LiDAR Man-Made Object Detection

William E. Boeing Department of Aeronautics & Astronautics
University of Washington, Seattle, WA 98195-2400

2018-06-04

Abstract

Light Detection and Ranging (LiDAR) sensors are typically used in large scale aircraft, but recent technological advancements have made sensors smaller and cheaper, making them a viable option in small unmanned aerial vehicles (UAVs). This project will investigate the use of LiDAR technology on an octocopter provided by the Autonomous Flight Systems Laboratory (AFSL) at the University of Washington (UW). Software is being developed to perform object detection on the LiDAR flight data to detect the presence of man-made objects scattered at UW Carnations farm. This will be accomplished by first separating all of the objects in the data reduction process into clusters, and then performing a 3D Hough transform algorithm on each object to flag it as artificial or not. This algorithm runs under the assumption that artificial objects contain flat surfaces and linear edges, while natural objects are shaped much more randomly. After running the algorithm against data obtained from test flights at Carnation Farms, the software successfully classified a tent as a man-made object while rejecting other natural objects such as bushes and trees.

I. Nomenclature

m_x	x-direction slope
m_y	y-direction slope
n	normal vector
n_x	x component of the normal vector
n_y	y component of the normal vector
n_z	z component of the normal vector
P_x	x value of the Point in the plane
P_y	y value of the Point in the plane
P_z	z value of the Point in the plane
θ	zenith direction (rad)
ρ	radius of curvature (m)
ρ_{max}	distance from the origin to the furthest point in the point cloud (m)
ρ_{min}	distance from the origin to the closest point in the point cloud (m)
ϕ	azimuth direction (rad)

II. Introduction

LiDAR is a remote sensing method that pulses lasers at a target and measures the reflection time to determine the distance. LiDAR sensors pulse lasers at extremely high frequencies at various angles in order to generate a three dimensional point cloud of the surrounding environment. These sensors are most commonly known for their use on road vehicles with autonomous driving capability which use the data to identify obstacles in real time. LiDAR is ideal for such applications because it provides a low-lag 360° map of the surroundings at relevant distance ranges for driving, but its usefulness on an aerial platform where scanning ranges are much greater is less known. For this project, a LiDAR sensor was mounted on a remotely piloted octocopter, both provided by the Autonomous Flight Systems Laboratory. There were three main objectives for this project. Initially, an algorithm was developed to detect artificial objects. The algorithm was then tested by performing flight tests at the UW Carnation Farms UAS Test Site to gather point clouds of artificial objects such as cars and boxes in natural surroundings. The last objective was to develop software to post-process the vast amount of LiDAR data collected during the test flights and develop a computer vision algorithm to flag the man-made objects in view of the sensor. The goal of this project was to create the first stepping stone towards developing a real-time flagging software which could be used towards multiple applications such as alerting pilots of nearby poachers in wildlife environments. Due to the limited scope of this project, the goal was not to identify

artificial objects with absolute precision, but rather flag possible objects to be verified by a human operator.

III. Theory

Euler Rotations

Due to the varying orientation of the LiDAR sensor and the varying attitude of the vehicle, coordinate rotations were performed to the data in order to maintain a consistent frame of reference. A set of points can be multiplied by a rotation matrix to perform a counterclockwise rotation in Euclidean space. In two dimensions, the rotation matrix is defined in Eq. 1.

$$R = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix} \tag{1}$$

The rotation matrix can easily be manipulated to perform rotations along a principal axis in 3D space. The axis of rotation is the normal axis of the plane that is being rotated. For example, the 3D rotation along the z axis is shown in Eq. 2.

$$R_z = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Hough Transform

There are many computer vision algorithms that can be used to process LiDAR data and perform object detection. The Hough Transform method was chosen for its relative simplicity. The Hough Transform is a method developed in 1962 by Paul Hough to detect lines and circles in images or two dimensional point clouds [1]. The algorithm can be slightly tweaked to be used to detect planes in three dimensional point clouds, which makes it the most promising method to achieve the objective of detecting man-made objects.

An understanding of 2D Hough Transform would make the 3D Hough Transform a lot more intuitive. The 2D Hough Transform is capable of detecting lines in a 2D Cartesian space where many points lie on. The first step of the algorithm is transforming all of the points from the Cartesian space to the Hough space. Recall that a line can be defined with two parameters: the slope (m) and the y-intercept (b), in the form

$$y = mx + b \tag{3}$$

To transform this equation to the Hough space, the equation is algebraically rearranged to solve for m and b as a function of the original parameters x and y, as shown below.

$$b = -xm + y \tag{4}$$

A line in the (x, y) space is transformed into a single point (m, b) in the Hough space. Similarly, a line on the Hough space represents a single (x, y) point in the Cartesian space. Intuitively, this makes sense. For a given (x, y) point, there are infinite number of lines of different slopes and y-intercepts that cross that point.

Figure 1 demonstrates this transformation from the Cartesian space to the Hough space for two (x, y) points. As expected, this resulted in two lines in the Hough space. The point (m, b) where these two lines intersect is the common line that the two (x, y) points lie on. The line with slope m and y-intercept b is plotted back on the Cartesian space to confirm that it is indeed the line that lies on both (x, y) points.

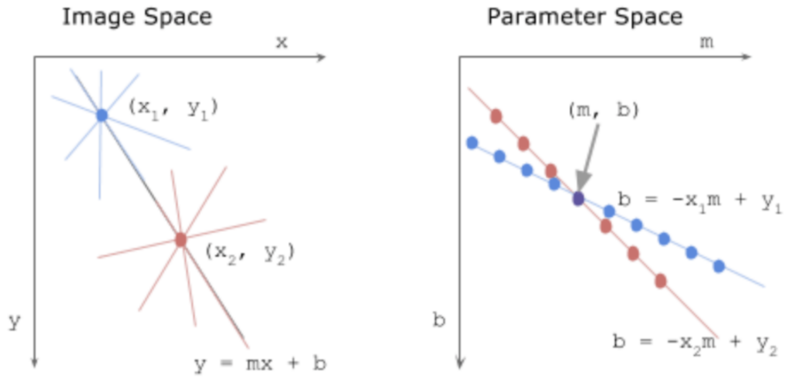


Figure 1: Transformation from Cartesian Space to Hough Space

The 3D Hough Transform follows the same principle, but instead uses spherical coordinates (θ, ϕ, ρ) to represent a plane that lies on a point (x, y, z). The graphical representation of the spherical coordinates are shown in Figure 2. The θ angle is the angle of the normal vector on the xy plane, which ranges from 0° to 360°. The ϕ angle is the angle between the xy plane and the normal vector in the z direction, which ranges from -90° to 90°. ρ is the normal distance from the plane to the origin.

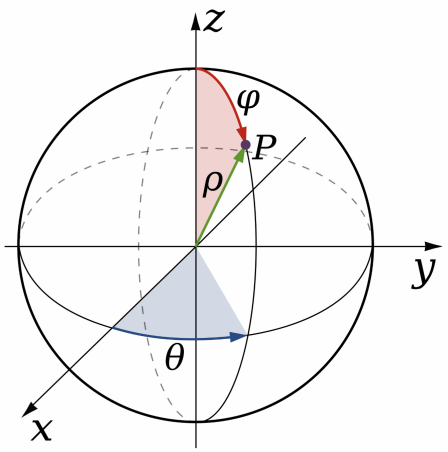


Figure 2: Graphical Representation of Spherical Coordinates

A plane can be represented with a point **P** on the plane and the normal vector **n** perpendicular to the plane [1]. This plane can be represented with Eq. 5.

$$\rho = \boldsymbol{P} \cdot \boldsymbol{n} = p_x n_x + p_y n_y + p_z n_z \tag{5}$$

The components of the normal vector can be broken down into its geometric components as shown in Eq. 6.

$$\rho = p_x cos(\theta) sin(\phi) + p_y sin(\phi) sin(\theta) + p_z cos(\phi) \tag{6}$$

A plane in the (x, y, z) Cartesian space can be represented as a single point (θ, ϕ, ρ) in the Hough space. Similarly, a plane in the Hough space represents a point in the Cartesian space. Points that lie on the same Cartesian plane can be identified by their respective planes intersecting at a common (θ, ϕ, ρ) value in the Hough space[1].

IV. Apparatus

The LiDAR platform used was Velodyne Pucklite VLP-16, shown in Fig. 3 on the next page which has a 30° vertical and 360° horizontal field of view when unobstructed by mounting hardware. The unit uses 16 rotating lasers that give 2° vertical resolution and between 0.1 and 0.4° horizontal resolution depending on the desired sample rate. The UAV, shown in Fig. 4 on which the LiDAR was flown is a remote-controlled octocopter owned by the UW AFSL with a GPS-enabled PixHawk programmable autopilot. It can operate about 15 minutes with a takeoff weight of 9.5kg at an altitude up to 500 feet. It also has a max power of 500 Watts and retractable landing gear.

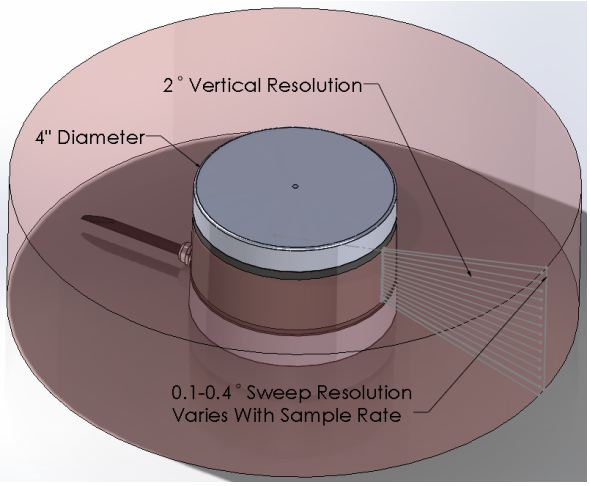


Figure 3: LiDAR System: Velodyne Pucklite VLP-16

Figure 4: Argo: UAV Used to Carry LiDAR Sensor (41in Boom Diameter)

LiDAR Mounting Bracket Design

The previous LiDAR mounting configuration, shown in Fig. 5 (next page), constrained the amount of data that could be collected and also gave undesired projected point resolution. This configuration also made it impossible for the 16 internal lasers of the LiDAR to obtain any data from objects directly underneath the octocopter. Since the lasers were projected radially in this configuration, considerable terrain was overseen and many of the lasers projected into empty space returning useless data (i.e. some of the lasers "hit" the surrounding air rather than objects on the ground). To alleviate this issue, a new mounting bracket configuration was developed to mount the LiDAR underneath the UAV to enable the LiDAR system to effectively scan the ground directly underneath as shown in Fig 7 (next page). The new configuration had the LiDAR resting on its side relative to the octocopter. The LiDAR was clamped between 3D printed cups and aluminum sheets in the same fashion as the previous configuration, except new all-thread was bent to hang the assembly in the new orientation as seen underneath the UAV in Fig. 6 (next page). Additional bolts were also added horizontally in order to clamp the LiDAR between the brackets. The proposed design was mainly driven with the intention of utilizing the geometry and features of the old design in order to minimize manufacturing cost and time. Additionally, this new configuration maintained the center of gravity of the octocopter to avoid over-working its motor controller. With this configuration, a more robust set of 3D point clouds could be obtained which greatly improved the accuracy of the human-made object detection algorithm developed by the team.

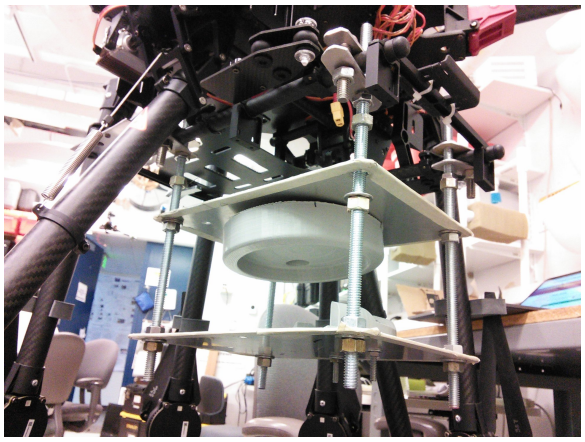


Figure 5: Previous Horizontal LiDAR Mounting Configuration

Figure 6: Modified LiDAR Mount on Bottom of UAV

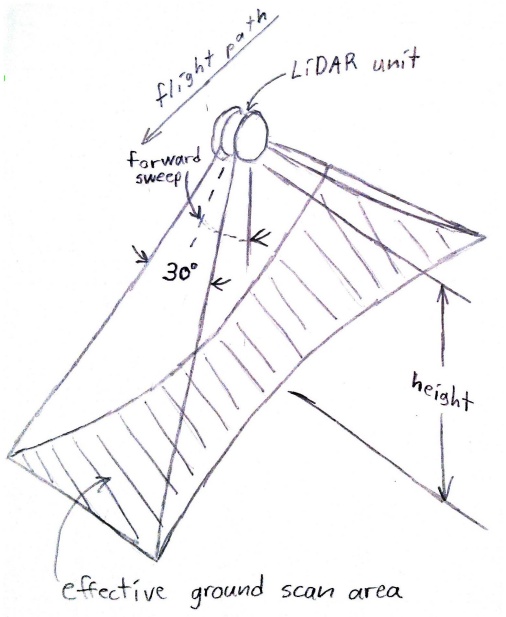


Figure 7: LiDAR Scanning Configuration

V. Approach

Flight Testing

In order to simulate a real-world application of the LiDAR for ground target identification, a flight test was devised at Carnation Farms, WA in coordination with the AFSL. A GPS track was flown as shown in Fig. 8 (next page). Several cardboard boxes roughly 2ft in all dimensions were placed in the field under the determined flight path, along with two identical camping tents with roughly 8ft square footprints.

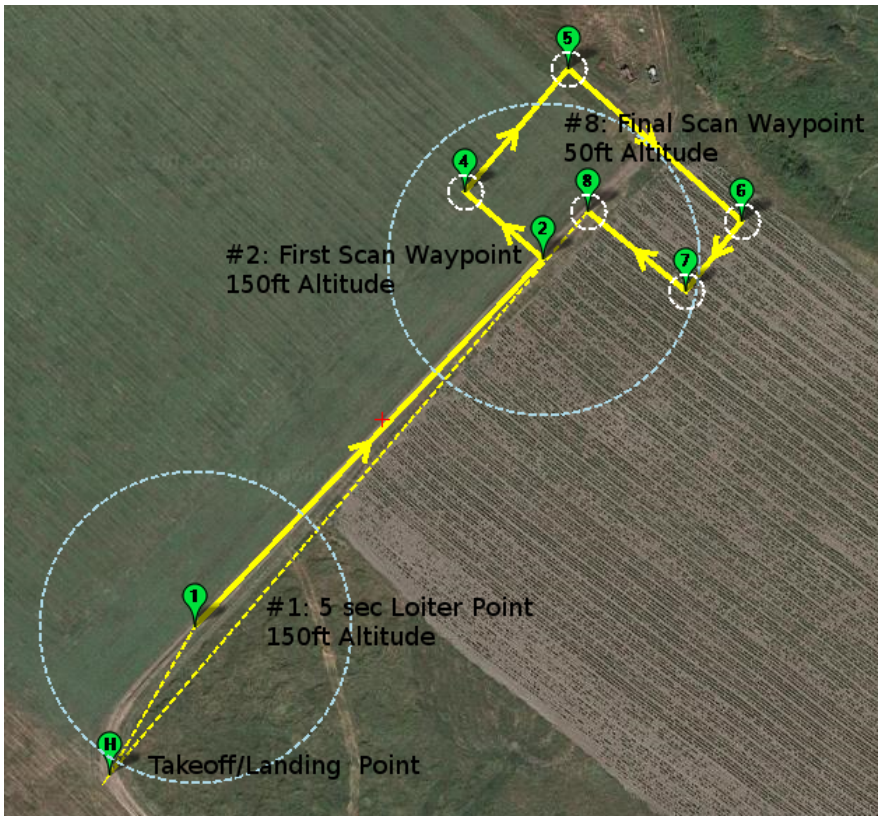


Figure 8: Carnation Farms LiDAR Flight Test Path

In order to gather data at a variety of resolutions, the flight altitude linearly decreased from 150ft to 50ft between the 2nd and 8th waypoints shown in Fig. 8. At the highest altitude this rendered an effective ground scan area 130ft long and 80ft at the narrowest section, with a 6ft or finer projected point resolution. The scan area and resolution decreased proportionally with the altitude.

Three total flights were executed along the same path. The first flight was performed with a replica mass equivalent to LiDAR mounted on Argo to verify the capability of the new mounting system to hold the real LiDAR securely and overfly the desired area to adequately capture all the objects laid on the ground. The second flight was performed with the LiDAR scanning the ground objects previously placed. For the third flight, all ground objects were removed to scan purely natural ground as a controlled data set.

LiDAR Data Conversion and Visualization

After successful collected data from the flight test, post-flight data processing was continued to stay of the road map of this project. The data from the LiDAR system was stored in

the form of a .pcap file, which was a video of point cloud snapshots and could not be read by conventional data analysis programs such as MATLAB or Excel. Therefore, each desired frame was exported from the VeloDyne VeloView software into a separate .csv file. Each frame of the raw data contained an order of 10^4 data points. Each data point contained the x, y, z coordinates relative to the LiDAR sensor, and a reflectivity constant which was proportional to the amount of light that was reflected from the point.

The preferred orientation of the coordinate system was to have the xy plane be the ground, and $+z$ be altitude. The sample data provided by the AFSL was already in the correct orientation, but the flight test data needed to be rotated because the LiDAR sensor was mounted vertically instead of horizontally. The drone was also pitched down as it was moving forward, so the ground appeared to be at an incline relative to the LiDAR sensor. Hence, there were two rotations necessary to orient the data correctly. One rotation was a 90° rotation along the z axis of the LiDAR sensor, and the other rotation oriented the ground so that it lied on the xy plane. The second rotation angle varied frame by frame because the angle of attack of the drone was constantly changing.

Object Clustering in LiDAR Map

The first step in object classification was to separate the objects in the LiDAR map so that the detection algorithm can be run on each individual object. Essentially, this was done by grouping together points that are close to one another. The algorithm is similar to Kruskal's algorithm, a computer science algorithm for finding the minimum-spanning tree in a graph [2]. To eliminate sparse points of the texture of the ground, only points with a z (altitude) value above a small threshold were kept. Due to the nature of aerial data, the projection of data points in the xy plane was considered to be reasonably accurate representation of the 3-D view from the LiDAR sensor. This phenomenon can be seen as similar to when one looks out the window on an airplane. The person would be technically seeing the ground in 3-D, but the view would appear 2-D because the height of the objects are so small compared to the altitude of the airplane.

The algorithm started by computing the 2-D distances between every combination of two data points. The data structure containing these distances were then sorted from least to greatest. The algorithm then looped through all of the pairs of points, and assigned those pairs to the same cluster if the distance between those points were under a defined threshold distance. If the distances between the current two points were greater than the threshold distance, then those points were put into separate clusters. This loop continued until all

of the points were assigned into a cluster. The x, y, and z coordinates of every point in a specific cluster were then placed into an array so further post processing can be done on individual clusters of points.

Hough Transform Algorithm

After the large point cloud was clustered into separate objects, the Hough transform Algorithm was performed on each object to determine whether or not it is an artificial object. The first method of the Hough transform algorithm that was implemented was the standard algorithm, which was the easiest to implement but not the most efficient. A second method of the Hough transform algorithm known as the randomized Hough Transform, which is much more efficient but more difficult to implement, was attempted, but not finished.

Standard 3D Hough Transform

Recall that a point $\mathbf{P}$ in Euclidean space would represent a surface in Hough space. The first step of the standard algorithm was to transform all of the (x, y, z) points into their respective Hough surfaces. For simplification purposes, the point cloud representing the object was translationally shifted so that the numerical average of the points lay on the origin. Due to the noise in the LiDAR data, checking if sets of points lay exactly on the same plane was not feasible. Therefore, a set of tolerances needed to be implemented. An accumulator matrix was formed to store all of the Hough data. The accumulator matrix was a linearly spaced size n x n x n three dimensional square matrix, with θ, ϕ, and ρ as its dimensions, shown in Fig. 9. Initially, all of the values in the matrix are 0. The θ values ranged from 0° to 360°, ϕ values ranged from -90° to 90°, and the ρ values ranged from -ρ_{max} to ρ_{max}. This value of ρ_{max} was chosen because actual values of ρ would never be larger than this value. The size n of the accumulator matrix was determined on a case-by-case basis depending on how many points the point cloud contained because of the high computational expense of the algorithm.

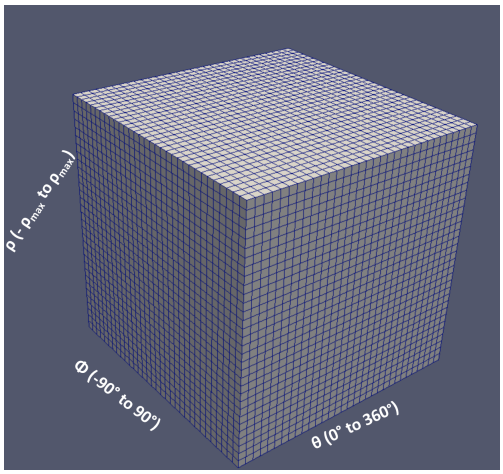


Figure 9: Visualization of Accumulator Matrix

For each Euclidean point, the corresponding value of ρ was calculated from Eq. 6 for each value of θ and ϕ in the accumulator matrix. The ρ values were then rounded to the nearest ρ value in the accumulator matrix, and the value of that specific (θ, ϕ, ρ) accumulator cell increased by 1. After Hough Transformation for every single point was done, the accumulator cells with the highest values represented the most probable planes in Euclidean space that the points lay on or near.

This algorithm worked well if only one plane was desired. This plane would simply be represented by the (θ, ϕ, ρ) accumulator cell with the highest value. However, if an object contained multiple planes, further processing would be necessary. The top five planes in the accumulator matrix would not necessarily represent the five most probable planes on the object. Depending on the spacing of the accumulator matrix, it would be very likely that the top five planes were actually duplicates of each other and represent the same set of points. This is shown in Figure 10. The example consists of 25 points that lie near the same plane with some random noise. It can be seen that the algorithm detected multiple planes when there should have only been one plane. Humans can easily spot that these planes are duplicates of each other, but the algorithm needs to be trained further to detect these duplicates.

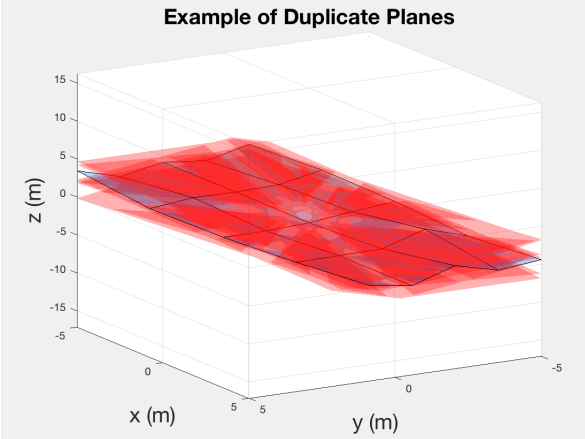


Figure 10: Example of Duplicate Planes from Standard Hough Transform

Due to the computational intensity of detecting duplicates, only planes that had accumulator values greater than a certain threshold were kept. Since the number of points in an object varied, the threshold value was set at a percentage of the number of points. Finding a threshold that would work for the majority of objects was another challenge. Theoretically, this threshold number also correlated with the number of flat surfaces an object contained. However, the algorithm would have no way of knowing the number of flat surfaces an object contained without human input, so this method could not be used.

The simplest way to accomplish the task of removing duplicate planes was to set a tolerance for each variable (θ, ϕ, ρ), iterate through all of the found planes, group planes that had all three variables in tolerance of each other, and the resultant plane would be the average of each of the three variables. Every object would have a different size and shape, thus it was difficult to find one set of tolerances that worked generically.

The last step was developing the flagging criteria. This last step has not been fully developed, but the approach is promising and can be performed manually through data visualization. The proposed criteria is that if more than 75% of the points in the dataset are within a certain small distance of the detected planes, they are to be flagged as artificial. This approach accounts for false positives, when planes are detected due to a coincidental arrangement of a small subset of points, while the rest of the data is randomly scattered.

VI. Discussion of Results

LiDAR Sample Data Visualization

Before the Carnation test flights took place, preliminary analysis was performed on sample LiDAR datasets gathered by AFSL on the UW campus in order to develop and test the object detection algorithm. One frame of data from the middle of Red Square can be seen in Fig. 11. The sides of the buildings on Red Square were captured well, but the points were sparse on the altitude axis due to the orientation of the lasers in the LiDAR sensor. The entirety of buildings were not captured because the LiDAR sensor was not at a high enough altitude to view the entire building. This sample data was used to develop the clustering algorithm before the flight test data was collected.

The LiDAR sensor generated extra points on the Z = 0 plane, shown in Figure 11; these extra points are the concentric circles in the middle of the map. The cause of the generation of these extra points was unknown, but they can be simply removed by eliminating all points within two meters of $z = 0$ as can be seen in Fig. 12 where the concentric circles from Fig. 11 have been removed.

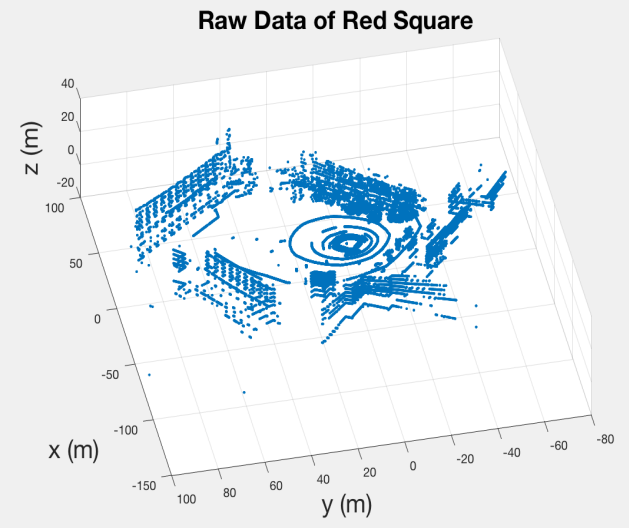


Figure 11: Sample LiDAR Data of Red Square

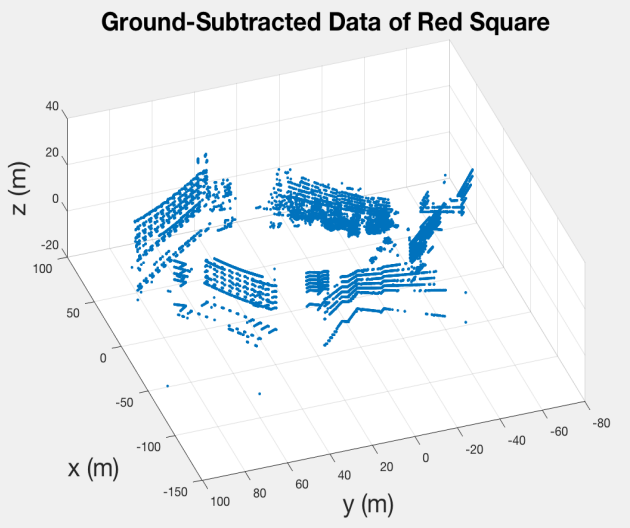


Figure 12: View of Red Square LiDAR Data with Ground Data Removed

Object Clustering

The object clustering algorithm described previously was implemented to the sample data. By cross referencing the video feed of the LiDAR data, several buildings and other objects,

shown in Fig. 13 were found to be in this specific frame of sample data. The separate point clusters are manually labeled to aid in the visualization of the data. To reduce the computational time, only every third point in the LiDAR dataset was used. Two points were considered to be part of different objects if the distance between them was greater than five meters. The algorithm also ignored clusters that had less than a total of 20 points. These numbers were chosen through trial and error with this particular dataset, but once more datasets are acquired, statistical analysis can be performed to determine the optimal thresholds.

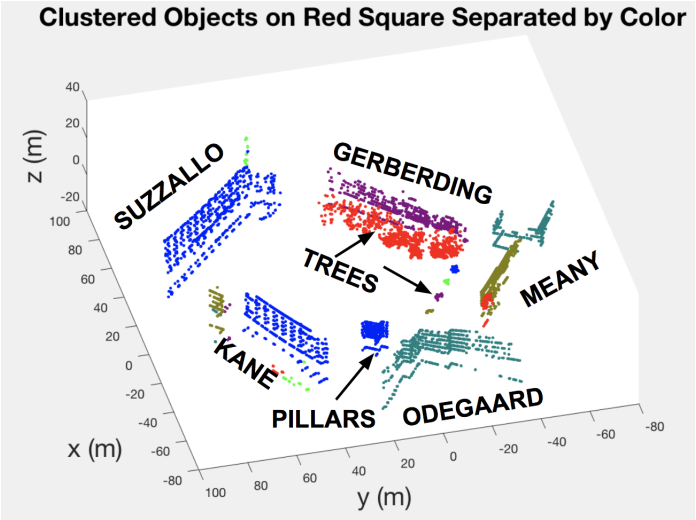


Figure 13: Object Clustering Algorithm performed on sample data

It can be seen that the buildings are accurately distinguished from each other. However, the trees are too close to each other for the algorithm to accurately separate them. This is partly due to the position of the LiDAR sensor and drone. If the drone were to capture data at a higher altitude, the trees may have appeared to be more separated.

The Standard Hough Transform algorithm was tested on the pillar to the left of Kane Hall. After tolerances and thresholds were tuned manually, the algorithm successfully located the two planes that captured the object. Figure 14 shows the result of the algorithm. Since these two planes captured at least 75% of the points, the object was flagged as artificial.

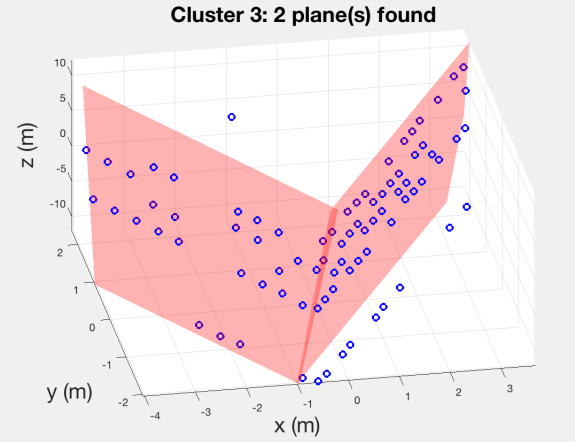


Figure 14: Standard Hough Transform Algorithm on Pillar

Carnation Data Analysis

An aerial view of the placement of one of the tents is shown in Figure 15 overlaid with the LiDAR data. The shapes of the ground contours were hyperbolic, as predicted in Figure 7. Two point cloud objects representing the tent and a possible bush can be seen more clearly by shifting the perspective to ground level and zooming in, as shown in Figure 16.

Figure 15: LiDAR Scan Frame Overlaid on Aerial Photo

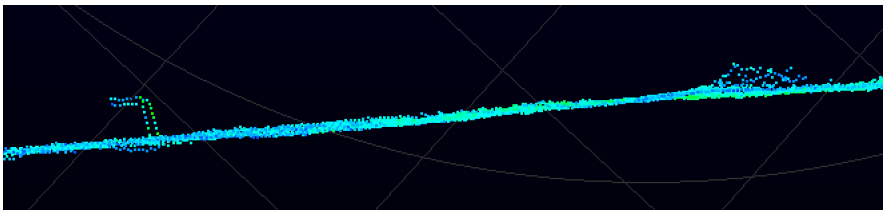


Figure 16: Zoomed in Raw Flight Test LiDAR Data Displaying Tent and Bush

Two coordinate rotations were performed to correct the orientation of the data: a 90° rotation in the z axis, and a $\tilde{4}5°$ rotation in the y axis so that the ground lay on the xy plane. All points below an altitude of 1.5 meters was considered to be ground data. The threshold distance for the object clustering algorithm for distinguishing objects was set at five meters. The resulting ground subtracted clustered data of the chosen frame is shown in Figure 17. Three separate objects were found: a tent, bush, and a collection of ground/tree data above 1.5 meters.

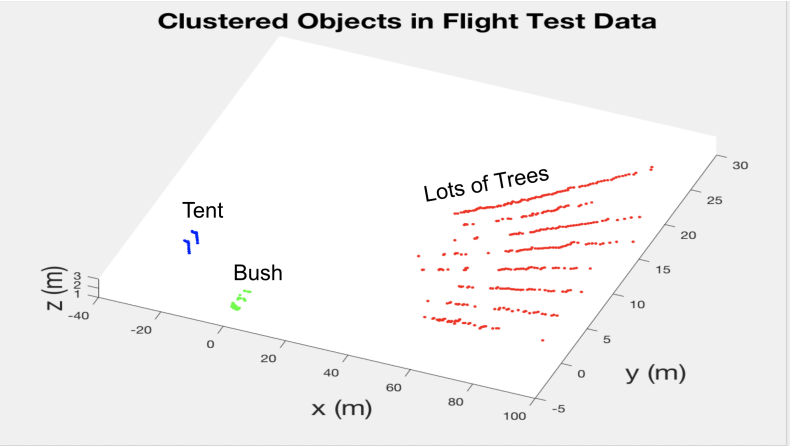


Figure 17: Ground Subtraction and Object Clustering of Flight Test Data

The tent and bush data were both separately processed by the Standard Hough Transform algorithm. An accumulator matrix with 100 discrete values for each dimension was used. The algorithm parameters and tolerances were tuned manually to produce two planes on the point cloud representing the tent. The result is shown in Figure 18. Since the LiDAR data was scanned in streaks, it was highly likely that the algorithm would detect planes parallel to these streaks. As seen from Figure 18, these planes would have laid on the xz plane. Therefore, the algorithm would remove detected planes if they were within a certain tolerance of being parallel to the xz plane. It can be seen that the detected planes captured nearly all of the points in this dataset, so this object was flagged as artificial.

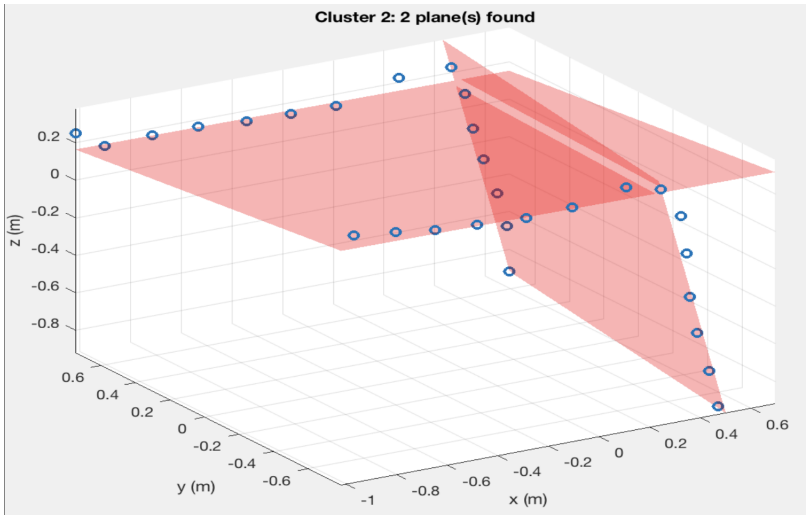


Figure 18: Standard Hough Transform Algorithm on Tent

The point cloud representing the bush was processed by the algorithm with the same parameters and tolerances as the tent. The output was one plane, as shown in Figure 19. It can be seen that this detected plane did not capture at least 75% of the points in this dataset, so this object was flagged as a natural object. The few points that lay on this plane were most likely coincidental.

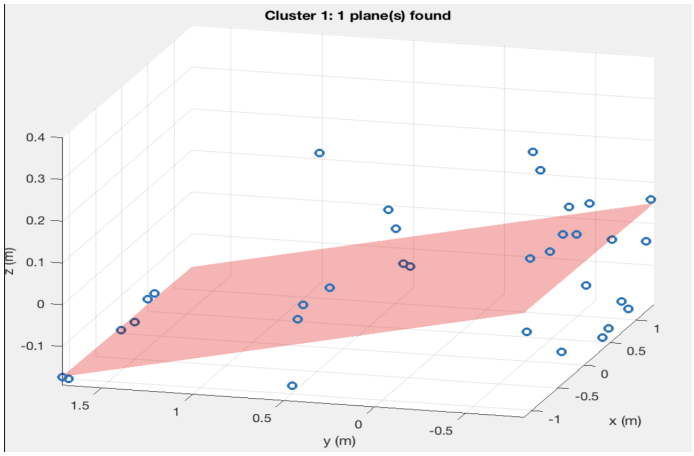


Figure 19: Standard Hough Transform Algorithm on Bush

VII. Conclusion

Implementing the Hough Transform on aerial scan data demonstrated reasonable potential for the use of LiDAR technology as a means of autonomously identifying artificial objects in natural settings. The algorithm was able to identify a tent from a reasonable altitude of 100ft, and correctly discarded a bush as natural. While the Hough transform would not be

adequate for sophisticated artificial object recognition, it verified the plausibility of using LiDAR to identify artificial objects in any setting. Furthermore, the experiment verified that existing LiDAR technology has the resolution necessary to gather useful data from a practical cruise altitude for multi-rotor UAV's.

VIII. Future Work

An alternate method known as the Randomized Hough Transform was in development, but is incomplete. This method drastically reduces the total computational time of the Hough Transform [1]. Instead of iterating through every single point and transforming them to Hough space, this algorithm randomly selects three points at a time and calculates the plane that they form using the Newton-Raphson numerical method [3]. The corresponding (θ, ϕ, ρ) accumulator cell is incremented, and the process is repeated continually. Once a certain accumulator cell reaches a minimum threshold, that plane is considered a detected plane, and the points associated with that plane are deleted from the point cloud. This process repeats until there are no more points left in the point cloud or until the algorithm has iterated many times without detecting a new plane. This algorithm would also automatically take care of the problem of duplicate planes as discussed earlier because points associated with detected planes are deleted.

Looking further into the future, efforts could be made to generalize the algorithm presented in this report to work on any given frame within any type of setting by implementing machine learning. The end goal would be to use LiDAR technology in real time scenarios to detect any type of artificial objects in aerial and terrestrial settings. Furthermore, vast data processing speed improvements would be necessary to make automated LiDAR scanning viable. Completion of the improved Hough method previously mentioned is one step in that direction. It is suspected that entirely new processing methods would be necessary to perform real-time object recognition with hardware mounted on the UAV.

IX. Acknowledgements

The authors wish to express their appreciation for the encouragement, advice and support received during their course of work provided by Christopher Lum, James Hermanson, Nancy Lee, James O'Neil, Hannah Rotta, Travis Clement, and Joshua Brockschmidt.

X. References

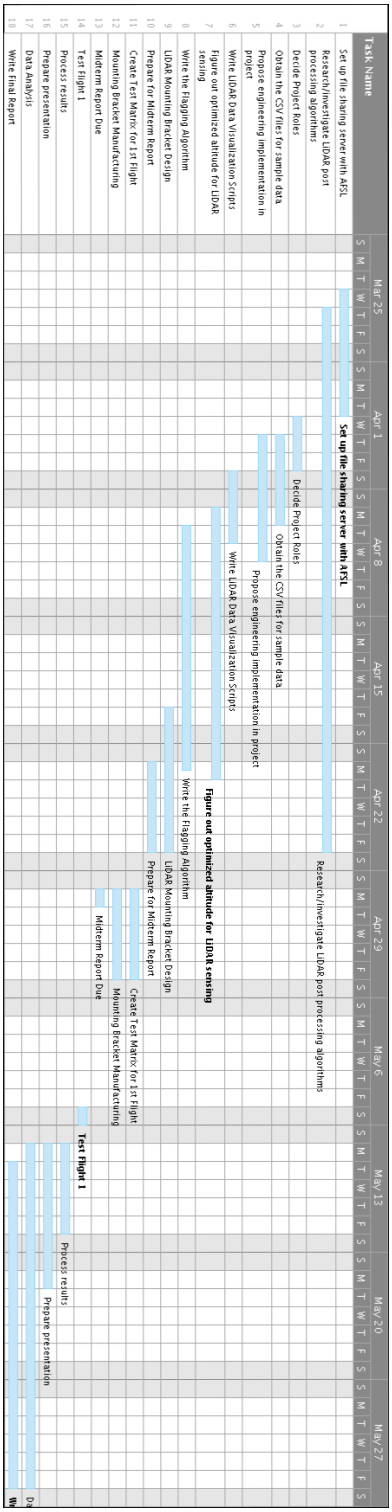
[1] Dorit Borrmann, Jan Elseberg, K. L. and Nuchter, A., "The 3D Hough Transform for Plane Detection in Point Clouds: A Review and a new Accumulator Design," 2011.

[2] Eisner, J., "State-of-the-Art Algorithms for Minimum Spanning Trees," 1997.

[3] Akram, S. and ul Ann, Q., "Newton Raphson Method," *International Journal of Scientific & Engineering Research*, Vol. 6, 2015.

XI. Appendices

Project Gantt Chart



Budget

Materials	**Cost**	**Needed to Purchase**
LiDAR system	$10,000	No
Transportation (1 trip)	$35 (gas)	Yes
Computer Software	$100	No
Drone	$1,200	No
Drone pilot	$120 ($30/hr)	Yes
Nuts and bolts	$10	Yes
Total Expenses: $165		