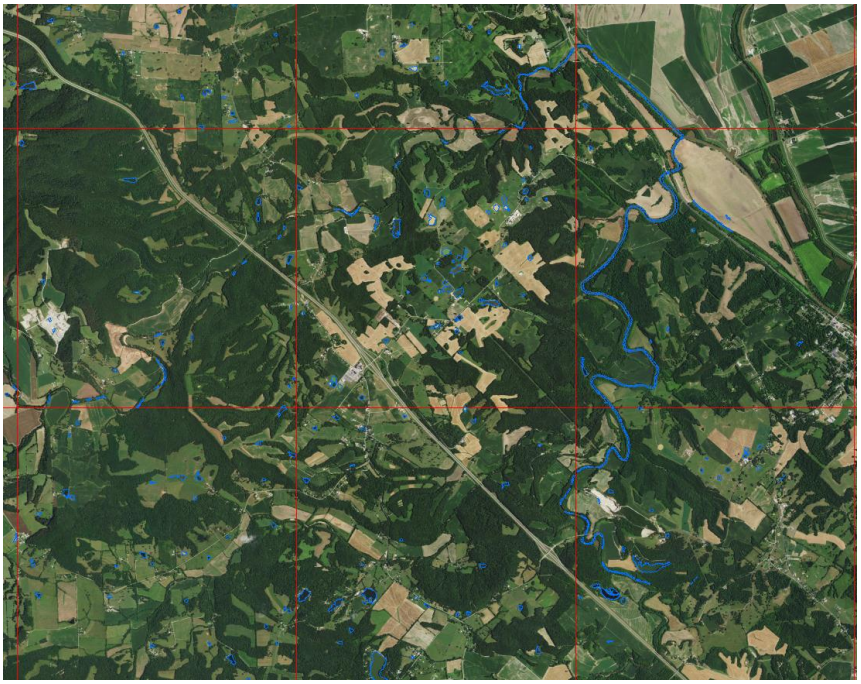

Figure 56: Area 05 enlarged version of Ste. Genevieve dataset

Figure 57: Area 06 enlarged version of Ste. Genevieve dataset

Chapter 3

Estimation of Solar Potential on an Urban Area

3.1 Introduction

Currently, the world is moving towards the green power generation, and solar power is one of the promising ways to do so. Significant power load comes from urban areas, where, theoretically, a large amount of building surface area is also available for solar panel installation and therefore, solar power generation. Generating electricity locally through solar panels rather than transmitting from long distance is beneficiary for the power system. The recent advances in solar power technologies, as well as green-power friendly governments policies, are also encouraging for the individual household to pursue own solar power unit. Therefore, property owners, as well as power generation companies, can be benefited by having a citywide solar potential map, which can provide available estimated annual solar energy at a given location. An efficient and reliable solar potential measurement is a prerequisite for an effective solar energy system development in urban areas. In addition, the calculation and visualization of solar potential on rooftops and building facades could open up a wide variety of options for solar panel installations. However, complex urban scenes make it hard to estimate this potential, partly because of shadows cast by buildings. In fact, the sunlight striking a specific location over a period of time varies according to its position in the globe, the surrounding environment, weather, and few other meteorological factors. All of these factors should be considered to derive a solar potential map.

Airborne LiDAR is a potential technology in this domain [44]. LiDAR-based 3D city models can be a good source data for solar potential mapping. High accuracy, large-scale data collection capabilities, and high data density give LiDAR an edge in this domain. This LiDAR-based 3D city model can distinguish among ground surfaces, building surfaces, and building facades. Although most of the current LiDAR-based local solar potential assessment algorithms

mainly address rooftop potential calculation, whereas building facades can contribute a significant amount of viable surface area for solar panel installation [45]–[47]. The solar analyst from ArcGIS and r.sun from GRASS are two well-known solar power estimation tools using raster based elevation data [48], [49]. These tools are capable of estimating solar power on large areas from digital terrain model. Vertical surfaces were ignored to estimate solar potential in both of these tools. Some work has been reported to estimate solar power in vertical planes using the 2.5D elevation information from LiDAR data. The building facades are assumed to be planar, and discontinuous features such as windows, balconies, and other irregular features are ignored [50], [51]. Despite this assumption, annual solar power per unit facades area is an important factor to consider.

In this Chapter, we present a method to generate city model from LiDAR point cloud, and calculate solar radiation from model. This method uses patches to build the city model. Therefore, the solar potential is also readily available for the façade and rooftops.

From a research point of you this is another feature extraction problem from LiDAR data. To us this is a second such example, after the fairly detailed treatment of water body extractions from airborne LiDAR, which was the main thrust of this dissertation. The water body extraction represents approximately 75% of the work and the feature extractions associated with solar power potential estimation about 25%.

3.2 Proposed method

In this work, we introduce a new algorithm to calculate solar potential of both rooftop and building facades. The workflow of the proposed algorithm is shown in Figure 58. An overview of the method follows.

Initially, a method was developed for the automatic extraction of building surfaces from LiDAR point cloud. This 3D modeling block contained building facades, building surfaces, and

ground surfaces. After the 3D city model was obtained, all surfaces were available for solar radiation estimation. Details of the 3D city modeling will be discussed in a later section.

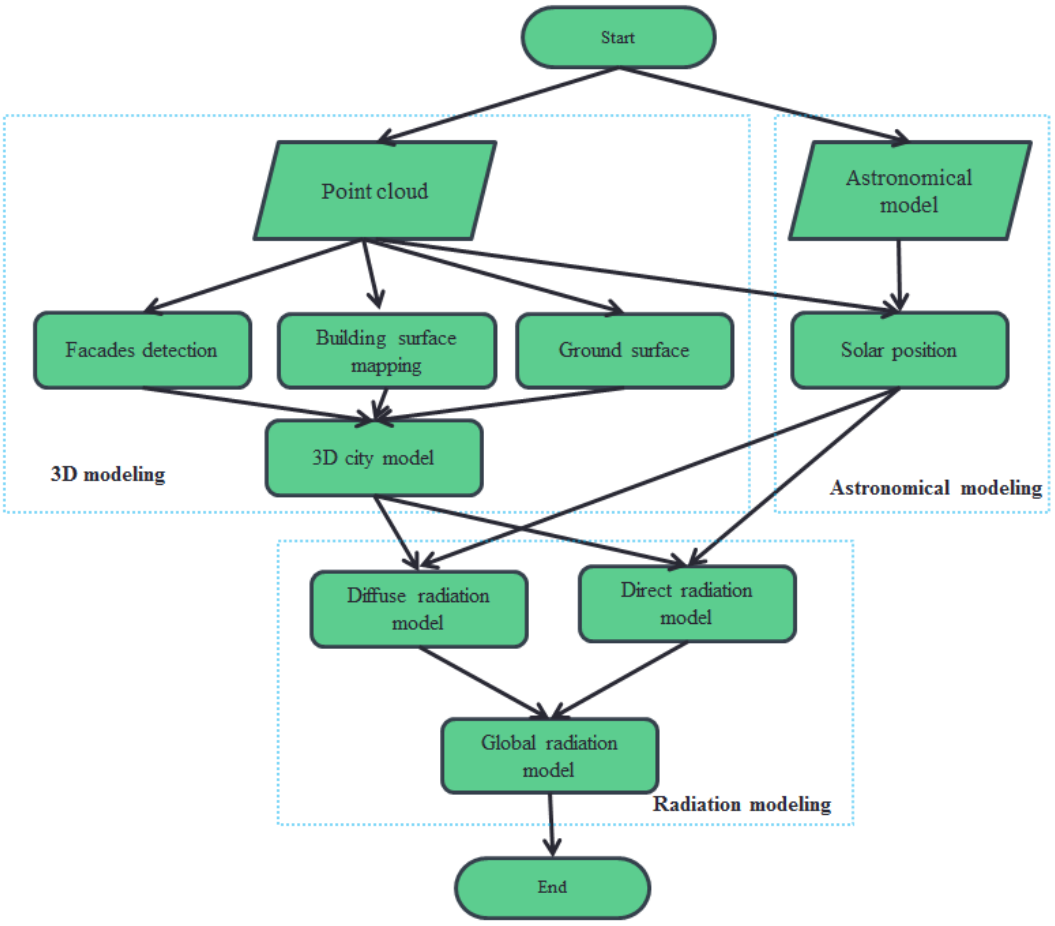

Figure 58: Solar potential mapping workflow

An existing and well known astronomical model was used to calculate the position of the sun at a given location and at a specific time [52]–[54]. This solar position vector contains azimuth and elevation angle of the sun at that specific time and place. We will discuss the astronomical model in detail in a later section.

We developed a new approach to produce a shadow model from the sun position vector and 3D city model. Solar radiation was calculated on each patch of the scene model. For a specific sun location, shadow model was calculated over the scene to estimate the direct radiation. Besides the direct sunlight, surfaces on the scene also receive indirect sunlight from diffuse radiation from the whole sky. Diffuse radiation is directly related to the how much sky is in the line of sight of that surface. The sky view factor was derived for all surfaces of the 3D city model. Both diffuse radiation and direct radiation were considered for the aggregate solar energy estimation [46].

3.2.1 3D modeling

At the very beginning, an algorithm has been proposed to generate a 3D map from LiDAR point cloud. This algorithm extracts building rooftops, facades, and ground surfaces. Workflow of the proposed method is shown in Figure 59. At the very beginning, the physical properties of elevation were explored. It was assumed that rapid elevation change is a key property of non-ground blocks. Angular filtering described earlier in our work related to water body extraction is a useful tool for elevation change detection.

First, LiDAR point cloud was rasterized to obtain a regular image-like gridded raster. In this case, minimum elevation was used to create the elevation raster, and inverse distance weighted interpolation was performed to estimate the elevation of a no LiDAR return pixel.

After this pre-processing phase, minimum elevation raster was used for object-based analysis. For building edges, elevation change is generally drastic. In the case of sharp elevation change, we can consider it as building boundaries. Therefore, the angular filtering was performed on the elevation raster, which provided all potential edges. As mentioned above, angular filtering provides a measure of maximum elevation difference around a pixel's local neighborhood. Additionally, the block, which has a very low elevation difference with all of its neighbors, can be approximated as a point on a planar surface. In the next step, a threshold angle of 45° was applied,

which means a neighbor x meters away can have a maximum of x meters elevation difference, to be considered to be in the same plane.

The find edges block stored all detected building edges in a binary raster. Now, sparse edges at different places were not considered as building edges. The sparse detections were removed by a connected component analysis. The connected component analysis was performed to connect all edge points from a single object as a single entity. Sparse edges, which do not have enough connected components, were ignored. Edges from separated buildings were assumed to be disconnected from each other. Therefore, the connected components provide isolated areas, where a single building or a group of buildings could be found. Thereafter, edges from each object were numbered in the descending order based on the number of connected edge points. If at least *n* points were not connected to represent an object, it was excluded.

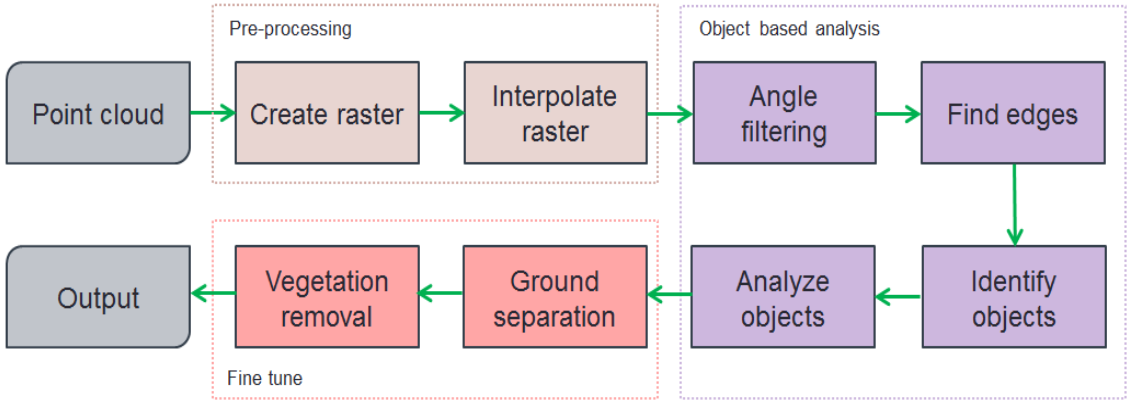

Figure 59: 3D model generation steps.

Now, the borders of all objects are well defined and need extensive and separate analysis. All isolated objects were analyzed in the analyze-objects block.

A convex-hull was prepared for each object from its connected edges. Therefore, we get isolated convex hulls were mapped over the scene, and each isolated convex hull had a building or a group of buildings. The largest planar surfaces excluding all of these convex hulls were potential

ground points. Therefore, elevation could be estimated inside those convex hulls from surrounding ground pixels.

An elevation model was estimated for each convex hull using all of the potential ground points over the whole scene. This interpolation was performed using same the IDW method described in the previous chapter. This estimation provides an approximate idea of the elevation beneath the object enclosed by a convex hull. Now, the actual elevation and the estimated elevation can be compared. If the absolute difference between the actual elevation and the estimated elevation was larger than a threshold, then that pixel was considered as non-ground. By using this criterion, all planar surfaces were classified from each object.

Surfaces were numbered according to the area of the surface. Planar surfaces of each object were helpful to build a 3–D model of the object. At the end of this stage, edge points association was resolved and assigned to a particular surface according to their elevation. An edge point was assigned to its neighboring surface, which has the lowest elevation difference with that point's actual elevation. The numbers of returns were used to check all vegetation. If the lowest elevation of a vegetation block has elevation difference below the threshold with its neighbor, then it represents the bare earth elevation. If not, the elevation was estimated by the inverted weighted difference method.

Outputs from different steps are shown in following figures. In Figure 60, all building edges detected by angular filtering are shown. All edge pixels were detected by the angular filtering and then followed by the connected component analysis. Connected component analysis connects all pixels from a building into a single object. All objects are shown in different colors in Figure 60.

A convex hull is drawn around each object to find the ground and non-ground points. All convex hulls are shown in Figure 61. Each convex hull contains an isolated objects or a group of connected objects.

Areas outside the convex hulls are valid ground pixels, which were used to find the DEM of the whole area. All valid ground pixels are shown in Figure 62. The calculated DEM provides the estimated ground level inside the convex hulls. Elevations of pixels far from the estimated ground level were considered as non-ground objects.

All building surfaces can be located by comparing the actual elevation with the estimated DEM. All building surfaces are shown in Figure 63. Clearly, a single building can have multiple surfaces at different height level. Surfaces with different height level were detected separately in this method.

Finally, previously detected edges were assigned to a nearby building surface based on the elevation. Edge pixels were assigned to that neighbor surface, which has smaller elevation difference with the edge elevation. Final outputs of all the detected building surfaces are shown in Figure 64.

At the end, vegetation areas were detected from the multiple return criteria, described in the previous chapter. Detected building surfaces, vegetation regions, and ground surfaces are shown in Figure 65. The detected building surfaces, building edges and ground surfaces were provided as inputs for radiation modeling. We have published the work in an ASPRS annual conference [42].

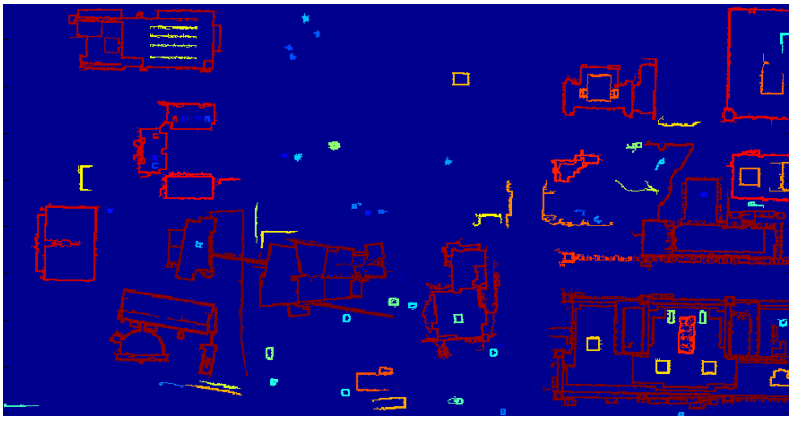

Figure 60: Angular filter followed by connected component analysis provides building edges.

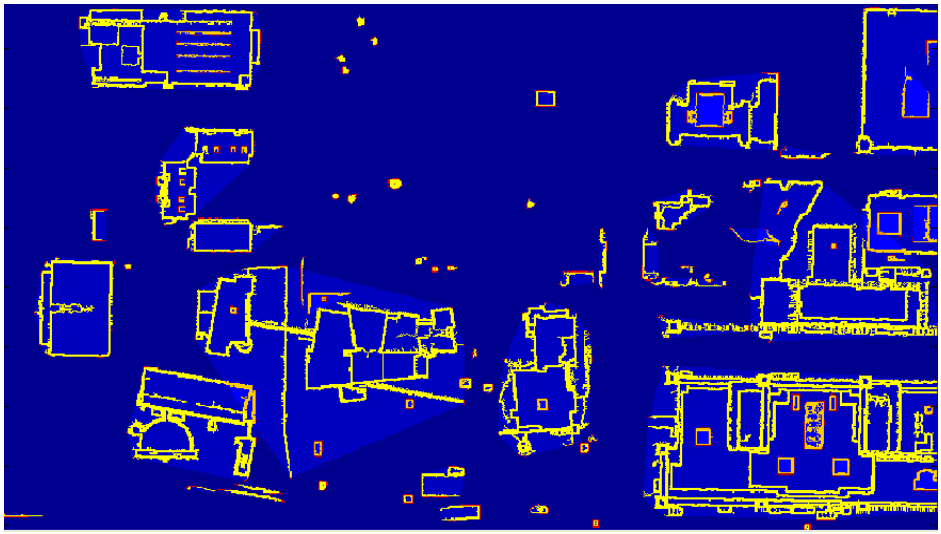

Figure 61: Convex-hulls for all objects to detect building surfaces

Figure 62: Available ground pixel outside of convex DEM generation.

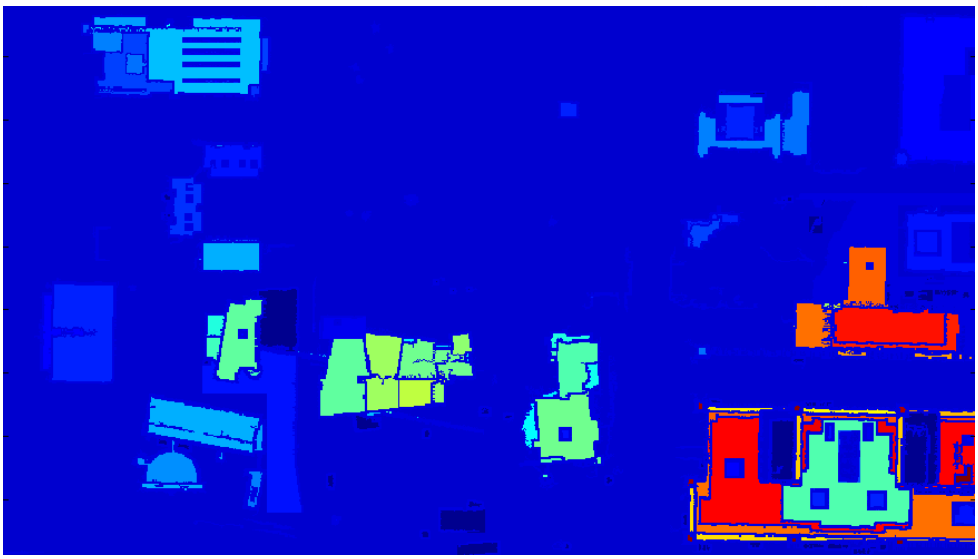

Figure 63: Detected plan building surfaces from convex hulls.

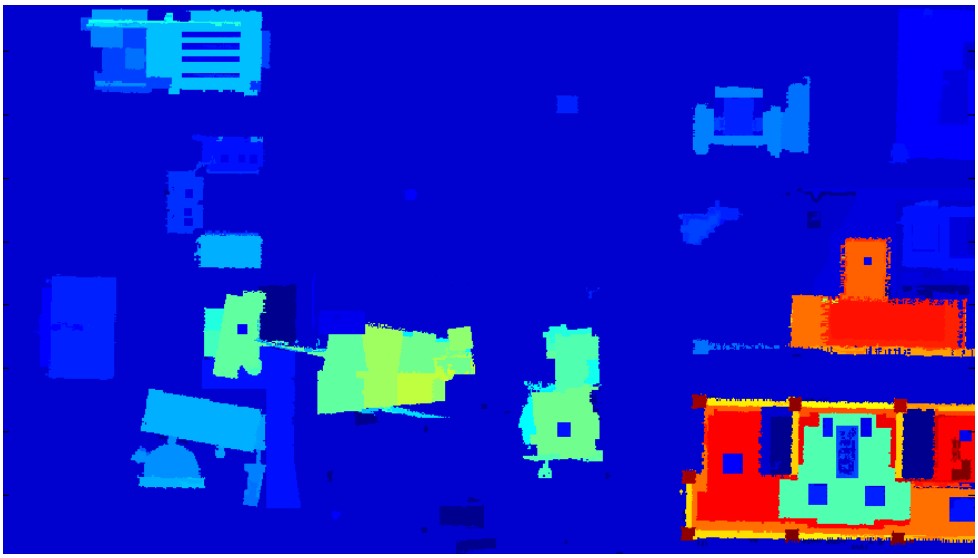

Figure 64: Final building surfaces form the 3D model.

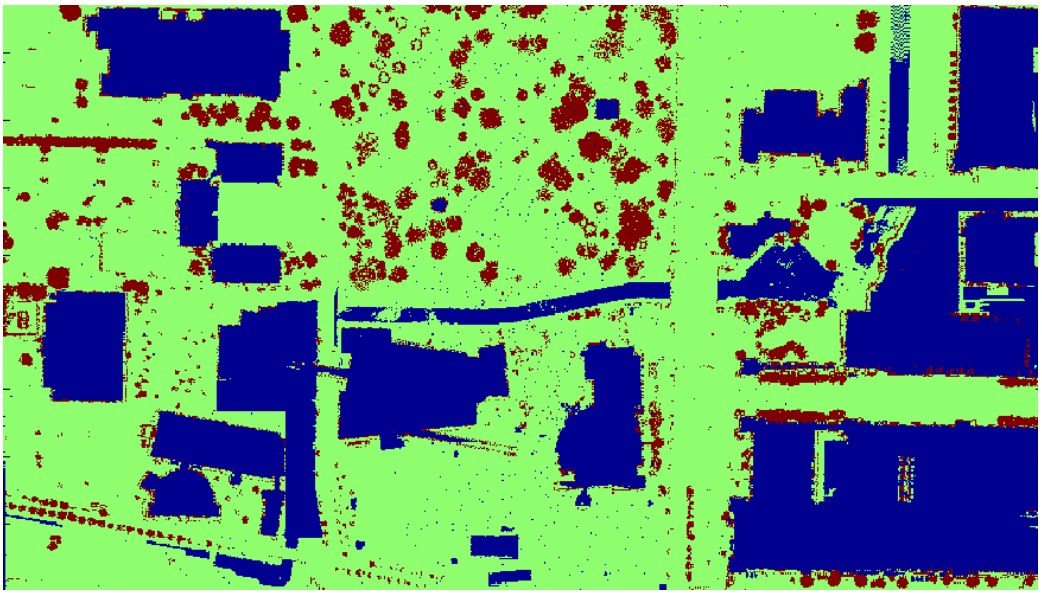

Figure 65: Ground pixels (green), Man-made objects (blue), and vegetation (brown) are shown.

As discussed earlier, LiDAR data is 2.5D and does not provide a 360-degree point cloud. Therefore, a plane should be estimated to represent the facade of the building. Additionally, some part of the façade may have sunlight but other parts may not. Therefore, the façade of the building has to divide into many parts to deal with partial shadow situation.

To tackle this scenario, we developed a patch-based representation. The patch-based representation is shown in Figure 66. The blue and the red bubbles are LiDAR returns from ground and buildings. The edge of the rooftop and the ground was connected by a vertical patch, which is shown in green. The vertical patch was divided into eight more sub-patches. Therefore, solar power can be estimated in eight different levels on that vertical patch.

Rasterized LiDAR data is a 2.5D based representation, where all points were evenly placed in XY-plane and no two point share the same x and y values. Here, x, y, and z correspond latitude, longitude, and altitude respectively. However, to represent vertical walls, multiple points

have to share same x and y values with different z values. The 2.5D representation is suitable if only rooftops were considered for solar power estimation.

We desire to consider facades of the building for solar potential calculations. Therefore, a new representation beyond 2.5D needs to be introduced to bring building facades in the scene. For this purpose, patches were used to represent each plane in the scene. Four adjacent points, e.g. (n, m) (n+1, m) (n+1, m+1) (n, m+1), are grouped together to make a patch. The order of the points was set in the counter-clockwise direction; therefore, the normal of each patch was in the upward direction. Azimuth and elevation angles of the normal vector and the area of the patch were also calculated for next steps. Identification of facades is available from the building modeling data. Facades were equally divided into eight small patches. Therefore, solar intensity can be calculated separately for each of those small segments. In Figure 67, patches are shown in a different colors based on the altitude of the patch. We can see that facades are made up of multiple patches in multiple elevation levels. For better visualization, only transparent patches are shown in Figure 68. It is clear that the patches form a net-like structure around the buildings. This patch-based representation enables us to represent solar potential on the building facades.

Once the 3D models are converted to patches where needed, the 3D modelling step is complete and we are ready to continue with the remaining steps needed for the solar potential calculations.

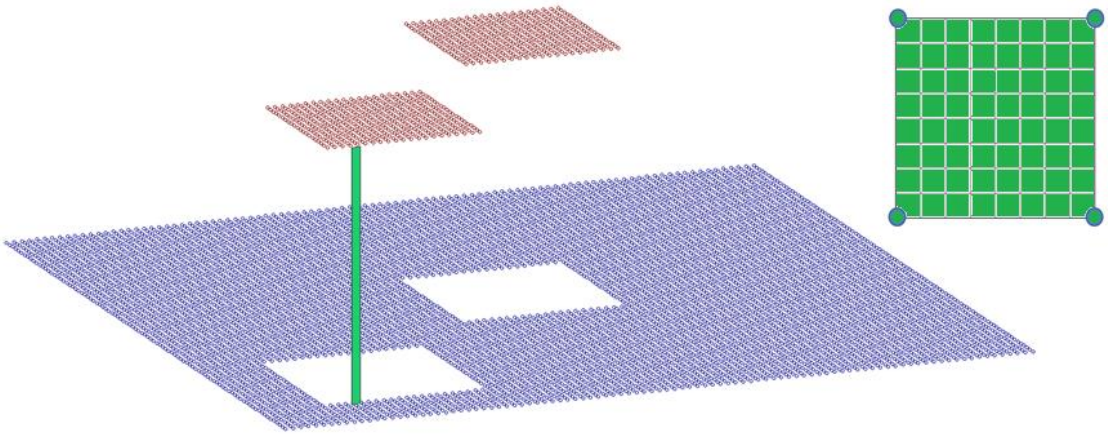

Figure 66: Patch-based representation of 2.5D LiDAR data.

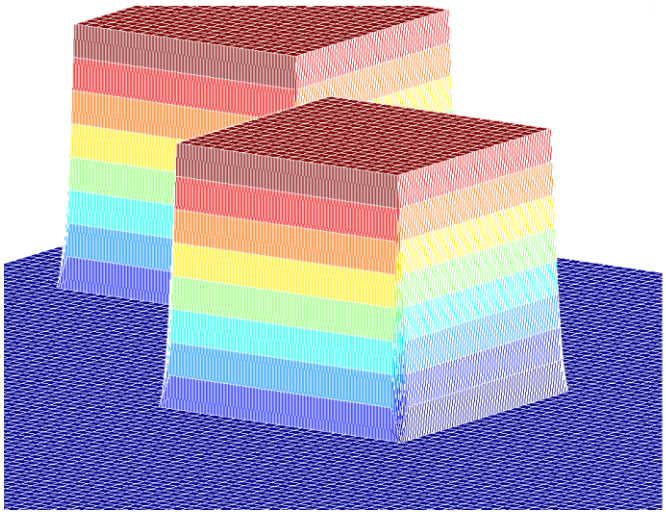

Figure 67: Patches are color-coded by their elevation in the test dataset. Ground, rooftops and different height of the facades are shown in different colors.

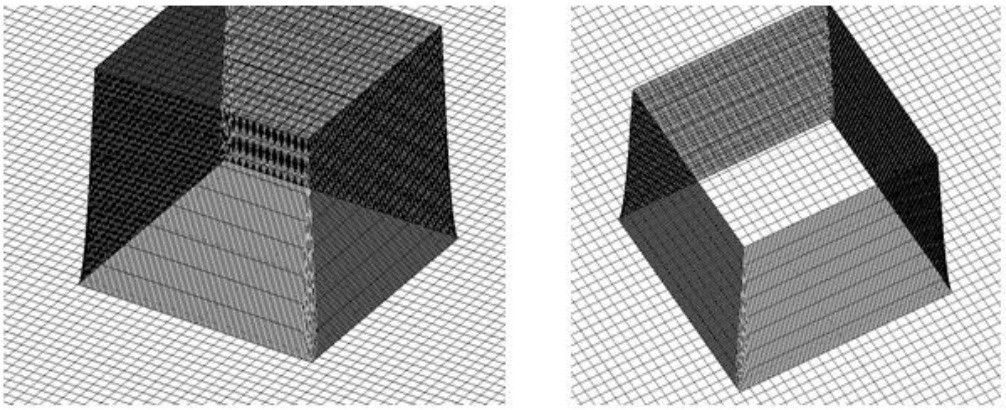

Figure 68: Patches are shown without vertices and color coding. Form like a net around the structure.

3.2.2 Astronomical modeling

The position of the sun for a specific time and place has to be known to map shadow over the scene. A well-known astronomical model was used to calculate the position of the sun at any given time [54], [55]. The position of the sun is represented with two values the elevation angle α and the azimuth angle γ. The azimuth angle is the angle of the sun position from direct north. Therefore, if the sun is in the north-east corner, its azimuth angle is 45 degrees. On the other hand, the elevation angle represents the sun position from the zenith. During solar noon, with the sun directly above the head, the elevation angle is zero, and during sunrise or sunset, elevation angle is 90 degrees. As discussed earlier, because of the parallelism property of sunbeam, the calculated elevation and azimuth of the sun was considered a constant for the whole area, when the study area is small. The elevation angle α and the azimuth angle β were calculated as follows,

$$\alpha = \sin^{-1}(sin\delta\ sin\varphi + cos\delta\ cos\varphi \cos(H))$$

$$\gamma = \cos^{-1}\left(\frac{sin\delta \cos\varphi - cos\delta\ sin\varphi \cos(H)}{cos\alpha}\right)$$

where, φ is the local latitude, H and δ are the sun hour angle and the sun declination angle respectively at the time and day of interest. The declination angle and the hour angle were calculated in a few steps described below.

First, local standard time meridian (LSTM) was calculated by multiplying the time difference between the local time and the GMT time with 15 degrees, $LSTM = 15° \times \Delta T$. The equation of time (EoT) was calculated from the day number (d) using the following equation,

$$EoT = 9.87\ sin(2B) - 7.53\ cos(B) - 1.5\ sin(B)$$

$$where, B = 360/365(d - 81)$$

The declination angle δ was calculated as $\delta = 23.45° sin B$. Thereafter, time correction factor (TC), local solar time (LST), and hour angle H were calculated as follows,

$$TC = 4(Longitude - LSTM) + EoT$$

$$LST = local\ time + TC/60$$

$$H = 15°\ (LST - 12)$$

The positions of the sun at the different times of the year were calculated using the above astronomical model. In Figure 69, the sun positions are shown in the sky for the first day of each month in the DFW area. From the figure, we see that the sun goes from east to west, and during the winter sun is inclined towards the south, and travels through near the zenith in the middle of the year.

After calculating the azimuth and elevation angle of the sun, the direct sun intensity on any patch can be calculated. The elevation and azimuth angle of the sun position and the normal vector of the patch can be used to calculate the solar intensity on that given patch. The solar vector (red and blue) and the normal of a triangular patch (yellow) are shown as examples in Figure 70 and Figure 71. When the sunbeam heats the patch perpendicularly, the patch receives the maximum amount of solar energy. This amount varies from the day, the month, and the year. We discuss this further in the typical meteorological year data section. The amount of solar energy

varies based on the patch orientation; we'll establish the relationship in the direct radiation model section.

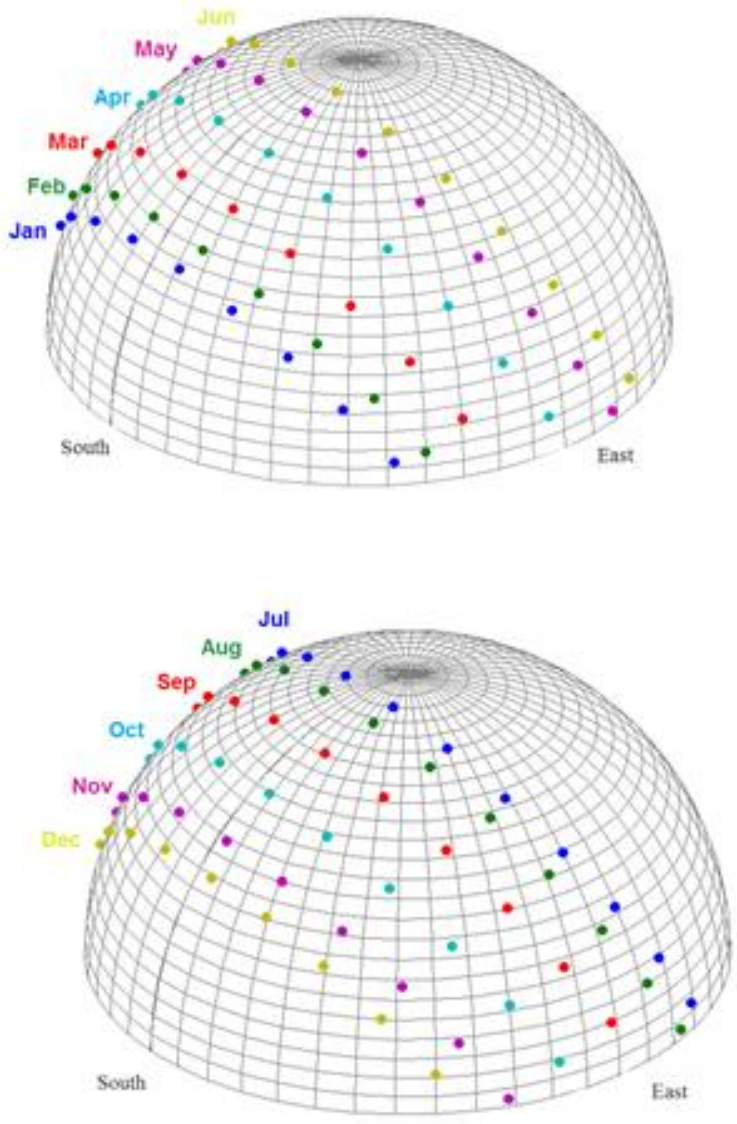

Figure 69: Position of the sun on the sky at a different day of the year in DFW area. The first day of January to June (Top), the first day of July to December (Bottom).

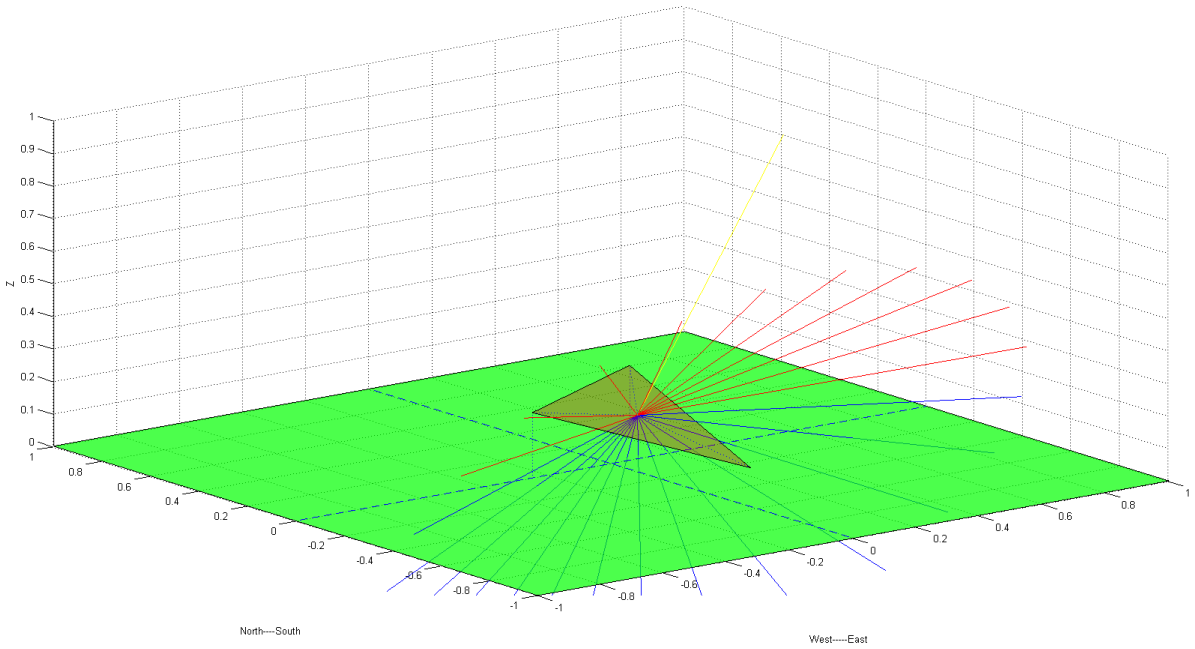

Figure 70: Surface normal angle with the sun vector

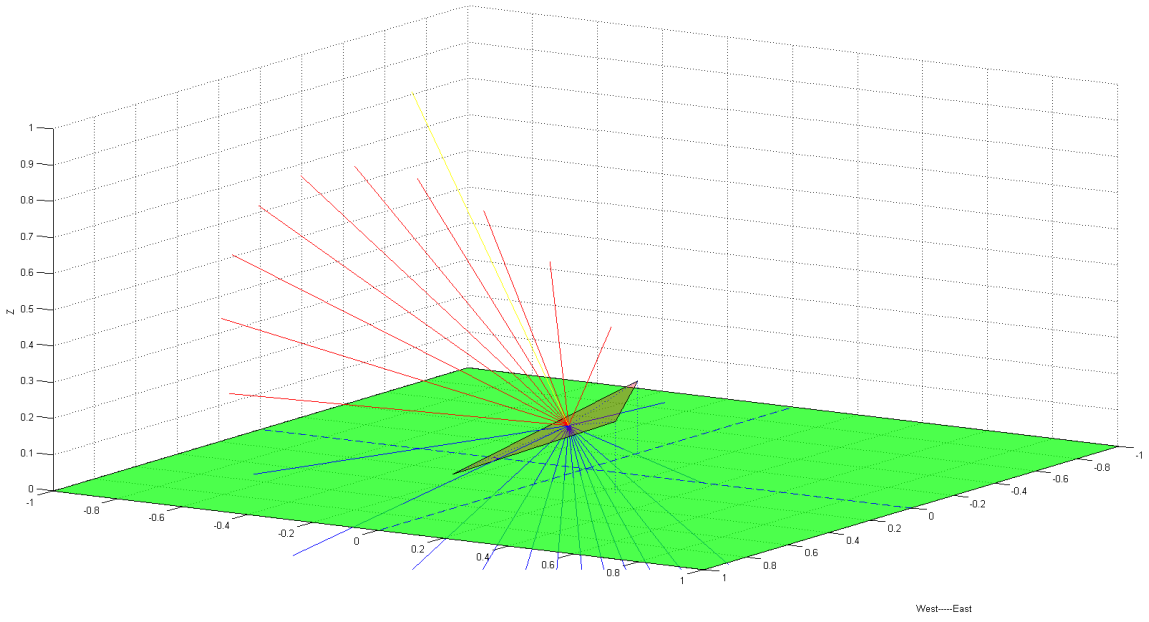

Figure 71: Surface normal angle with the sun vector form different angle

3.2.3 Typical meteorological year (TMY) data

The intensity of the sun on earth surface varies over time based on the sun distance, and the amount of atmospheric distance sunlight travels before reaching the earth surface. The National Solar Radiation Data Base (NSRDB) archives have directly recorded TMY data for different location over the whole United States. In the TMY dataset, Direct Normal Irradiance (DNI) and Diffuse Horizontal Irradiance (DHI) are available for every hour of a year. DNI is the amount of solar radiation received by a surface per unit area, which is held perpendicular to the incoming sunlight. On the other hand, DHI is the amount of diffuse radiation received by a horizontal surface per unit area, which is due to the scattered-sunlight, comes equally from all directions over the whole sky. The TMY datasets are available for public download at National Renewable Energy Laboratory (NREL) website [56].

The amount of solar radiation received by a flat plane per square meter is shown in Figure 72. This hourly data come from the DFW area for the month of January and July. Received solar energy is zero before sunrise and after sunset. In July, the day hour is longer than the day hour in January. Additionally, in July, the amount of solar energy received at any time is higher than the solar energy received in January at the same time of the day. This figure shows the typical summer and winter solar energy on a flat plane over the whole day.

Based on the DNI and DHI data, sun intensity can be estimated for any patches by considering patch orientation, the shadow on patches, and percentage of sky visibility from that patch. The steps of global irradiance calculation will be described in the next two sections.

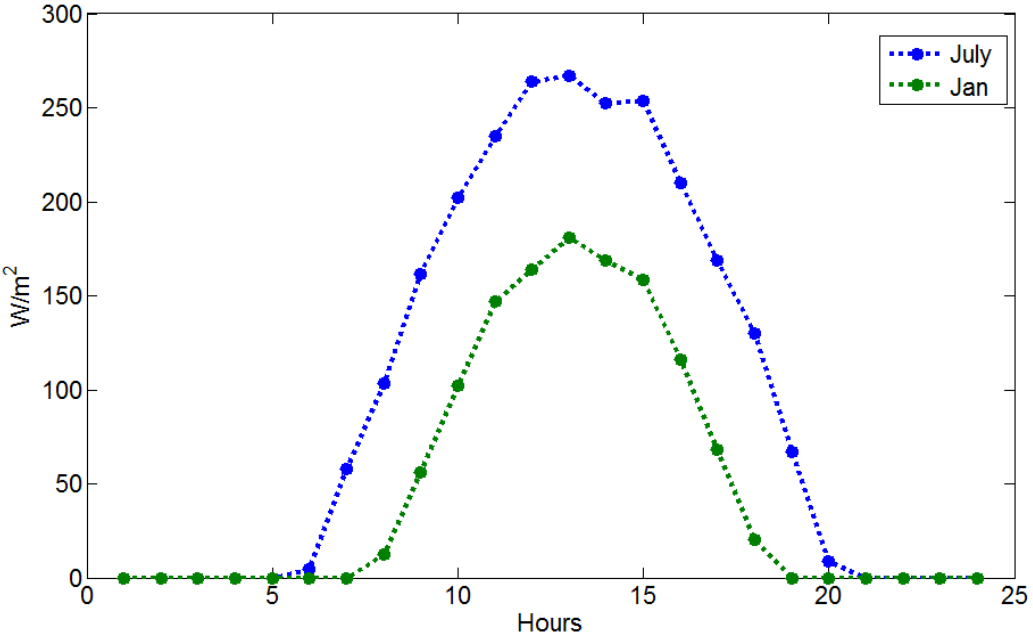

Figure 72: Amount of solar radiation received by a flat plane per square meter is shown for two months at the different time of the day.

3.2.4 Direct radiation model

Direct sunlight is one form of solar energy, which can be estimated by the direct radiation model. A patch may get the direct sunlight or may be blocked by the shadow at a specific time of the day. Therefore, at first, a shadow map was generated for a given sun position over the whole scene.

Sunray lines are parallel to each other, and an object in one ray line can only cast a shadow in the same ray line, not in any others. Therefore, it'll be computationally less expensive if the ray lines are calculated beforehand. To compute the ray lines, it should be noted that, the scene is rasterized. Therefore, any position on the scene is defined by two integers (x, y). For a given azimuth angle sunrays first hit the border pixels. For example, for a zero degree azimuth, sunrays hit the top/north pixels first. Then the subsequent ray lines were drawn from the first hit pixels.

After a certain distance, the ray may split into two adjacent pixels, because the calculated x and y were non-integer values. If any ceiling, round, or floor operation are used then few pixels may be left out without assigning any ray line, which is not the case in reality. To avoid this situation, the ray lines were assigned to both pixels. If any pixels had been already designated with a ray line value then no new values were considered. For better visualization and implementation efficiency, the starting point of each ray lines started with an integer value, and subsequent pixels were assigned with incremental value in decimal place.

In Figure 73, ray lines are shown for two different solar azimuths—0 degrees, and 330 degrees. In the left side of In Figure 73, ray lines are shown for the zero degrees azimuth; different colors represent separate ray lines. It is clear that all sunrays were coming from north to south direction. In the right side of the In Figure 73, sunrays are coming from 330 degrees azimuth. As this ray lines are drawn over a digitized raster, the quantization effect can be seen in the right side figure. Now, after calculating ray lines, we are ready to find out shadows from the LiDAR elevation data. Any object located on a ray line can only cast a shadow in the latter part of that ray line.

Each ray line had a starting point at the scene—the first patch of the scene a sunray encounter. Using trigonometry, the minimum height required to avoid the shadow by the next patch in the same ray line was calculated. This is called as the shadow limit for that patch. If all points of a patch were below the shadow limit, it was considered a patch under the shadow; in that case, same shadow limit was used to calculate the shadow limit of the next patch in that ray line. On the other hand, the patch was considered as lighted if any point of the patch was above the shadow limit. In that case, the height of the current patch was used as the new shadow limit, and the shadow limit for the next patch was calculated accordingly. This shadow calculation method is fast because it handles each patch only once. Facade patches were segmented in multiple patches

as we described in patch building section. Only patches, which were below the shadow limit, were considered shaded patches, patches above the shadow limit were considered as the lighted patches.

A simplified version of the shadow calculation process is shown in Figure 74. The dotted yellow line represents the reference sunbeam. Therefore, all sunbeams are parallel to that reference beam. The first object is the leftmost green stem. The shadow limit was defined by the solid sunbeam just touch the top of the green stem. The next stem was under the shadow limit, therefore, it became a shade stem. Now, the third stem from the right had a portion above the shadow limit and a portion below the shadow limit. Therefore, the portion of that stem below the shadow limit was considered under the shadow. Now, new shadow limit was drawn from the top of the stem. This process was continued to the end of the ray line.

The shadow limit and shadow are shown on a simulated dataset in Figure 75. Two adjacent building of 20 meters height were in this simulated scene, and a $2m \times 2m$ grid was used for rasterization. The left subplots show the shadow limit and the right subplots show the shadows for two different elevation angles. For a 15 degree elevation angle, shadow limits are stretched to the end of the figure and cast taller shadows. The sides of one building are also get shadow from another building. For a 60 degree elevation angle, the shadows are smaller, and limited to just near the building base.

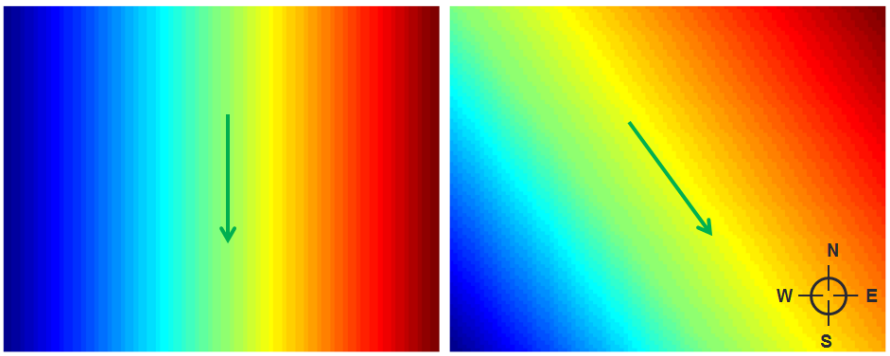

Figure 73: Ray lines are shown in different colors, the same color means same ray lines. Ray lines map for solar azimuth zero(Left) and solar azimuth 150(right).

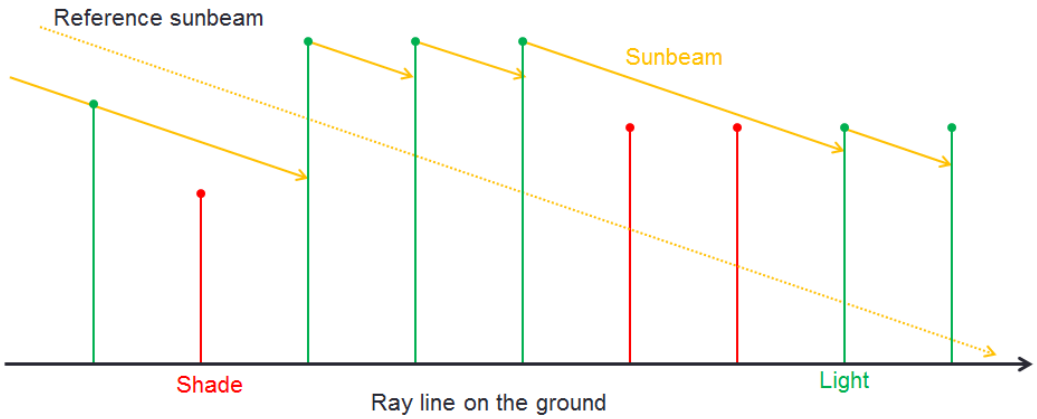

Figure 74: Shadow calculation along a single ray line.

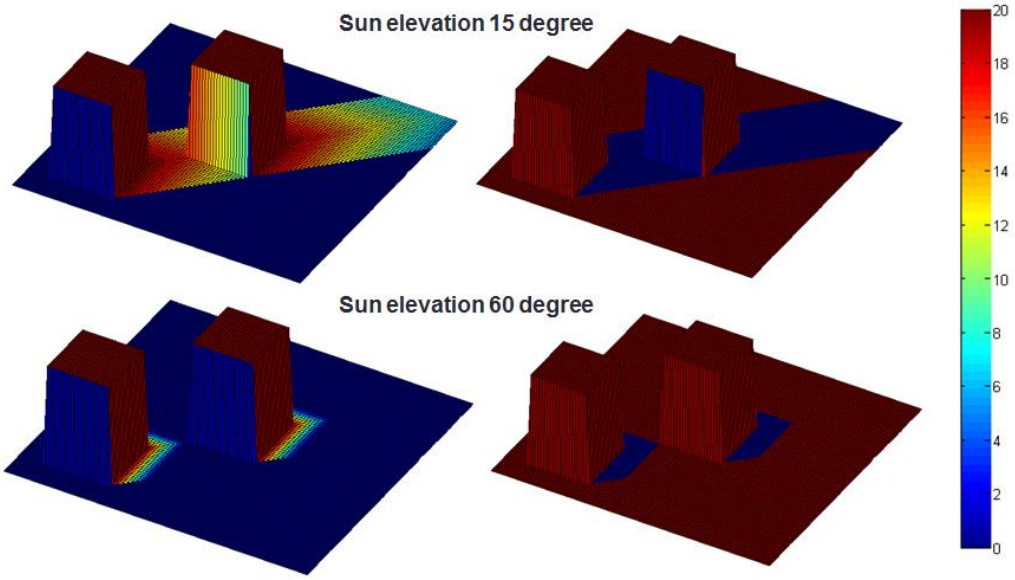

Figure 75: Shadow limits (left) and shadows (right) for two different elevation angles.

After calculating the potential shadow location, solar intensity on each patch without shadow was calculated. The received solar intensity of a patch can be determined by the angle between the sunray and the normal of the patch. The patch will get the highest direct solar intensity if the sunray is perpendicular to the patch, which is the DNI value of that hour from the TMY data. The received intensity is calculated as follows,

$$B_s = DNI\left(\cos\alpha \sin\gamma \cos(\beta - \beta_{patch}) - \sin\alpha \cos\gamma\right)$$

where, α and β are the sun elevation and azimuth angles, γ and β_{patch} are patch tilt and azimuth angles. B_s is the solar intensity on the surface. Patch tilt is zero for flat planes, and 90 degrees for vertical planes.

In Figure 76, received intensity by patches is shown for four different angles, where the solar intensity on the flat planes was set as 1. From the figure, we see that for lower elevation angles, when the sun was on the horizon, facades facing that direction received more sunlight than flat patches. On the other hand, for higher elevation angles—the sun close to the zenith, flat patches received more sunlight than facades. The shadow on facades of the building is also clear from the figure. The shadow is pixelated because a $2m \times 2m$ grid was used. Smaller grid size can be used for smother representation.

3.2.5 Diffuse radiation model

Additional to direct sunlight, another form of solar power reception by any surface is the diffuse radiation. This is the amount of solar power received by the surface from all over the sky. Sun beam scatters in the atmosphere, and the whole sky hemisphere works as light sources for the ground. Diffuse Horizontal Irradiance (DHI) in TMY data is the amount of diffuse radiation per unit area received by a flat surface if the whole sky can be seen from that surface. Many patches did not have the full view of the sky because of occlusion and orientation. For example, a vertical plane only sees half of the hemisphere. The amount of diffuse radiation received by a plane is

proportional to the visibility of the sky, because for diffuse radiation the whole sky becomes a light source. Therefore, in this diffuse model, the whole sky was segmented in 1081 regions, from 15 to 90-degree elevation and 0 to 360-degree azimuth using 5 degrees increment. All the segments are shown in Figure 77 on the half hemisphere.

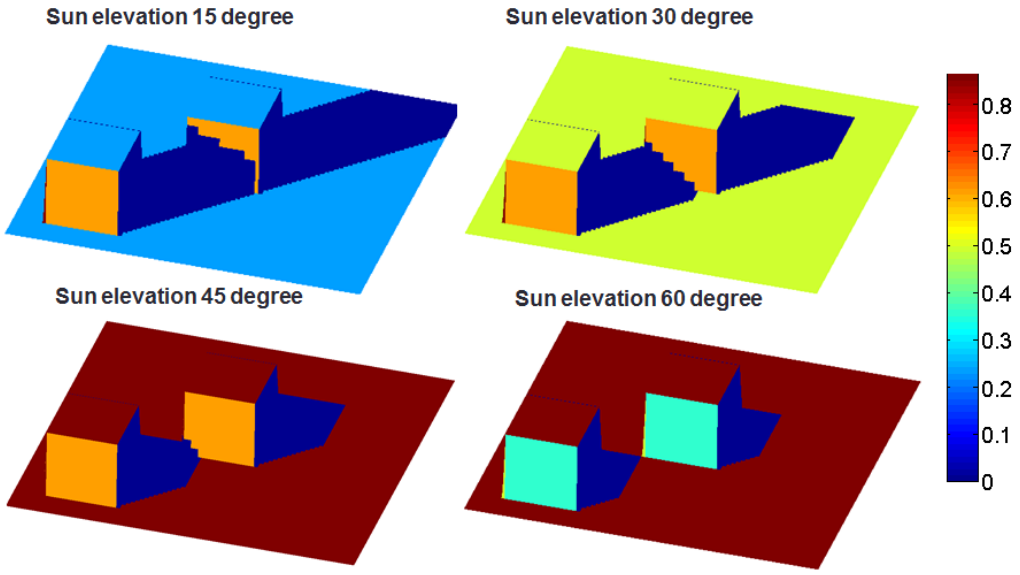

Figure 76: Direct solar radiation on the simulated scene for solar intensity one on the flat plane.

The previous shadow mapping algorithm was used for these 1081 separate sources to find shadows. The shadow map is a binary map representing shadow and no shadow region. Therefore, any patch can see a maximum of 1081 sources, if it is not occluded by any objects and stay flat. Sky view factor was calculated by averaging these 1081 shadow maps, where one indicates that the patch gets the full view of the sky and zero means the whole sky is out of sight from that patch. The sky view factor for the test data is shown in Figure 78. From Figure 78, from which, we see that all of the facades have a sky view factor not more than half. The vertical facades miss the half of the hemisphere because of their orientations, and if there is any adjacent building, the occlusion reduced the sky view factor less than the half. Additionally, sky view factor increased gradually far from the buildings and became one for flat surfaces without any occlusion.

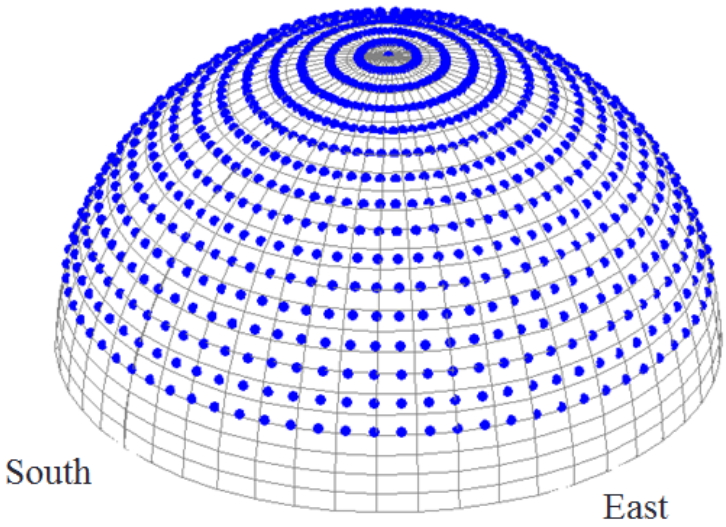

Figure 77: To calculate sky view factor, 1081 sources were considered all over the sky. All sources are shown in sky half hemisphere.

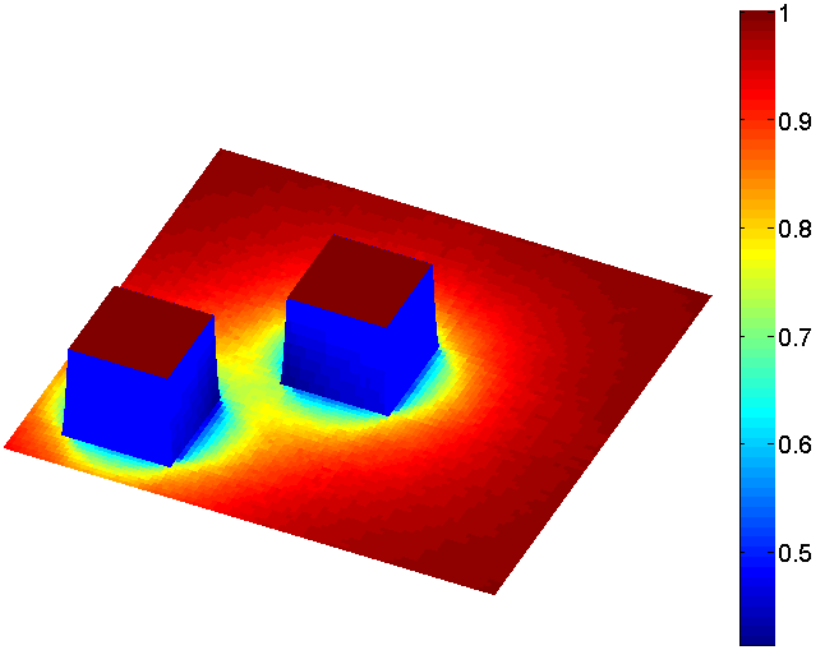

Figure 78: Sky view factor on the test dataset. Facades have sky view factor below 0.5 because half of the sky was out of sight for vertical facades.

3.3 Results

The proposed method was applied on an 187 square meters urban area in the DFW region. A 2m-by-2m elevation raster was created from the corresponding LiDAR point cloud, where the data density was 4points/m2. The solar potential was estimated in one-hour intervals for the whole year. In this section, we draw some conclusions from this estimation results.

In Figure 78, the average daily solar potentials are shown for four different months of the year. April, July, September, and December were chosen to cover different sessions of the year. The solar potential is shown in $\frac{Wh}{m^2 \times day}$. In the middle of the year, the summer season in Texas, the solar potential was the highest in the whole area, around 7kWh/m^2/day. In April, solar potential on roof-top and flat surfaces were also high, around 5.5kWh/m^2/day. The solar potential was gradually increased from the beginning to the middle of the year and then gradually decreased at end of the year. In September, it was around 4.5kWh/m^2/day, and it was the lowest during December, around 3.3kWh/m^2/day. It can also be noted that south-facing facades received the most solar potential, and the north-facing facades received the least solar potential because the sun is tilted toward the south side. In the end and the beginning of the year, south facades received the most solar potential. This interesting fact can be directly related with the Figure 69. From the December subplot in Figure 78, it is more pronounced that long shadow cast to the north side of the buildings. It'll be interesting to investigate the amount of solar potential received by each side of these building facades over the whole year.

In Figure 80, the annual solar potential is shown from two different angles with two zoomed versions of the same area. It is clear that rooftops and flat surfaces received the most solar potential than any other surfaces. Building facades also received a significant amount of solar potential, where south facing facades received the highest amount of sunlight. From the annual

solar potential estimation, we can also conclude that the south facing walls received the most sunlight over the whole year.

For a better analysis, facades are grouped as north-facing, south-facing, east-facing, and west-facing facades. In Figure 81, daily solar potential over facades facing in four directions is shown. South facing facades received the highest solar potential almost all over the year, except during summer. In summer, south-facing and north-facing facades received almost the same amount of solar potential. Although, east-facing and west-facing facades received the highest amount of solar potential in the summer time because of long day hour and higher solar intensity at the period of the year. Therefore, south-facing facades are the most preferable, and then east-facing and west-facing facades are preferable for solar panel installation for this area. North-facing facades are the least desirable location to mount solar panels.

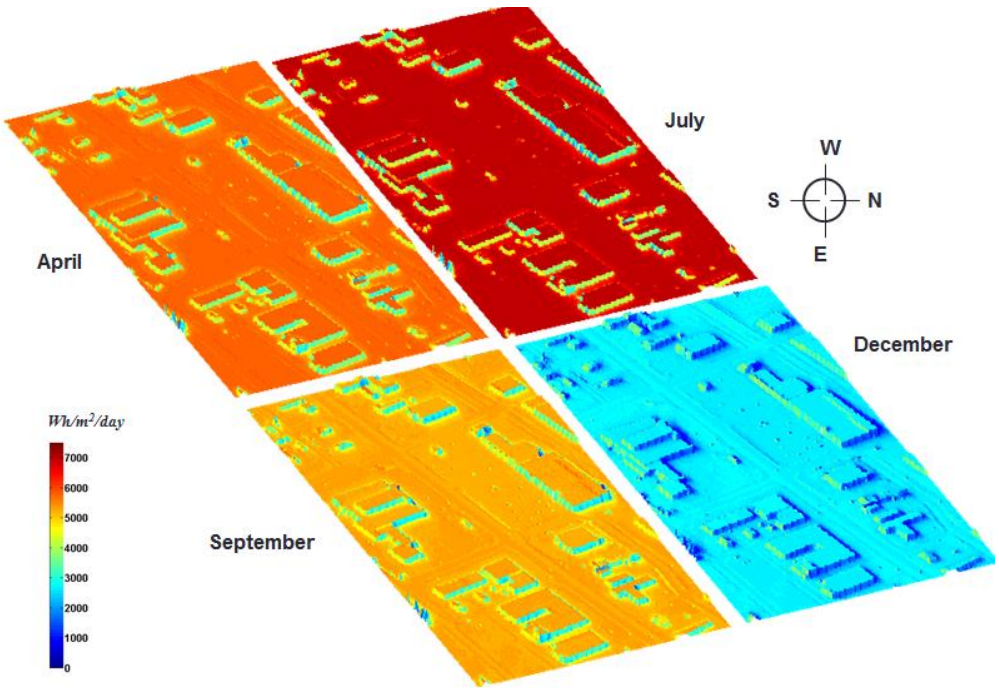

Figure 79: Average amount of solar power received per day at different months of the year. Solar power reception was the highest during the summer season.

Finally, the amounts of solar potential received by facades facing four different sides are compared with the rooftop solar potential in Figure 82. We see that south-facing facades received 1 to 1.4 times more solar potential than rooftops during the winter season. Although, during summer solar power received by the south-facing sides drastically went below the solar power received by the east and west facing facades. East and west facing facades received the almost the same amount of solar energy over the whole year. It can be noted that north-facing facades received the lowest amount of solar power among all facades but during summer it receives the highest solar potential in comparison to the other times of the year.

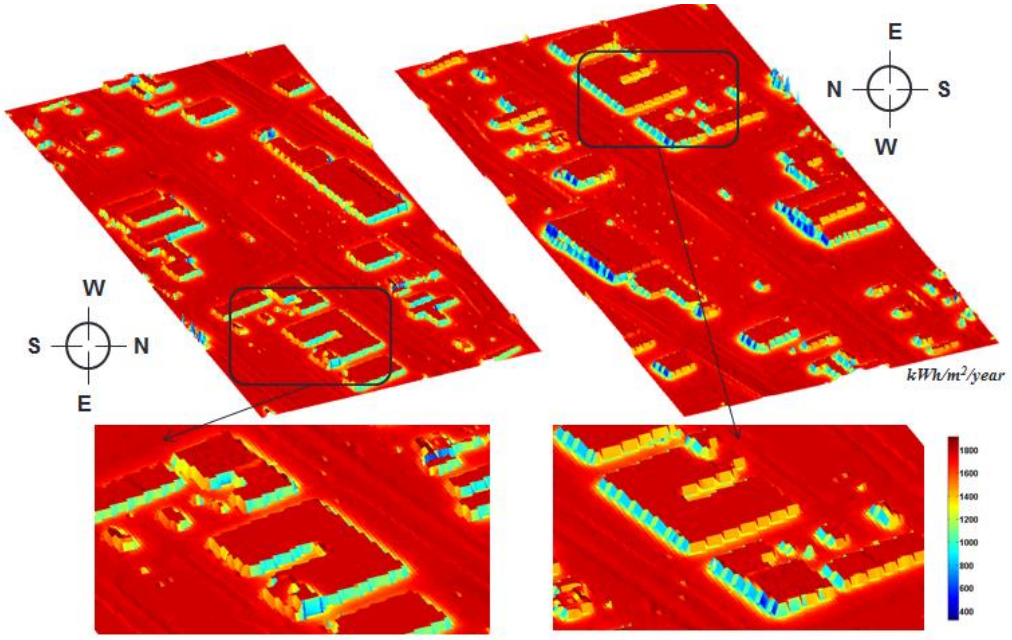

Figure 80: Annual solar power received per square meter on the dataset. Two different angles and two zoomed versions are shown for better visualization.

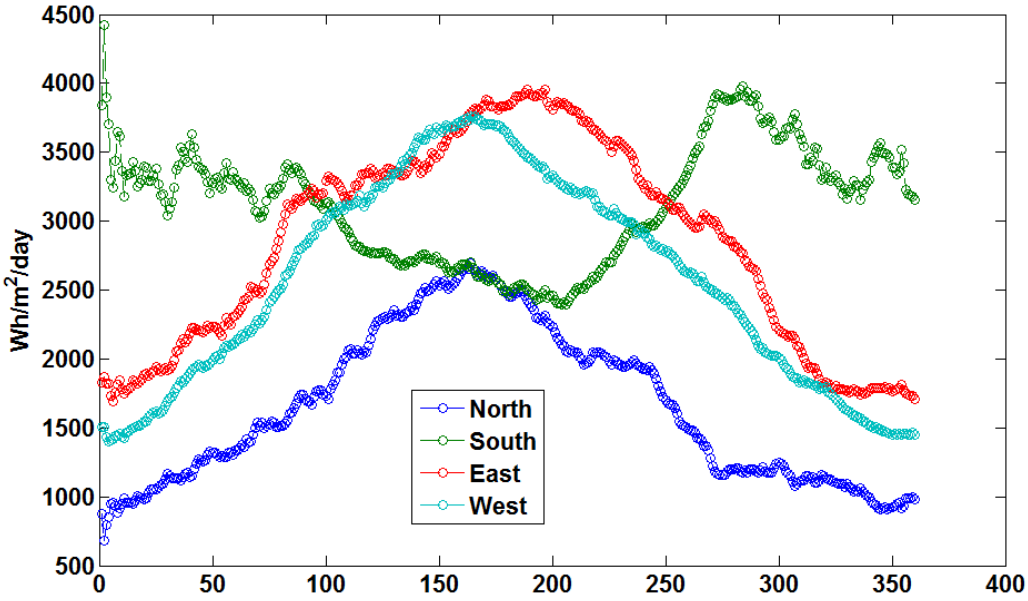

Figure 81: Daily average solar power received by facades facing at different directions.

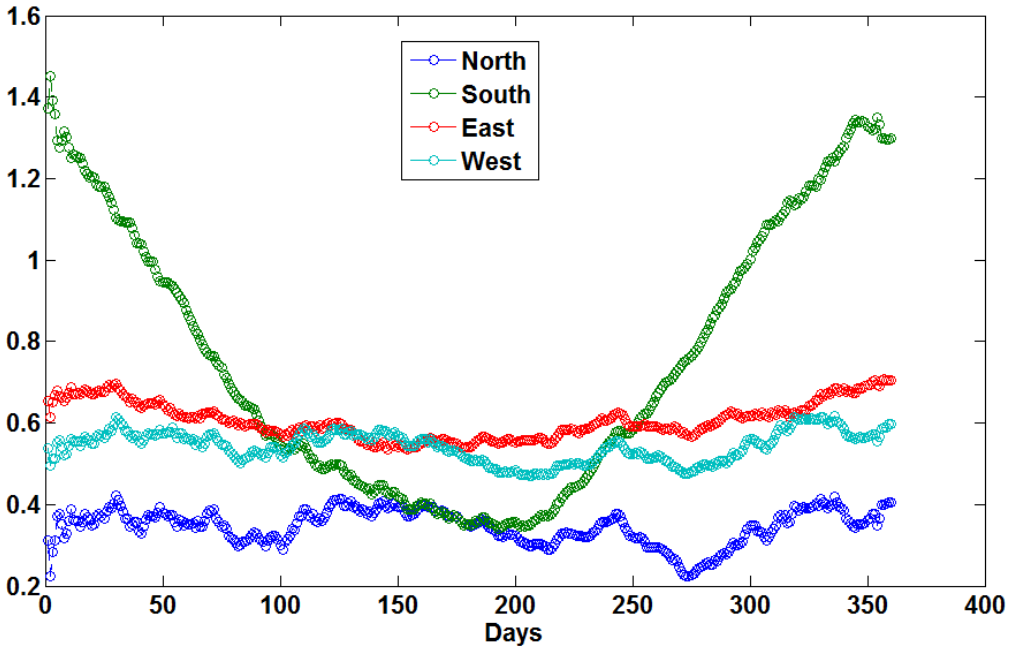

Figure 82: Daily average solar power received by different facades compare with the rooftop.

Finally, the amounts of solar potential received by different surfaces were presented with an amount of bill saving per day. For this purpose, few typical values were considered for the study area. This is an example of calculating saving on electricity bill using solar potential estimation.

Table 1: Solar potential on different surfaces over the year in kWh/m^2/day

Surface type	Jan-Mar $kWh/m^2/day$	Apr-Jun $kWh/m^2/day$	July-Sept $kWh/m^2/day$	Oct-Dec $kWh/m^2/day$	Annual $kWh/m^2/year$
North facade	0.74	1.45	1.23	0.65	374.05
South facade	2.98	1.98	2.23	3.17	944.49
East facade	1.93	2.66	2.89	1.82	850.21
West facade	1.58	2.64	2.42	1.50	743.32
Rooftop	3.67	6.27	6.18	3.38	1780.80

Here we considered: Typical conversion efficiency of poly-crystalline modules is 16%. Typical inverter efficiency is 95%. Typical reduction for temperature is -6%. Typical residential module size is 65 inches by 39 inches. 12 modules setup is popular for household systems. Average electricity bill for residential area in Dallas is 11cent/kWh.

Table 2: Amount of bill saving in dollar from estimated solar potential

Surface type	Jan-Mar $\$/day$	Apr-Jun $\$/day$	July-Sept $\$/day$	Oct-Dec $\$/day$	Annual $\$/year$
North facade	0.23	0.45	0.38	0.20	114.99
South facade	0.92	0.61	0.69	0.97	290.36
East facade	0.59	0.82	0.89	0.56	261.37
West facade	0.49	0.81	0.74	0.46	228.51
Rooftop	1.13	1.93	1.90	1.04	547.45

Chapter 4

Conclusions and Future Work

4.1 Waterbody detection conclusions and future work

We proposed a novel algorithm for still and flowing waterbody detection using both the elevation and the intensity of airborne LiDAR data. Co-registered elevation and intensity information is one of the most valuable properties to exploit from LiDAR data. The proposed algorithm examines only potential areas of interest, rather all areas covered by the LiDAR, using a seeded region growing method. To demonstrate the performance of the proposed algorithm, quantitative results are shown for three different datasets, where manually detected boundaries were available. Visual inspection of three different datasets covering from 500 km^2 to 1500km^2 projects was shown to validate the efficacy of this algorithm.

This algorithm has been developed to run inside the ArcGIS environment and has been adapted for extremely large-scale applications such as state-wide and small country-wide projects. The tools intelligently build batches from large datasets, perform water detection in each batch, merge detections for final results, and perform post processing to handle unique issues in batch boundary regions. The toolbox has been validated by NRCS for large scale dataset. Besides these accomplishments, a few improvements can be made on these ideas in the future, which are listed below.

1) The current tool creates batches and runs each batch separately to produce partial detection results. Each batch can be processed independently in a single machine. However, large projects may take a significant amount of time on a single machine. This code can be easily extended to operate in the Hadoop framework. In that case, each batch can be processed by a slave machine independently. In the batch creation phase, chunks of LAS file can be distributed to slave machines to generate the

neighborhood relation. This improvement enables this tool to run on clusters of computers.

2) Currently, a KS-test was performed to distinguish intensity distributions of water and land objects statistically. This can be performed using a supervised machine learning classification algorithm. The PDF from the waterbodies and the land objects can be used to train an SVM or Neural network to distinguish these two distributions in the iteration phase. This may help for those large projects where intensity data is way far from general situation.
3) Finally, this tool utilized ArcGIS Python API (ArcPy) for rasterization and LAS file reading. Therefore, ArcGIS license is mandatory to use this tool. To make it truly open source and free, other open source packages can be used for LAS file reading and rasterization.

4.2 Solar potential conclusions and future work

We proposed a novel algorithm to estimate solar potential over a scene including rooftops and facades of the buildings. Rather than using the 3D LiDAR point cloud, rasterized elevation data was used for faster operation on a large areas. A patch based representation was utilized to calculate solar potential on any vertical or horizontal surfaces. A building, vegetation, and ground detection method were proposed for airborne LiDAR data. A patch batch 3D model was generated from detected roof-tops and facades.

The solar vector was calculated using an astronomical model, and solar radiation information was gathered from the TMY data collected and standardized by NREL. All of this information was used to estimate the Direct Radiation Model and Diffuse Radiation Model.

The proof-of-concept was shown on a test dataset, and finally, a LiDAR dataset was used for practical estimation. The solar potential is shown for four different seasons over the year.

It was seen that flat planes receive the highest amount solar potential during the summer season. Furthermore, the annual solar potential for each surface is also shown for the same area. It was clear that south facing facades receive the highest amount of solar potential over the year. Therefore, solar potential received by facades facing in different sides were compared. It was concluded that east and west facing facades receive the almost same amount of solar potential, and reach the peak during summer. On the other hand, south facing facades received highest solar potential over the year but dropped during the summer. Additionally, north facing facades receive the lowest amount of solar potential over the whole year.

The core algorithm was developed in this work, and a few conclusions were drawn from a single test dataset. Besides these, a few improvements can be made on these algorithms in the future, which are listed below.

1) Current algorithm does not consider weather effect. Integrating historical weather data may help to a better estimate of the solar potential.
2) This whole algorithm was developed on MATLAB. Multiple scripts were used to finish different steps of this algorithm. The whole process can be brought under single software for user-friendly operation.
3) This algorithm has been tested on an urban scene but yet to be tested on large scale projects. Large scale implementation can be an improvement for this tool.
4) The visualization was performed on a patch in a separate figure. Applying the model on exact geolocation, and visualize over aerial images will be an interesting future work.

References

[1] J. B. Campbell and R. H. Wynne, *Introduction to Remote Sensing*. New York, NY: Guilford Press, 2011.

[2] K. Schmid *et al.*, "LiDAR 101: An Introduction to LiDAR Technology, Data, and Applications," *NOAA Coastal Services Center*, p. 76, 2008.

[3] G. Petrie and C. K. Toth, *Introduction to Laser Ranging, Profiling, and Scanning*. London, UK: CRC Press Taylor Francis, 2008.

[4] L. D. Smullin and G. Fiocco, "Optical Echoes from the Moon," *Nature*, vol. 194, p. 1267, 1962.

[5] M. E. Hodgson and P. Bresnahan, "Accuracy of Airborne LiDAR-derived Elevation," *Photogrammetric Engineering & Remote Sensing*, vol. 70, no. 3, pp. 331–339, 2004.

[6] ASPRS, "LAS Specification Version 1.4 – R12." The American Society for Photogrammetry & Remote Sensing, Jul-2013.

[7] J. D. Spinhirne, "Micro Pulse LiDAR," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 31, no. 1, pp. 48–55, 1993.

[8] C. Hug, A. Ullrich, and A. Grimm, "Litemapper-5600 a waveform digitizing LiDAR terrain and vegetation mapping system," *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 36, no. Part 8, p. W2, 2004.

[9] G. Petrie, "Current Developments in Airborne Laser Scanning Technologies," in *Proceedings of IX International Scientific & Technical Conference*, Attica, Greece, 2009.

[10] A. Habib, K. I. Bang, A. P. Kersting, and J. Chow, "Alternative Methodologies for LiDAR System Calibration," *Remote Sensing*, vol. 2, no. 3, pp. 874–907, 2010.

[11] M. A. Lefsky, W. B. Cohen, G. G. Parker, and D. J. Harding, "Lidar Remote Sensing for Ecosystem Studies LiDAR, an Emerging Remote Sensing Technology that Directly Measures the Three-dimensional Distribution of Plant Canopies, an Accurately Estimate Vegetation Structural Attributes and should be of Particular Interest to Forest, Landscape, and Global Ecologists," *BioScience*, vol. 52, no. 1, pp. 19–30, 2002.

[12] X. Liu and Z. Zhang, "LiDAR Data Reduction for Efficient and High Quality DEM Generation," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 37, pp. 173–178, 2008.

[13] X. Liu, "Airborne LiDAR for DEM Generation: Some Critical Issues," *Progress in Physical Geography*, vol. 32, no. 1, pp. 31–49, 2008.

[14] X. Meng, L. Wang, J. L. Silván-Cárdenas, and N. Currit, "A Multi-directional Ground Filtering Algorithm for Airborne LiDAR," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 64, no. 1, pp. 117–124, 2009.

[15] X. Meng, N. Currit, and K. Zhao, "Ground Filtering Algorithms for Airborne LiDAR Data: A Review of Critical Issues," *Remote Sensing*, vol. 2, no. 3, pp. 833–860, 2010.

[16] A. F. Jones, P. A. Brewer, E. Johnstone, and M. G. Macklin, "High-resolution Interpretative Geomorphological Mapping of River Valley Environments using Airborne LiDAR Data," *Earth Surface Processes and Landforms*, vol. 32, no. 10, pp. 1574–1592, 2007.

[17] D. P. Thoma, S. C. Gupta, M. E. Bauer, and C. Kirchoff, "Airborne Laser Scanning for Riverbank Erosion Assessment," *Remote Sensing of Environment*, vol. 95, no. 4, pp. 493–501, 2005.

[18] J. French, "Airborne LiDAR in Support of Geomorphological and Hydraulic Modelling," *Earth surface processes and landforms*, vol. 28, no. 3, pp. 321–335, 2003.

[19] K. A. Addo, M. Walkden, and J. P. t Mills, "Detection, Measurement and Prediction of Shoreline Recession in Accra, Ghana," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 63, no. 5, pp. 543–558, 2008.
[20] G. Schumann, P. Matgen, M. Cutler, A. Black, L. Hoffmann, and L. Pfister, "Comparison of Remotely Sensed Water Stages from LiDAR, Topographic Contours and SRTM," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 63, no. 3, pp. 283–296, 2008.
[21] J. Adams and J. Chandler, "Evaluation of LIDAR and Medium Scale Photogrammetry for Detecting Soft-cliff Coastal Change," *The Photogrammetric Record*, vol. 17, no. 99, pp. 405–418, 2002.
[22] N. S. Anders, A. C. Seijmonsbergen, and W. Bouten, "Geomorphological change detection using object-based feature extraction from multi-temporal LiDar data," *Geoscience and Remote Sensing Letters, IEEE*, vol. 10, no. 6, pp. 1587–1591, 2013.
[23] J. P. M. O'Neil-Dunne, S. W. MacFaden, A. R. Royar, and K. C. Pelletier, "An Object-based System for LiDAR Data Fusion and Feature Extraction," *Geocarto International*, vol. 28, no. 3, pp. 227–242, 2013.
[24] H. F. Stockdonf, A. H. Sallenger Jr, J. H. List, and R. A. Holman, "Estimation of Shoreline Position and Change using Airborne Topographic LiDAR Data," *Journal of Coastal Research*, pp. 502–513, 2002.
[25] S. P. Leatherman, "Shoreline Change Mapping and Management along the US East Coast," *Journal of Coastal Research*, pp. 5–13, 2003.
[26] H. Liu, D. Sherman, and S. Gu, "Automated Extraction of Shorelines from Airborne Light Detection and Ranging Data and Accuracy Assessment based on Monte Carlo Simulation," *Journal of Coastal Research*, pp. 1359–1369, 2007.
[27] A. F. Elaksher, "Fusion of Hyperspectral Images and LiDAR-based DEMs for Coastal Mapping," *Optics and Lasers in Engineering*, vol. 46, no. 7, pp. 493–498, 2008.
[28] H. Liu, L. Wang, D. J. Sherman, Q. Wu, H. Su, and others, "Algorithmic Foundation and Software Tools for Extracting Shoreline Features from Remote Sensing Imagery and LiDAR Data," *Journal of Geographic Information System*, vol. 3, no. 02, p. 99, 2011.
[29] G. Chust, I. Galparsoro, A. Borja, J. Franco, and A. Uriarte, "Coastal and Estuarine Habitat Mapping, using LiDAR Height and Intensity and Multi-Spectral Imagery," *Estuarine, Coastal and Shelf Science*, vol. 78, no. 4, pp. 633–643, 2008.
[30] Y. Chen, W. Su, J. Li, and Z. Sun, "Hierarchical Object Oriented Classification using Very High Resolution Imagery and LiDAR Data over Urban Areas," *Advances in Space Research*, vol. 43, no. 7, pp. 1101–1110, 2009.
[31] D. Gilvear, A. Tyler, and C. Davids, "Detection of Estuarine and Tidal River Hydromorphology using Hyper-spectral and LiDAR Data: Forth estuary, Scotland," *Estuarine, Coastal and Shelf Science*, vol. 61, no. 3, pp. 379–392, 2004.
[32] A. Antonarakis, K. S. Richards, and J. Brasington, "Object-based Land Cover Classification using Airborne LiDAR," *Remote Sensing of Environment*, vol. 112, no. 6, pp. 2988–2998, 2008.
[33] B. Höfle, M. Vetter, N. Pfeifer, G. Mandlburger, and J. Stötter, "Water Surface Mapping from Airborne Laser Scanning using Signal Intensity and Elevation Data," *Earth Surface Processes and Landforms*, vol. 34, no. 12, pp. 1635–1649, 2009.
[34] A. Brzank, C. Heipke, J. Goepfert, and U. Soergel, "Aspects of Generating Precise Digital Terrain Models in the Wadden Sea from LiDAR–water Classification and Structure Line Extraction," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 63, no. 5, pp. 510–528, 2008.
[35] B. Höfle and N. Pfeifer, "Correction of Laser Scanning Intensity Data: Data and Model-driven Approaches," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 62, no. 6, pp. 415–433, 2007.

[36] I.-C. Lee, B. Wu, and R. Li, "Shoreline Extraction from the Integration of LiDAR Point Cloud Data and Aerial Orthophotos using Mean Shift Segmentation," in *Annual Conference Baltimore, Maryland, ASPRS*, 2009.
[37] S. S. Deshpande, "Improved Floodplain Delineation Method Using High-Density LiDAR Data," *Computer-Aided Civil and Infrastructure Engineering*, vol. 28, no. 1, pp. 68–79, 2013.
[38] P. Passalacqua, T. Do Trung, E. Foufoula-Georgiou, G. Sapiro, and W. E. Dietrich, "A Geometric Framework for Channel Network Extraction from LiDAR: Nonlinear Diffusion and Geodesic Paths," *Journal of Geophysical Research: Earth Surface (2003–2012)*, vol. 115, no. F1, 2010.
[39] P. Passalacqua, P. Tarolli, and E. Foufoula-Georgiou, "Testing Space-scale Methodologies for Automatic Geomorphic feature Extraction from LiDAR in a Complex Mountainous Landscape," *Water Resources Research*, vol. 46, no. 11, 2010.
[40] G. Toscano, U. Gopalam, and V. Devarajan, "Auto Hydro Break Line Generation Using LiDAR Elevation and Intensity Data," in *Proceedings, ASPRS Conference, Louisville, Kentucky*, 2014.
[41] G. J. Toscano, "A LiDAR-based Auto Hydro Breakline Generation Algorithm for Standing Waterbodies," University of Texas Arlington, Arlington, Texas, 2015.
[42] P. P. Acharjee, G. J. Toscano, and V. Devarajan, "A Novel Angular Filter based LiDAR Point Cloud Classification," in *ASPRS Annual Conference*, Tampa, Florida, 2015, pp. 42–51.
[43] F. J. Massey Jr, "The Kolmogorov-Smirnov Test for Goodness of Fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.
[44] S. Freitas, C. Catita, P. Redweik, and M. C. Brito, "Modelling Solar Potential in the Urban Environment: State-of-the-art Review," *Renewable and Sustainable Energy Reviews*, vol. 41, pp. 915–931, 2015.
[45] D. Santos-Martin and S. Lemon, "SoL–A PV Generation Model for Grid Integration Analysis in Distribution Networks," *Solar Energy*, vol. 120, pp. 549–564, 2015.
[46] J. A. Jakubiec and C. F. Reinhart, "A Method for Predicting City-wide Electricity Gains from Photovoltaic Panels based on LiDAR and GIS Data Combined with Hourly Daysim Simulations," *Solar Energy*, vol. 93, pp. 127–143, 2013.
[47] N. Lukač, D. Žlaus, S. Seme, B. Žalik, and G. Štumberger, "Rating of Roofs' Surfaces regarding their Solar Potential and Suitability for PV Systems, based on LiDAR Data," *Applied energy*, vol. 102, pp. 803–812, 2013.
[48] P. Fu and P. M. Rich, "Design and Implementation of the Solar Analyst: an ArcView Extension for Modeling Solar Radiation at Landscape Scales," in *Proceedings of the Nineteenth Annual ESRI User Conference*, 1999, pp. 1–31.
[49] J. Hofierka, M. Suri, and others, "The Solar Radiation Model for Open Source GIS: Implementation and Applications," in *Proceedings of the Open source GIS-GRASS users conference*, 2002, vol. 2002, pp. 51–70.
[50] P. Redweik, C. Catita, and M. Brito, "Solar Energy Potential on Roofs and Facades in an Urban Landscape," *Solar Energy*, vol. 97, pp. 332–341, 2013.
[51] C. Magalhães Carneiro, "Extraction of Urban Environmental Quality Indicators using LiDAR-based Digital Surface Models," École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 2011.
[52] T. P. Chang, "The Sun's Apparent Position and the Optimal Tilt Angle of a Solar Collector in the Northern Hemisphere," *Solar energy*, vol. 83, no. 8, pp. 1274–1284, 2009.
[53] I. Reda and A. Andreas, "Solar Position Algorithm for Solar Radiation Applications," *Solar energy*, vol. 76, no. 5, pp. 577–589, 2004.

[54] A. B. Sproul, “Derivation of the Solar Geometric Relationships using Vector Analysis,” *Renewable Energy*, vol. 32, no. 7, pp. 1187–1205, 2007.
[55] C. Honsberg and S. Bowden, “Photovoltaic Education Network,” 06-Mar-2017.
[56] NREL, “National Solar Radiation Data Base,” 06-Mar-2017.