A QUANTITATIVE ANALYSIS OF THE FUSION OF 3-D SCANNING LIDAR SYSTEMS AND 2-D IMAGING SYSTEMS

THESIS

Michael F. Milton Jr., Captain, USAF
AFIT-ENG-MS-19-M-044

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-19-M-044

A QUANTITATIVE ANALYSIS OF THE FUSION OF 2-D IMAGING SYSTEMS AND 3-D SCANNING LIDAR SYSTEMS

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

In Partial Fulfillment of the Requirements for the

Degree of Master of Science in Electrical Engineering

Michael F. Milton Jr., BS

Captain, USAF

March 2019

AFIT-ENG-MS-19-M-044

A QUANTITATIVE ANALYSIS OF THE FUSION OF 2-D IMAGING SYSTEMS AND 3-D SCANNING LIDAR SYSTEMS

Michael F. Milton Jr., BS

Captain, USAF

Committee Membership:

Dr. Stephen C. Cain
Chair

Lt Col Scott J. Pierce, PhD
Member

Maj David J. Becker, PhD
Member

AFIT-ENG-MS-19-M-044

Abstract

This research will demonstrate the feasibility of fusing the superior spatial resolution of a 2-D imaging system with the precise range to target information of a 3-D imaging system to create a LIDAR imaging system that can accurately find what and where a target is. The 3-D imaging system will use a scanning method as opposed to a flash method that has been used in similar research. The goal of this research is to improve performance of scanning LIDAR so it has better spatial resolution. The research in this thesis proves that incorporating 2-D imaging data into 3-D scanning LIDAR data improves the spatial resolution of the LIDAR system, at least for simplistic environments. This idea is introduced to improve LIDAR systems for missile seekers. Incorporating this system in missile seekers will allow improved target tracking compared to a 3-D scanning system alone.

Acknowledgments

I would like to express my sincere appreciation to my faculty advisor, Dr. Stephen Cain, for his guidance and support throughout the course of this thesis effort. Your insight and experience were certainly appreciated and I would not be where I am today without you. I would, also, like to thank my committee members, Lt Col Scott Pierce and Maj David Becker. Your expertise was invaluable to my success.

Michael F. Milton, Jr.

Table of Contents

Page

List of Figures

Figure Page

List of Tables

Table Page

A QUANTITATIVE ANALYSIS OF THE FUSION OF 2-D IMAGING SYSTEMS AND 3-D SCANNING LIDAR SYSTEMS

I. Introduction

Background

In the world of target detection, there are a multitude of different options available with each one displaying its own advantages and disadvantages. One method that has been proven to be accurate with high resolution and high point density is LIDAR (Carter and others, 2012:2). LIDAR, or light detection and ranging, involves the use of laser pulses to determine the distance to a target. The target becomes illuminated by the laser pulses and the return time is measured for each pulse. This time, along with the wavelength of the laser pulse, creates a 3-D representation of the target (Richmond and Cain, 2010:1).

There are also different types of LIDAR systems. For the purposes of this study, the difference between a flash LIDAR and scanning LIDAR system will be discussed. In general, a flash LIDAR system uses a laser beam width that encompasses the entire surface area of the target. The advantage this method has over a scanning LIDAR system is that a flash LIDAR system can interrogate a scene much faster than a scanning system. An issue with this system is the pixel pitch. The pixel pitch is the distance between each pixel in an array (Dolce, 2011:1). The pixel pitch of a flash LIDAR system is at least 4

times larger than a scanning system. This causes spatial resolution problems.

Scanning LIDAR systems use a smaller laser beam that must be scanned across the desired target area. At each dwell area, the LIDAR system propagates a laser pulse to the target and back to the receiver and calculates the intensity and power received from the return pulse. With this information, along with the time it takes for the pulse to make its journey and wavelength of the pulse, the range to the target from that single pulse is found. The beam is then shifted and the process is repeated until the entire area is scanned.

One disadvantage of using a scanning LIDAR system is that it takes more time to scan across the target area than a flash lidar. If the target moves or the scanning system moves during this time, there could be errors in the range estimates. Also, atmospheric noise along with internal noise can cause errors as well. Experts state that "LiDAR pulses may be affected by heavy rains or low hanging clouds because of the effects of refraction" ("Advantages and Disadvantages of LiDAR," 2018). One solution to help remedy these range errors is the purpose of this research. The belief is that a 2-dimensional imaging system can provide additional position information that will improve any imperfections in the 3-dimensional LIDAR data. The Air Force uses LIDAR for various applications (Walsh, 2011). Knowing what and where a target is located is essential to the offensive and defensive realms of the military. Improving on these capabilities is constantly at the forefront of Air Force research. This research will show that an improvement on scanning LIDAR can be accomplished which will improve the spatial and range accuracy of scanning LIDAR systems.

Goals of Research

The main goal of this research is to show that the spatial and range accuracy of 3-D scanning LIDAR system data can be improved by fusing 2-D imaging data with it. This will be shown through simulated data and lab data.

Assumptions

The assumptions in this research are:

- The LIDAR pulse returns exist within the range gate of the system
- The LIDAR location is known in simulated data
- The target area and LIDAR system are stationary
- The light is not fully coherent so it can be modeled as linear and shift invariant

These assumptions are based on Captain Paul Dolce's thesis (Dolce, 2011:3).

Related Research

This section will describe various research strategies for fusing 2-D and 3-D images.

Research 1

A Statistical Approach to Fusing 2-D and 3-D LADAR Systems (Dolce, 2011)

Dolce's work focused on increasing the spatial resolution of a 3-D flash LIDAR system by implementing an algorithm using 2-D data in unison with the flash LIDAR data. Flash LIDAR systems are limited by the hardware which lends to the lack of spatial resolution. Expectation maximization, EM, is used to estimate the range and bias

associated with the 3-D system. Expectation maximization is explained in six steps. According to Capt Paul Dolce,

> The first step of the EM approach is to create a statistical model for the measure data, which is known as the incomplete data. Inventing a set of mythical data (complete data) and its relationship to the incomplete data is the second step. The third step is to select a statistical model for the complete data such that it adheres to the relationship of the complete to incomplete data. Next is to form a complete data log-likelihood. In step five, the conditional expectation of the complete log-likelihood is computed with respect to the incomplete data. The last step is to maximize the conditional expectation with respect to the parameter that is being estimated. (Dolce, 2011:18-19)

A comparison to other interpolation techniques is used to show how this algorithm is the most effective solution to increasing the spatial and range resolution of the flash LIDAR system. For results, Capt Dolce used the root mean square error (RMSE) and graphs to compare his proposed algorithm and other interpolation methods. Comparisons were made using two simulated targets and measured data. In each case, his proposed algorithm had the lowest range RMSE. For the first target, his proposed method had a 40% lower RMSE than the second lowest method. For the second target, there was a 103.825% lower RMSE. For the measured data, there was a 7.73% lower RMSE.

This work will be an extension to a previous research effort that used a flash LIDAR system by Capt Dolce. This research will differ because it will be using a scanning method instead of a flash method. The difference in a flash LIDAR and scanning LIDAR system is that a flash LIDAR system uses each laser pulse to illuminate the entire area being searched at once while a scanning LIDAR system uses a narrow laser pulse to search smaller area and then scans to the next area until the entire background has been scanned. In this thesis's research, a 2-D imaging system will "tell" the scanning system how the fire laser beam pulse is moving while the system is

scanning. This integration of data could make a scanning system have a pixel registration precision comparable to a flash LIDAR system. The 3-D system does not know where the beam is pointing in the scene, but the 2-D system can tell where the target really is. The 3-D system will get a range from the pixel where it believes the target is. This information could be inaccurate because of turbulence or the target moving while the system is scanning, but the 2-D camera does not have that problem. A flash LIDAR system does not have these same problems so the results will differ from Capt Dolce's work. The hypothesis is that adding the information from the 2-D imager will improve the accuracy of a scanning LIDAR system.

Research 2

Framework for 2D-3D image fusion of infrared thermography with preoperative MRI (Hoffman and others, 2017).

Hoffman's research fuses preoperative 3-D magnetic resonance imaging (MRI) with intraoperative 2-D infrared thermography. Infrared thermography can be used to differentiate healthy brain tissue from tumor tissue by correlating the emitted infrared radiated of the exposed cerebral cortex during neurosurgery with the temperature distribution of the brain. The problem with this method is that these images can be difficult to analyze because the results are not on the human visual spectrum. To solve this problem, fusion with a different type of imaging system is looked into. One of the key aspects that needs to be handled precisely for this fusion to work is to have a precise registration method. Image registration is the alignment of images in a way that the same

features in each image must be in the same spatial position, or simply, if an image from the first method is laid on the same image from a different imaging method, then the images need to line up as perfectly as possible. The two different types of image registration methods discussed are feature-based image registration and calibration-based image registration.

Feature-based registration uses either intensity-based features or shape-based features. Either way, extracting the features and solving how well they correspond is a complex and cumbersome problem that could take more time than is acceptable if real-time performance is needed. Calibration-based registration uses camera parameter estimation and camera tracking information that can create a robust coordinate transformation. Since there is no need to do any correspondence, the calibration-based registration is more robust therefore making it the desired image registration method.

After registration, the images are aligned and stacked. To accomplish this, the 2-D image needs to be translated, rotated, and scaled to fit the 3-D image. A projection step is also needed. Projection takes each pixel of the 2-D image and matches it to the 3-D voxel of the MRI image. Texture mapping is used to accomplish this. The results of this research indicated an accuracy of 2.46mm. The study of 2-D-3-D image fusion of infrared thermography with preoperative MRI is similar to the research in this thesis because it is a 2-D and 3-D fusion. This study is different than the research of this thesis because unlike using two different imaging modalities and fusing them together, the research in this thesis is using the existing 3-D data from a scanning LIDAR system and adding a 2-D sensor to observe the same modality. Then using that 2-D data to improve

the 3-D data. This method is more efficient because there are not two separate images, there is no need for a registration step.

Research 3

Texture Design and Draping in 2-D Images (Winnemöller and others, 2009)

This research is a look into creating a system to design and manipulate textures in 2-D images. The goal is to be able to allow artists to create, arrange, and manipulate textures in images without the need for 3-D modeling. 3-D modeling, while more accurate, is more complex and time consuming. This method will give close to the quality of 3-D modeling with the ease of 2-D imaging. This is accomplished by a method called texture-draping. This work differs from the work in this thesis by not using 3-D data at all, but instead using sketch-based shape-modeling. Shape-modeling involves artists designing the least complex normal fields that will allow the desired image-space effect. Artists will also be able to manipulate 2-D texture-coordinates.

Research 4

2-D-3-D Fusion for Layer Decomposition of Urban Facades (Li and others, 2011).

This conference paper presents a method for fusing 2-D images and 3-D scanned LIDAR data. To accomplish this, the 2-D images are registered with the 3-D data creating depth layers in the 2-D images. The images are then decomposed into rectangular planar fragments. The depth information from the LIDAR data is then diffused into the 2-D images by solving a multi-label assignment problem. Repetition

detection is used in each planar layer. Finally, a model of the 3-D image is produced that is enhanced, layered, and textured using their algorithm. The type of environment this research is focused on is urban building facades. The paper focuses on the fact that LIDAR data, while quick and easy, provides noisy, sparse, and sometimes completely missing data. Using separate 2-D photograph(s) decidedly increases the accuracy of the 3-D data.

Unlike the research for this thesis, the paper does not mention any quantitative results comparing the 3-D LIDAR scans alone with the fusion of the 3-D LIDAR scans and 2-D images. Rather, the paper focuses on the comparison of the rendered images from the LIDAR data and the fused data. While both the research in this conference paper and the research for this thesis use 2-D image data, the use of the 2-D data differs between the two. This thesis is focused around using the 2-D images to analyze the location of a laser beam on a target background while this paper uses 2-D images to fill in the sparsity of the 3-D LIDAR data.

Research 5

An Application of Markov Random Fields to Range Sensing (Diebel and Thrun, 2005).

This research paper discusses the application of Markov Random Fields (MRF) to generate high-resolution range images. They combine the low-resolution range images with the high-resolution camera images to create high-resolution range images. This is accomplished by using that fact that depth discontinuities in range system data often happen simultaneously with color or brightness changes in camera images. With this

knowledge, a multi-resolution MRF is used to integrate the range and image data by finding the mode of the probability distribution defined by the MRF.

The MRF takes two inputs, the image pixels and the range measurement. From these inputs the reconstructed range, image gradient, and depth discontinuity are found. To produce results, the authors used a sweeping laser range finder and a digital camera with 5 mega pixels per image. The outputs of these devices produced laser range measurements and camera images. Their results show improved detail and accuracy of different scenes when applying the MRF algorithm. Once again, there are no quantitative results in this report, only images to show that the MRF images "look" better than the original images.

Research 6

Deep Photo: Model-Based Photograph Enhancement and Viewing (Kopf and others, 2008).

The research from this paper discusses a way to improve outdoor photographs by combining them with 3-D digital terrain and urban models with image registration. With this registration comes depth, texture, and geographic information systems (GIS) data. Using this data, a multitude of operations can be performed on photographs to enhance them such as, dehazing, relighting, changing the view, and adding geographic information.

To register an image with a 3-D model at least four pair of points must be specified. Assuming the rough position from which of the photograph was taken is known, a model of the image can be rendered, while the parameters of the image an be

found by solving a nonlinear system of equations. Creating a geometric model of the image allows for a photograph to be enhanced by removing haze and color shifts. The viewpoint of the original photograph can also be changed with an accurate enough geometric model. If GIS data is present, then this data can be displayed and change dynamically as the image is changed.

This research is similar to this thesis by the way it is fusing 2-D and 3-D data to enhance the data. There are many differences however. In this paper's research, the 3-D data is used to improve the 2-D photographs while conversely, the goal of this thesis is to improve 3-D LIDAR data with 2-D image data. This paper also fails to provide quantitative results from the improvement of the photographs.

Research 7

Integrating Automated Range Registration with Multiview Geometry for the Photorealistic Modeling of Large-Scale Scenes (Stamos and others, 2008).

This research is devoted to creating a system that combines different registration techniques to produce photorealistic modeling of urban environments. This is accomplished by registering the 3-D range data to match 3-D features in the images. This creates a dense point cloud. Then the 2-D photographs are registered with the 3-D model. Finally, the 2-D photographs generate another 3-D model that is made up of the 3-D point cloud that is created from a sequence of 2-D photographs that are processed using a Multiview geometry algorithm. To finish this research, the author created an algorithm that can recover the rotation, scale, and translation to best align the two 3-D

models automatically. This allows the photographs to have the most accurate texture mapping onto the 3-D model.

Stamos' research is similar to what is presented in this thesis in the way 2-D images are used to improve a 3-D point cloud. The difference is displayed in how the 2-D images are integrated.

Research 8

Performance Characteristics of a Scanning Laser Imaging System Through Atmospheric Turbulence (Nairat and Voelz, 2012).

This research is based on determining the effects of atmospheric turbulence on scanning LIDAR systems. The focus is only on the illumination portion of the process or the "shooting" of the laser beam to a target. The study is focusing on the transverse, or angular, image resolution of a scanning system in the presence of atmospheric turbulence.

The research delves into the propagation of the beam truncated by the aperture at the source plane. The effects of the average beam profile in terms of the angular frequency spectrum are also analyzed. It is determined that for long-range imaging, if the beam size is sufficiently large, the angular beam divergence is limited by the atmospheric turbulence rather than the beam geometry. This affects the resolution accuracy and range accuracy of the system. The results of the research conclude that "resolution will be reduced by more than 90% in homogenous turbulence when the beam waist is on the order of the atmospheric coherence length (Fried parameter)" (Nairat and Voelz, 2012:5).

Nairat and Volelz's research is similar to this thesis in the study of scanning LIDAR. The work in this related research, however, is more of an investigative study on

how atmospheric turbulence affects scanning LIDAR systems while the research in this document is focused on the improvement of a scanning LIDAR system. Nairat and Volelz's research will be important in design considerations of a scanning LIDAR system, as well as performance estimates.

Thesis Organization

Chapter II will provide a description of the LIDAR model. Chapter III will delve into methodology of the research, including derivations of equations. Chapter IV will discuss the results from both the simulated and lab data. Chapter V will cover conclusions and future research.

II. Scanning LIDAR Model/Literature Review

Chapter Overview

The purpose of this chapter is to show how a scanning LIDAR system is modeled and to discuss the main components of the system.

Hardware

A scanning LIDAR is composed of many parts. Figure 1 shows a notional diagram of a scanning LIDAR system. According to ("How does LiDAR work?", 2019), the main sections of a scanning LIDAR system are:

- Laser source
- Scanner and optics
- Photodetector and receiver electronics

Global positioning systems and inertial measurement units are used in some cases, ("What is LIDAR?", 2018) but these components will not be discussed in this document as they are not essential to the research.

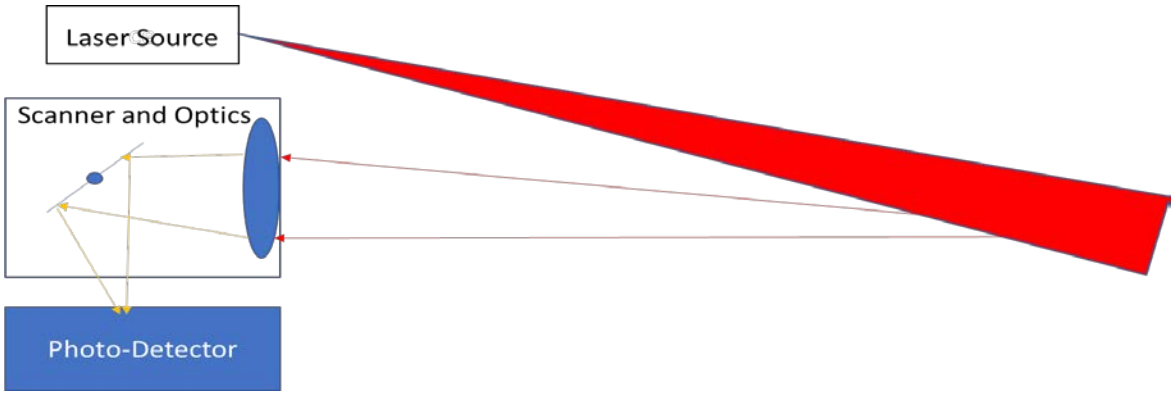


Figure 1: Scanning LIDAR system showing basic components and their operation

Laser Source.

LIDAR can use many different wavelengths of light for its laser pulses. Anywhere between 250nm and 2000 nm can be usable wavelengths based on the application needed. Shorter wavelengths give a higher resolution but longer wavelengths can be used at longer ranges. There are multiple pulse shapes that can be used to model a laser pulse. For this research, a Gaussian pulse model was chosen. Using a Gaussian pulse model, first the energy per pulse must be derived. The energy per laser pulse, in joules, can be found using Equation (1).

$$E_t = \frac{P_{avg}}{PRF} \tag{1}$$

In this equation, P_{avg} is the average laser power and PRF is the pulse repetition frequency of the laser. Once the energy per pulse is found, the instantaneous laser power, in watts, can be calculated as a function of time, t. Equation (2) shows how the energy is distributed in time.

$$P_t(t) = \frac{E_t}{\sigma_w \sqrt{2\pi}} e^{\frac{-t^2}{2\sigma_w{}^2}} \tag{2}$$

where σ_w is the width parameter of the Gaussian pulse shape in seconds. This model of beam propagation gives a beam diameter, D_b, shown in Equation (3).

$$D_b = \theta_t * R \tag{3}$$

where θ_t is the angular divergence of the beam, in radians, and R is the propagation distance in meters.

Scanner and Optics.

The scanning mechanism creates a consistent steam of laser pulses while the optics determine how the pulses are collected once reflected from the target area. A lens is used to focus the beam. If a focused beam or unfocused beam that is propagated to the far-field passes through an aperture with diameter D_t in meters, the beam width of said beam is proportional to the angular limit of resolution (diffraction-limited) for any optical system. When the Lens maker's equation is satisfied, Equation (4), the diffraction-limited beam size is achieved.

$$\frac{1}{f} = \frac{1}{d1} + \frac{1}{d2} \tag{4}$$

where f is the focal length, d1 is the distance from the object to the lens, and d2 is the distance from the lens to the image.

If the beam is collimated, the diffraction-limited beam spot will be at the same distance as the focal length of the lens. Another way to achieve a diffraction-limited beam spot is to propagate a collimated field a distance great enough to meet the far-field condition (Richmond and Cain, 2010:9). This condition is shown in Equation (5).

$$R_{ff} > \frac{2D_t^2}{\lambda} \tag{5}$$

where λ is the wavelength of light.

Focusing optics are used to focus the light returning from the target onto the detector array. These focusing optics can be modeled as a phase screen with Equation (6).

$$t_{lens}(w_p, s_q) = e^{\frac{-j\pi\left(w_p^2 + s_q^2\right)}{\lambda f_l}} \tag{6}$$

where (w_p, s_q) are coordinates in the receiving plane.

There are various diffraction effects of the optics which are accounted for by the point spread function or PSF. The PSF also accounts for the atmospheric turbulence experienced by the system. These effects produce an impulse response, $h_{tot.}$ This impulse response is a part of a linear shift-invariant system. Equation (7) describes how diffraction effects are incorporated into the LIDAR model. P_{det} is the 3-D LIDAR return predicted by geometric optics. The variable h_{tot} is the PSF of the system including optics and atmospheric diffraction effects.

$$P_{img}(m_1, n_1, t_k) = \sum_{m=1}^{N} \sum_{n=1}^{N} P_{det}(m, n, t_k)\, h_{tot}(m_1 - m, n_1 - n, t_k) \quad (7)$$

Photodetector and Receiver Electronics.

The photodetector detects the returning laser pulse and records it. There are two factors that drive the efficiency of the receiver: the optics transmission and the quantum efficiency of the detector. These factors affect the amount of signal power measured by the system. Optics transmission is the amount of energy that makes it to the detector from the total energy received by the receiver. This is displayed in fraction form and is usually highly efficient. This efficiency is part of the laser range equation that determines the signal power captured at the detector, as seen in Equation (8) (Richmond and Cain, 2010:14).

$$P_{det} = \frac{\tau_o \tau_a^2 D_R^2 \rho_t (dA) P_t}{R^2 \theta_R (\theta_t R)^2} \quad (8)$$

where:

- τ_o is the optics transmission
- τ_a is the atmospheric transmission

- D_R is the diameter of the aperture of the LIDAR receiver optics in meters
- ρ_t is the target surface reflectivity
- dA is the effective target surface area in meters
- P_t is the transmitted laser power in watts
- R is the range between the LIDAR system and the target
- θ_R is the target surface angular dispersion in steradians
- θ_t is the beamwidth of the LIDAR transmitter in radians

For a photodetector to work, the photons of light received from the returned pulse must be converted to electrons or more specifically photoelectrons. These photoelectrons produce a current that can be converted to a voltage. The photoelectric effect is used for this conversion. The photoelectric effect occurs when light strikes a material. When this occurs, energy is transferred from the photons of light to electrons. In many LIDAR systems, avalanche photodiodes (APDs) are used to increase the gain of this conversion. In standard photodetectors, single photons have a probability of producing a single photoelectron. With APDs, if one photoelectron is produced, a flood of photoelectrons are also produced increasing the gain greatly. The quantum efficiency of the detector determines how probable an avalanche of electrons will occur. The type of noise that is produced in this process of photon to photoelectron conversion is readout noise. A readout amplifier is used to measure the photoelectrons converted from the returned photons. As the amplifier measures the charge of the photoelectrons, the random scatter of the charge creates slight discrepancies in the charge reading. The measure of this scatter is the readout noise (“Understanding CCD Read Noise”,2018).

Summary

This chapter discussed the components that make up a scanning LIDAR system and touched on how this system operates. Three main hardware components of a scanning LIDAR system are the laser source, scanner and optics, and photodetector and receiver electronics. These components work together to create the laser pulse, propagate the laser pulse to a surface, and subsequently receive and process the returned waveform.

III. Research Methodology

Chapter Overview

This chapter explains the process involved in research of this thesis. Appendices A, B, and C contain the MATLAB code associated with this methodology.

Gaussian Beam

The Gaussian shape can be used to describe pulse shapes produced by laser illuminators (Richmond and Cain, 2010:31). The continuous spatial Gaussian function is shown below as g(x,y), where σ is the standard deviation and (x,y) are the spatial coordinates.

$$g(x,y) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{9}$$

The beam is then scaled to show that the beam carries 1 J of energy. Scanning LIDAR systems will build 3-D maps of scenes by steering a beam similar to the one shown in Figure 1, back and forth across the scene. Figure 2 shows the Gaussian beam shape.

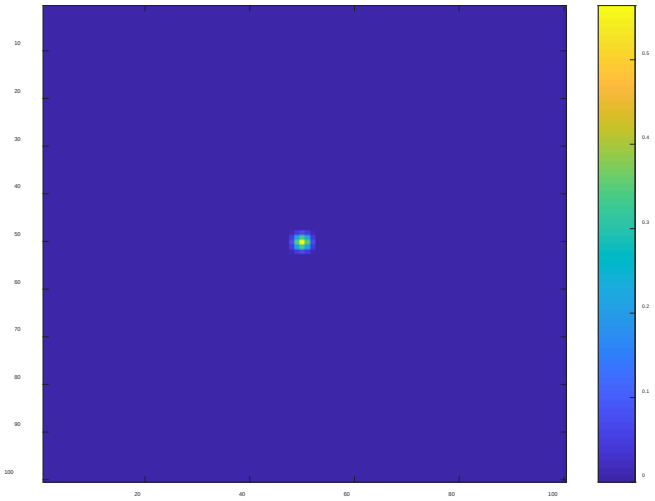


Figure 2: Spatial Gaussian beam shape

Scanning the Beam

Now that the beam and the target profile are created, the beam needs to scan across the target area. As the beam scans, random noise is added to simulate noise in the system. This noise interferes with the steady scanning motion of the beam and instead causes the beam to spatially not be where the LIDAR system expects it to be at each scan spot. This will become an important point later in the chapter. At each scanning position, the beam pulse is propagated down to the target area and reflected back to the receiver. The pulse energy distributed by diffraction is found by Equation (10),

$$E(t) = .001 * (g(x,y))^2 \tag{10}$$

where $g(x,y)$ is the beam power and it is multiplied by .001 to create 1mJ because the Gaussian in Equation (9) is normalized to have 1 watt of power so multiplying the pulse energy by .001 normalizes the power to 1mW.

Next, the power in the outgoing pulse at each range is found by applying Equation (2) and shifting it in time relative to the delay experienced by light as it travels from the laser source to the target and back,

$$P(t) = \frac{E(t)}{\sqrt{2\pi}\sigma_w} e^{\frac{-(t-\frac{2Z}{c})^2}{2\sigma_w^2}} \tag{11}$$

where σ_w is the width parameter of the Gaussian pulse shape in seconds, Z is the range to the target in meters, t is the time in seconds, and c is the speed of light.

The intensity of the target, I_t, is found by multiplying the atmospheric transmission by the pulse power.

$$I_t = \tau_{atm} * P(t) \tag{12}$$

where τ_{atm} is the atmospheric transmission or the loss of laser beam energy via absorption and scattering through the atmosphere (Richmond and Cain, 2010:10).

The reflected power, P_{ref} is determined to be the same value as the target intensity times the target area. Next, the intensity at the aperture, I_{rec} is found by,

$$I_{rec} = \frac{\tau_{atm} P_{ref}}{\theta_r Z^2} \tag{13}$$

where τ_{atm} is the atmospheric transmission, P_{ref} is the reflected power, θ_r is the reflection angle for Lambertian targets, and Z is the range from the target area.

The received signal power can be found by,

$$P_{rec} = \frac{\tau_{opt} a p_d^2 \pi I_{rec}}{4} \tag{14}$$

where τ_{opt} is the receiver optics transmission and ap_d is the diameter of the aperture.

Once the received signal power is found, it is then convoluted with the target profile to give the received signal power from every point in the target area at the correct range.

Target Profile

The target profile for the target area is created next. The target profile models the surface area of the target, which is range-dependent, multiplied by the range-dependent reflectivity. The target profile must be range-gated. If the range to the target is 10,000 m, the range gate must encompass that range, for example the minimum range can be 9990 m and the maximum range can be 10,010 m. This gives a range gate of 20 m. To convert this to a time sample, the range gates are multiplied by 2 times the speed of light. It is multiplied by 2 because the time accounts for the time for the pulse to get to the

target and reflect back to the receiver. The sample time for the range gate is found from the sampling frequency. If the sampling frequency is 500 MHz then the sample time is

$$\frac{1}{f_s} = \frac{1}{500\ MHz} = 2\ ns \tag{15}$$

Now the range of times in the range gate can be used by dividing the time gate by the sample time. The target reflectivity can be found if the surface type is known. It will most likely not be consistent throughout the target area. To create the target profile, the target area in Figure 3 and target reflectivity are combined. Equation (16) describes how the target profile helps to compute the power received by the LIDAR receiver, where T_p is the target profile. This equation shows how the target interacts with the pulse to produce the 3-D signal at the detector, P_{tot}.

$$P_{tot}(m,n,t_k) = \sum_{kk=1}^{N_s} P_{rec}(m,n,t_k - t_{kk}) T_p(m,n,t_{kk}) \tag{16}$$

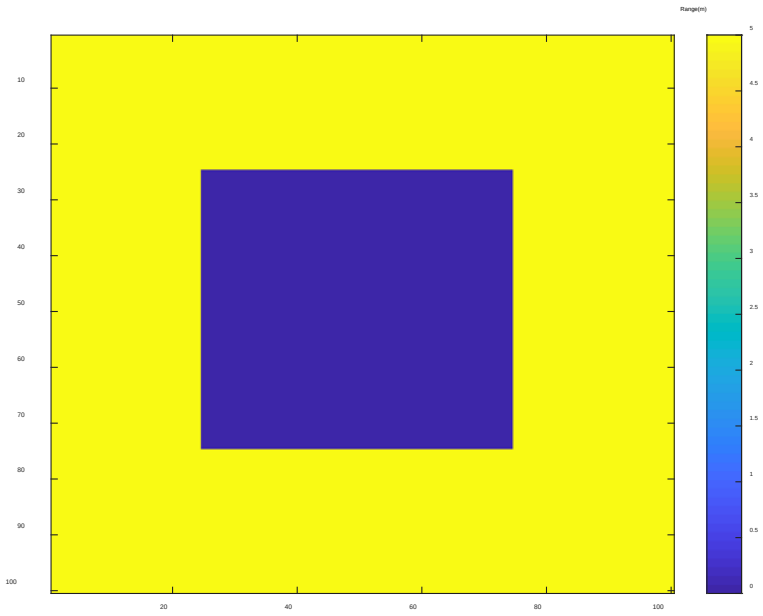
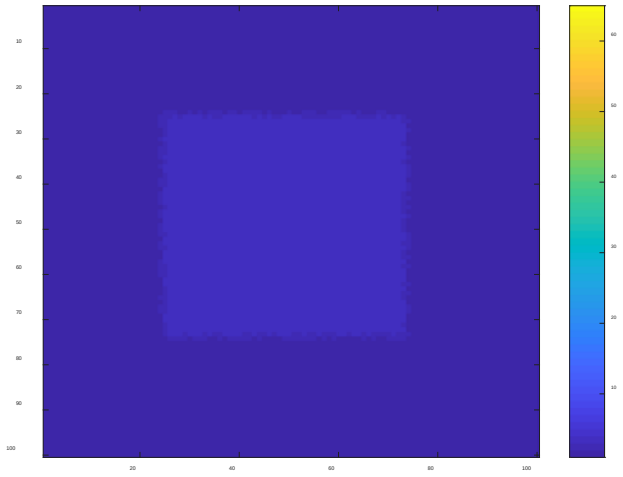


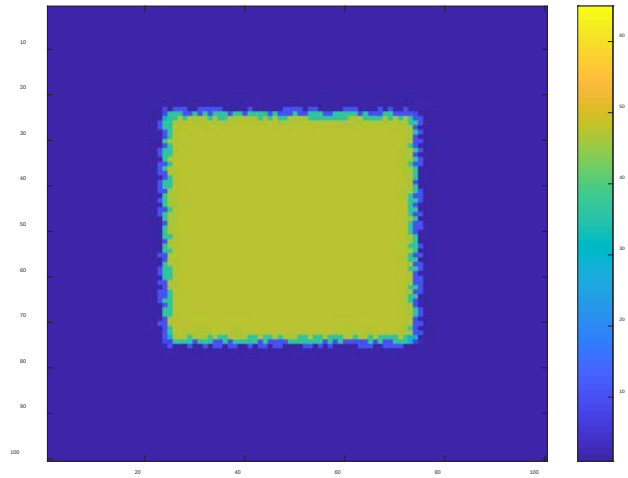
Figure 3: Example Target Area

Once again noise is added to the signal to simulate the noise a pulse would experience traveling through the atmosphere to the target area and back. This noise is simulated Gaussian white noise which is a close comparison to the real noise that would be experienced.

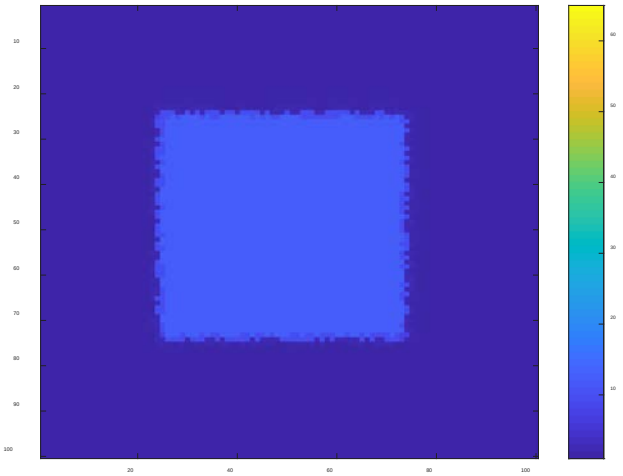
From this total received signal power, the waveform at each time can be found. Figure 4 shows the progression of the waveform as it travels down to the target area. One can see that the top of the building is the first area to be seen by the waveform. As the waveform progresses to the ground, the top of the building fades away and the ground is now illuminated.



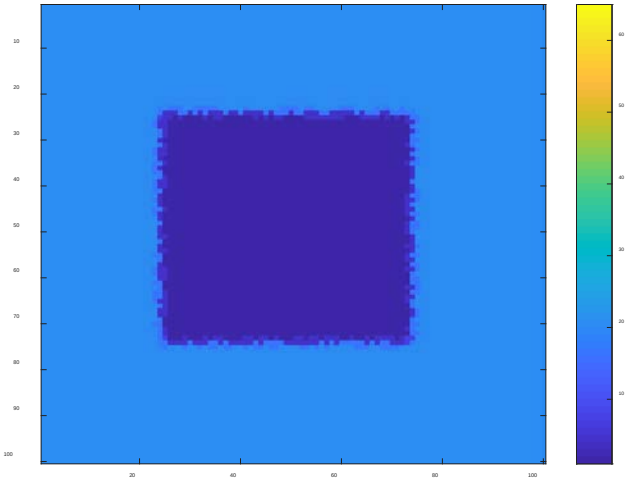
(a) 64 ns of waveform propagation



(b) 68 ns of waveform propagation



(c) 72 ns of waveform propagation



(d) 76 ns of waveform propagation

Figure 4: Images show how the beam and scene interact at different propagation times.

Adding Noise

Two types of noise were added to simulate real world situations. The first noise added was 2-dimensional uniformly distributed random numbers added to the Gaussian beam as it scanned across the array. This noise is used to create a non-uniform scanning motion. The second type of noise added is white Gaussian noise. This noise is added to the waveform as it propagates to the target. This noise affects the range estimate accuracy of the system. There are multiple sources of noise in a LIDAR system. These sources include statistical fluctuations in the light as it arrives at the detector, system noise, and unwanted photons (Richmond and Cain, 2010:15). More specifically, these can be categorized as photon counting noise, laser speckle, thermal noise, and background noise.

Photon counting noise: During the detector's finite integration time window, there is an expected number of photons to be counted. It is the nature of photons to arrive at random times. The number of photons that are counted during this window is a Poisson random variable. The variance of this noise is

$$\sigma_{pc}^2 = \frac{2q_e^2 B\eta P_{tot}}{h\nu} \tag{17}$$

where q_e is the elementary charge in coulombs, B is the bandwidth of the detector circuit, η is the quantum efficiency of the detector, h is Planck's constant and υ is the frequency of the laser light in Hertz.

Laser speckle: As the laser reflects off of a target surface, the electromagnetic field creates the laser speckle from the field's interference with a large collection of

independent coherent radiators. If modeled as a negative binomial random variable the variance of the measured photon counts, $\sigma^2_{speckle,}$ can be expressed as

$$\sigma^2_{speckle} = E\left[N_{signal}\right]\left[1 + \frac{E[N_{signal}]}{M}\right] \tag{18}$$

where M is the number of degrees of freedom of light. If M = 1, the light is fully coherent and if M goes to infinity the light is fully incoherent. $E[N_{signal}]$ is the expected number of photons. This variance encompasses both the speckle noise and the aforementioned photon count noise.

Thermal noise: Since the detector cannot reach 0 K, it will radiate some photons, generating noise. Equation (19) is used if the detector is accompanied by an A/D converter via a capacitor.

$$Q_n^2 = \frac{k_b TC}{q_e^2} \tag{19}$$

where k_b is Boltzmann's constant, T is the temperature of the circuit, and C is the capacitance of the circuit. The constant q_e is the fundamental electron charge.

Background noise: Any light that does not come from the LIDAR system's laser transmitter is considered background noise. Because of the poisson nature of the noise, the variance of the background noise is equal to the number of photoelectrons produced by the background. The mean number of photoelectrons produced by the background is given by

$$E[N_b] = \frac{S_{IB}\Delta_\lambda A_B \rho_t \eta \tau_a \tau_o D_R^2 \Delta t}{4R^2 h\nu} + E[N_{dark}] \tag{20}$$

where Nb is the number of photoelectrons produced by the background, S_{IB} is the intensity of the background light at the target in units of W/m^2 per μm of electromagnetic

bandwidth, A_B is the area of the photodetector projected at the target, Δt is the integration time of the detector, and N_{dark} is the dark current photoelectron count.

Creating the 2-D Imager

The 2-D imager can be created from the simulated 3-D data that was found above. To find the 2-D data, $I_{2\text{-}D}$, the total received signal power is summed in the 3rd or time dimension leaving a 2-D composite. Equation (21) shows this process.

$$I_{2D}(m_1, n_1) = \sum_k P_{tot}(m_1, n_1, t_k) \tag{21}$$

From this 2-D data the coordinates of each pixel can be found and from there the intensity of the waveform can be found. Searching for the maximum intensity for each waveform will determine the center of the Gaussian beam on the target area. With the beam's actual location, the estimated range can be found. This range is the range that the beam is actually located as opposed to where the LIDAR system believes the range is at. What makes these ranges vary is the fact that noise was added to the beam as it scanned and as it propagates. The LIDAR system believes that the beam is moving in a uniform pattern without accounting for the noise so the 2-D system is here to correct these errors. Figures 5 and 6 show how these errors express themselves in the data.

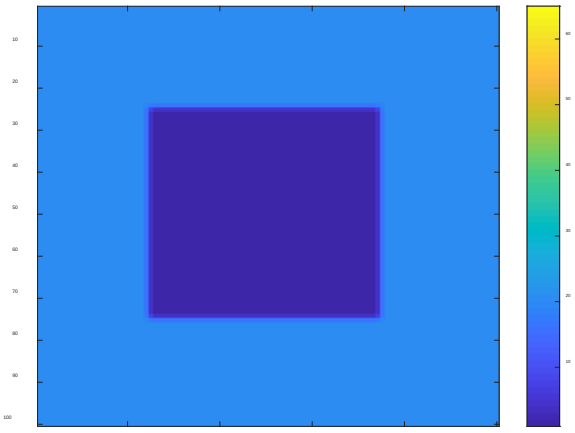


Figure 5: What the LIDAR believes it sees

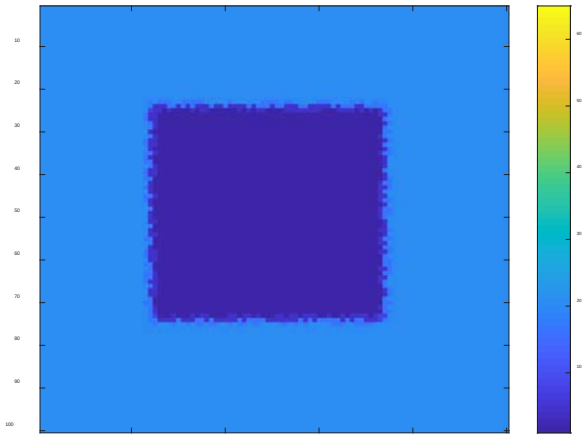


Figure 6: What the LIDAR actually sees with noise

Creating the Point Cloud

To check if the 2-D imager actually improves the LIDAR data, a scatter plot is used to visualize two plots; one plot not using the 2-D data and the other plot using the 2-D data. To quantitatively check the difference the root mean square error is used for both plots. The errors are the difference between the estimated range found from the 2-D imager and the actual range from the given target area. This difference is then squared and added to the difference of the previous pixel place. Once all of the errors are added together, the square root of that error divided by the number of pixels is then calculated. This is the root mean square error.

Lab Setup

Although not a true real-world experiment, this hardware-in-the-loop lab setup adds a more realistic version of this research. There are four main components that make up this experiment: the camera, the lens, the computer system, and MATLAB.

Camera.

The camera, shown in Figure 7, is a Thorlabs 8-megapixel monochrome scientific charge-coupled device (CCD) camera that is hermetically sealed and cooled. The CCD array is 3296x2472 pixels. For this experiment the exposure time was set at 5 ms, the gain to 120, the black level was 54, and the readout speed was 20 MHz.

Figure 7: Image of the Camera

Lens.

The lens, shown in Figure 8, used for the experiment is a KPX085 N-BK7 Precision Plano-Convex Lens with a focal length of 62.9 mm and a diameter of 25.4 mm.

Figure 8: Optics Lens

Computer System.

This component includes the CPU with two connected monitors. Figure 9 shows the setup. One monitor is up front and controls the MATLAB scripts while the other monitor is at the opposite end of the optics table. This second monitor is used to display the target area to the camera.

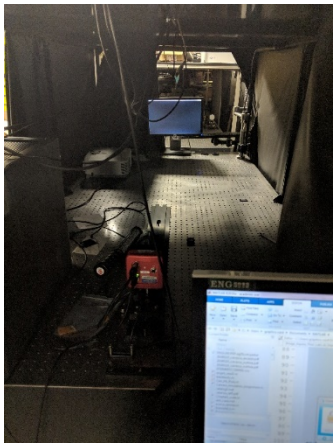


Figure 9: Computer setup

MATLAB.

The MATLAB code from the simulated data is slightly modified to account for the real-world aspect of the experiment. Specifically, the simulated range noise is removed from the code since real noise will be produced by the camera, the camera is taking a snapshot of the beam scanning across the target area at every beam dwell, and a conversion method must be produced to compare the array size of the target area with the array size of the camera's CCD array. The beam and target area are still simulated through the code. The simulated 2-D imager is replaced by the actual camera.

Lab Methodology

Figures 10, 11 and 12 depict the set-up of the lab experiment. The camera is 50.8 mm from the lens and the lens is approximately 2 m from the rear monitor. These distances satisfy the Lens maker's equation.

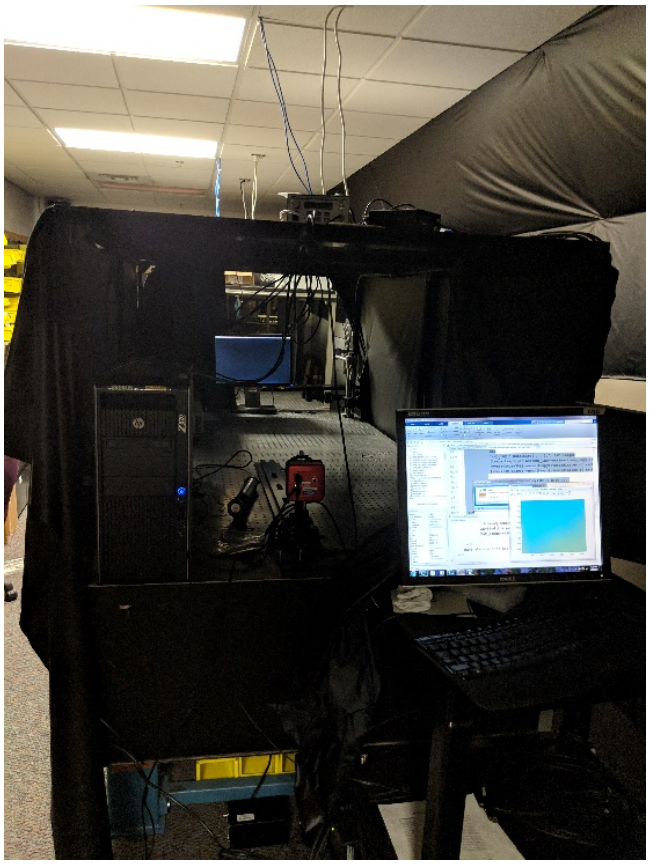


Figure 10: Front view

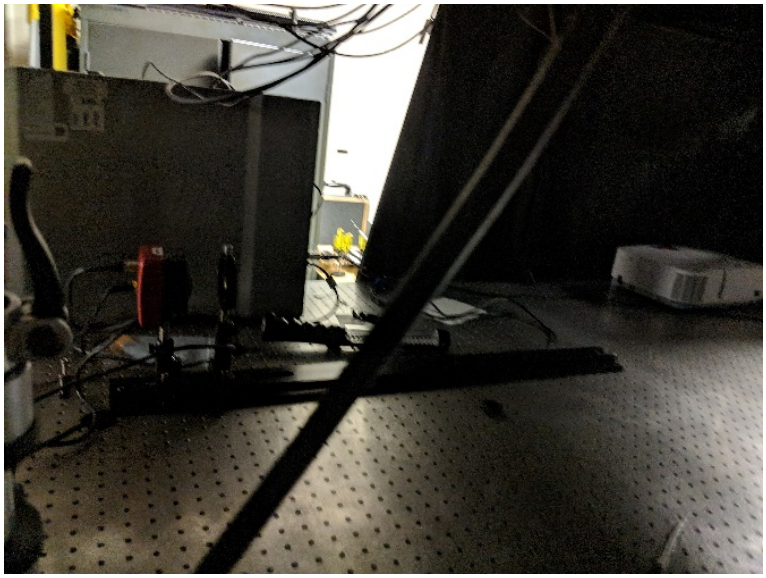


Figure 11: Side view

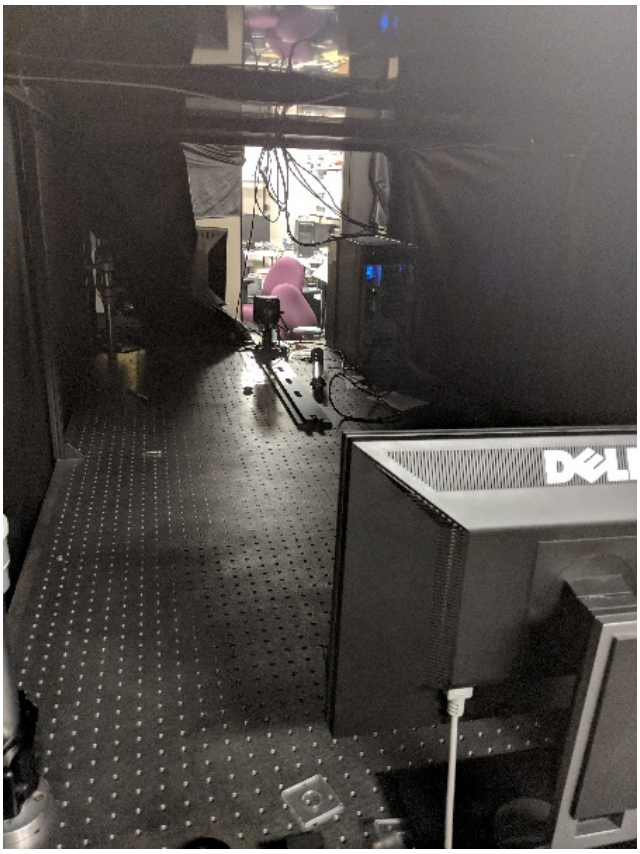


Figure 12: Rear view

The first step in performing this experiment is to integrate the camera with MATLAB. The camera has its own software, ThorLabs, that is used to set parameters and record images but to make sure that the camera and the code is synced perfectly, the camera is initialized as a video input in MATLAB. Once the camera is in MATLAB, various parameters can be adjusted. In this case, the region of interest was changed from the full 3296 x 2472 pixel array to a 1296 x 1972 pixel array. The entire CCD array is not needed because the camera only needs to view the monitor. At the camera's location, if the camera were to use the entire array the camera's view would contain the surrounding background as well as the monitor.

Once the camera is initialized, the same code used for the simulations is used for this experiment. The only difference is that a snapshot is taken every beam shift. This camera snapshot records the received signal power and places it in a 3-D variable that varies with time. Now, instead of using the received signal power to find the 2-D image coordinates, the snapshot image is used. The estimated range was found first using no error, shown in Figure 13. Figure 14 also shows the estimated range with error. Notice how the edges of the target area are noisy because of the beam error, mirroring the noise effect in the simulation.

In the simulated data, there was a one-to-one pixel conversion from the initial target environment and the estimated range. With this lab data, the pixels in the CCD array are not the same size as the target environment pixels. Therefore, a one-to-one comparison cannot be made when attempting to find the error.

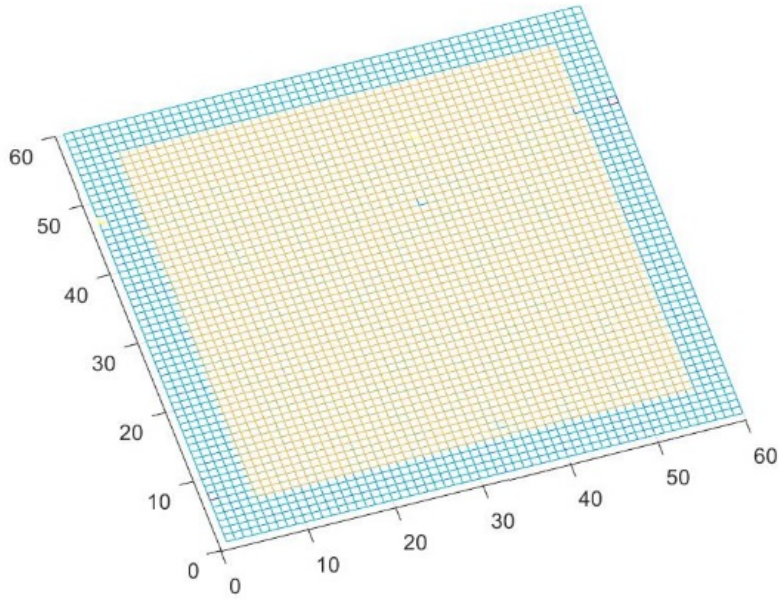


Figure 13: Estimated range with no noise

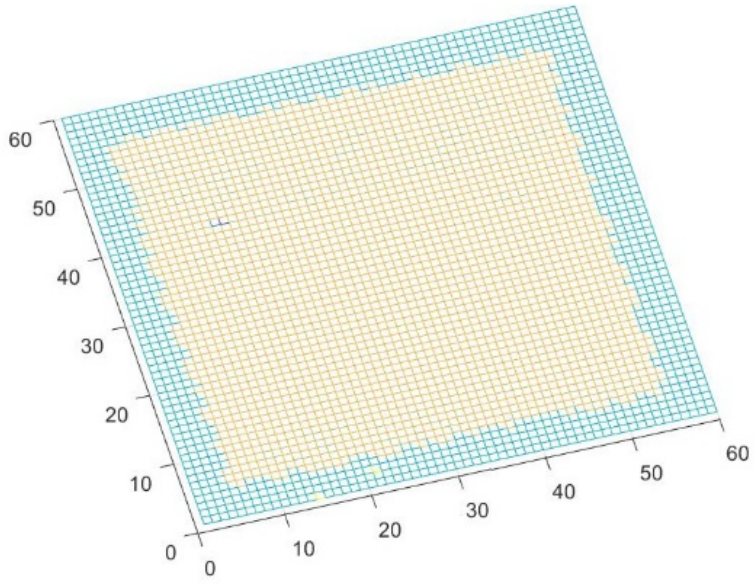


Figure 14: Estimated range with noise

Summary

This chapter discussed the methodology used to produce the results of this research, both simulated and lab tested. The first step was to create a simulated Gaussian beam pulse. This beam would be propagated to one area of the target and reflected back to be processed. The system would then shift the beam across the target, processing the returned waveform at each shift. Next, the target profile of the target area is modeled. The target profile multiplies the range-dependent surface area of the target and the range-dependent reflectivity. The target profile is then convolved with the returned pulse waveform to produce the 3-D signal. Simulated noise is added to the system to provide a more realistic waveform propagation and scanning movement.

The 2-D imager is created by summing the total received signal power. As the beam propagates and scans the scene, the maximum intensity of the scene is calculated.

This maximum intensity is the center of the beam on the scene. Finding the coordinates of this beam gives the system the actual coordinates of the beam that are not necessarily known from the LIDAR data alone. Creating a point cloud from the LIDAR data compared to the target area gives a qualitative view of the target area and also the root mean square error of this difference.

The optics lab was used to provide a more real-world example of this research. This lab utilized a camera and lens to take snapshots of a computer monitor that displayed the scanning beam. Because the camera has many more pixels in its array than the target area resolution, a conversion is needed to compare the pixel sizes. External forces that changed the camera's view of the rear monitor prevented the conversion from succeeding but qualitative plots of both datasets, Figure 13 and Figure 14, were shown that illustrate the improved quality of the LIDAR data fused with the 2-D imager data.

IV. Analysis and Results

Chapter Overview

This chapter will discuss the results of the fusion of 3-D scanning LIDAR data and 2-D imager data described in Chapter 3. A look will be taken at the 3-D data on its own and then compared to the combination of 3-D and 2-D data. Root mean square error (RMSE) will be used to determine if the fusion actually decreases the error a significant amount. Equation (22) shows the method of RMSE where $Range_{true}$ is the target area and $Range_{est}$ is the estimated ranges from the LIDAR data and 2-D data. N represents the number of pixels in each dimension. Both simulated data and lab data will be discussed.

$$RMSE = \sqrt{\frac{\Sigma_{x=1}^{N}\Sigma_{y=1}^{N}(Range_{est}(x,y)-Range_{true}(x,y))^2}{N^2}} \tag{22}$$

Results of Simulation Scenarios

The simulated data used a 100x100 resolution grid. First, a recreation of the target area was done using only 3-D data. Each pixel was plotted as the estimated range from that 3-D data. Figure 15 shows the results of that scatter plot.

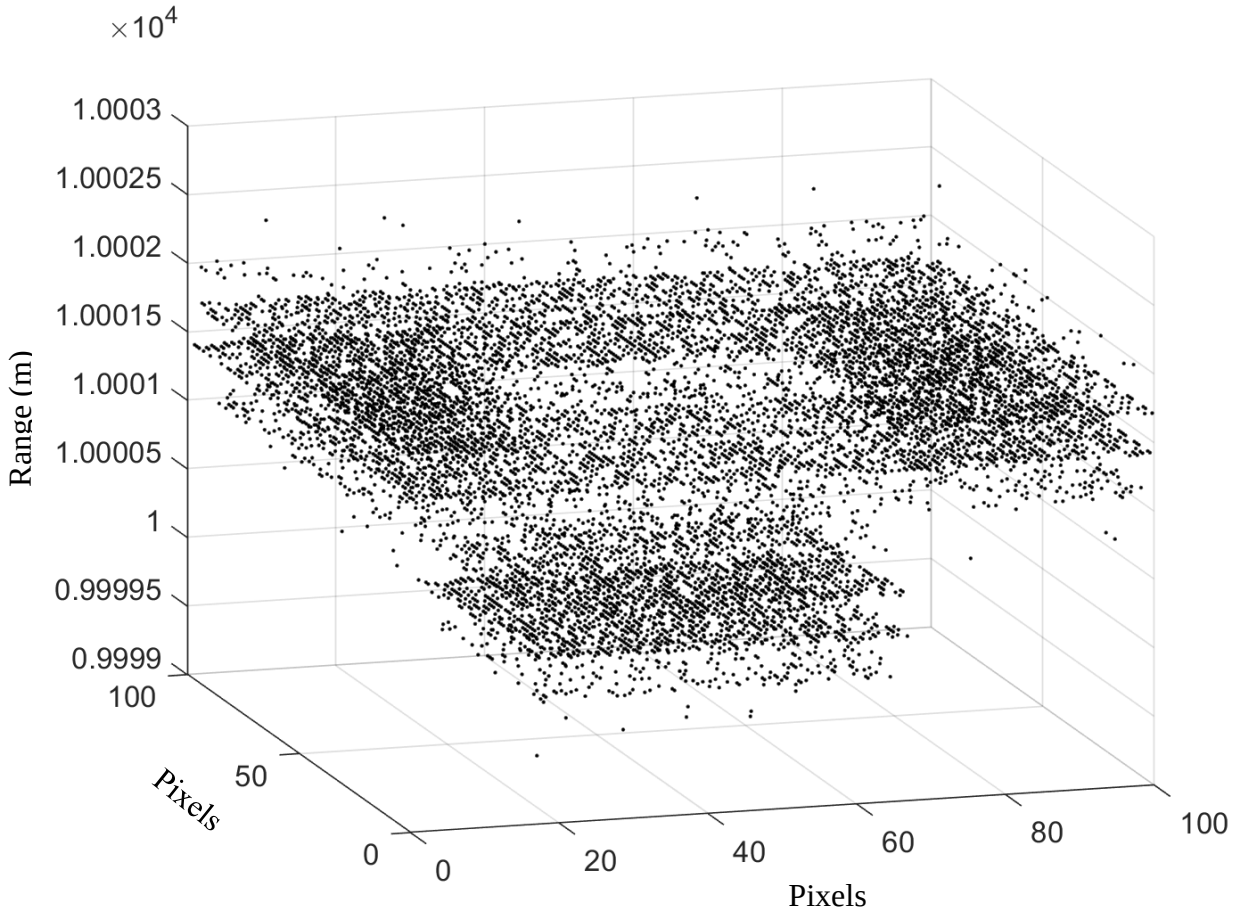


Figure 15: Scatter plot of 3-D data only

Next, the same scatter plot was performed using the estimated range using the 3-D data in union with the 2-D imager data. Figure 16 shows the results.

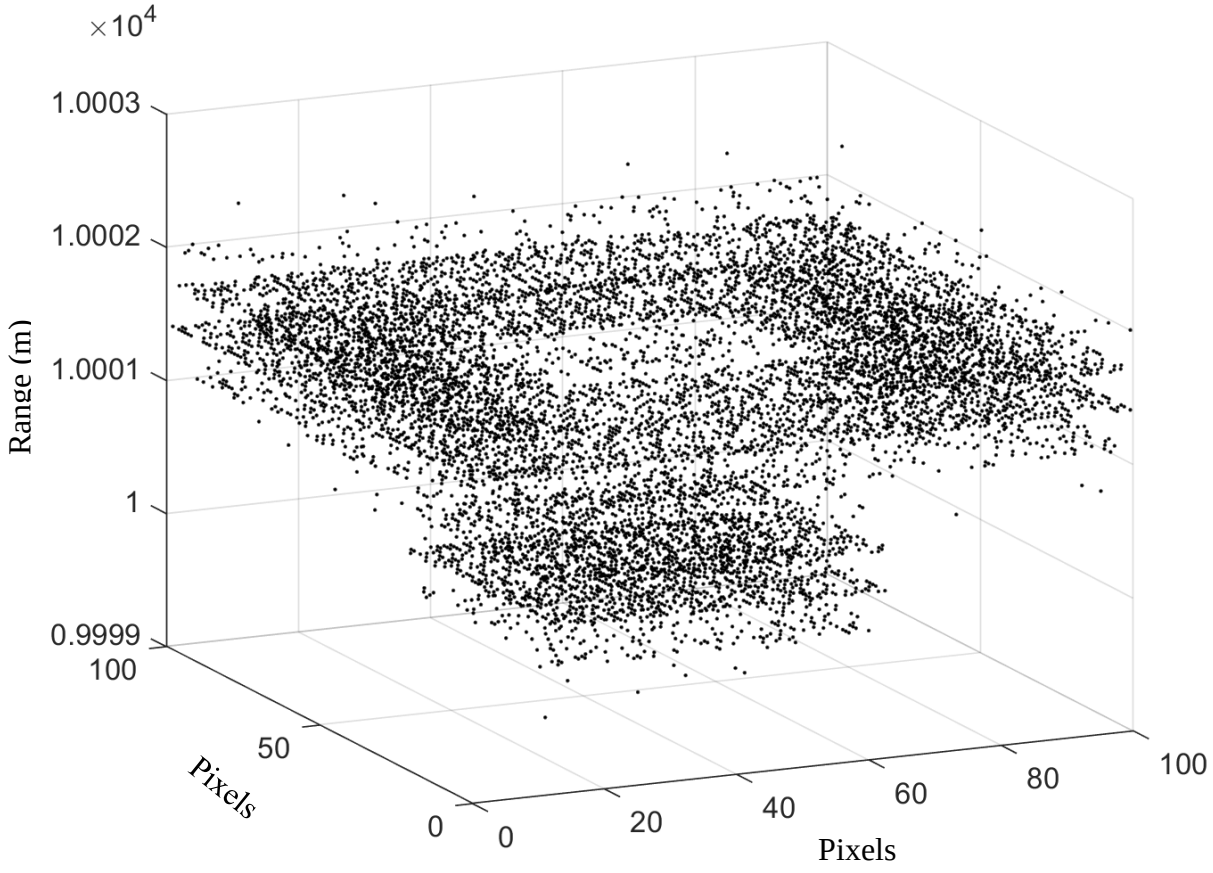


Figure 16: Scatter plot of 3-D/2-D data fusion

From Figure 15 and Figure 16, one can see that the 2-D data helps recreate a more similar image to the original target area. While difficult to spot in the complete scatter plot, the construction of the scatter plots shows the difference in the 3-D data and 2-D data. Figure 15 is constructed in a linear fashion, where each point is plotted as if the beam is scanning without noise, whereas Figure 16 is constructed by the true location of the beam. One can also notice how the area in the middle of the larger square is less dense in the fused data, Figure 16. With no noise, this area would have no data points, so a lower density of points means a greater accuracy.

To add some quantitative results, the RMSE is used. Multiple iterations were performed and averaged to produce an average RMSE for both criteria. Different pixel resolutions were also used to look into how the resolution affects the results. Table 1 shows the average of ten simulations of each resolution grid size.

Table 1: Ten trial average of RMSE of varying resolutions

Resolution size	Range RMSE of 3-D data only (m)	Range RMSE of 3-D and 2-D data fusion (m)	Difference (m)	Percentage Improvement in range error
100x100 grid resolution	.3065	.2664	.0401	+13.99%
80x80 grid resolution	.2935	.2538	.0397	+14.51%
60x60 grid resolution	.2927	.2359	.0568	+21.49%
40x40 grid resolution	.3034	.2079	.0955	+37.36%
20x20 grid resolution	.3718	.1577	.2141	+80.87%

This data shows that the addition of 2-D data adds a significant improvement to the accuracy of a 3-D scanning LIDAR system. An interesting note on this data is that one would expect that with a higher resolution, the error would decrease but, in fact, the opposite is the case. With the fusion algorithm used, the error with a 20x20 grid is significantly less than the error in the 100x100 grid. An explanation for this error increase is that with less data points available for comparison, there is less opportunity for error to accumulate. As more points are compared, the errors between those data points add to the total error.

To add more variation to the research, a more detailed target environment was used, shown in Figure 17.

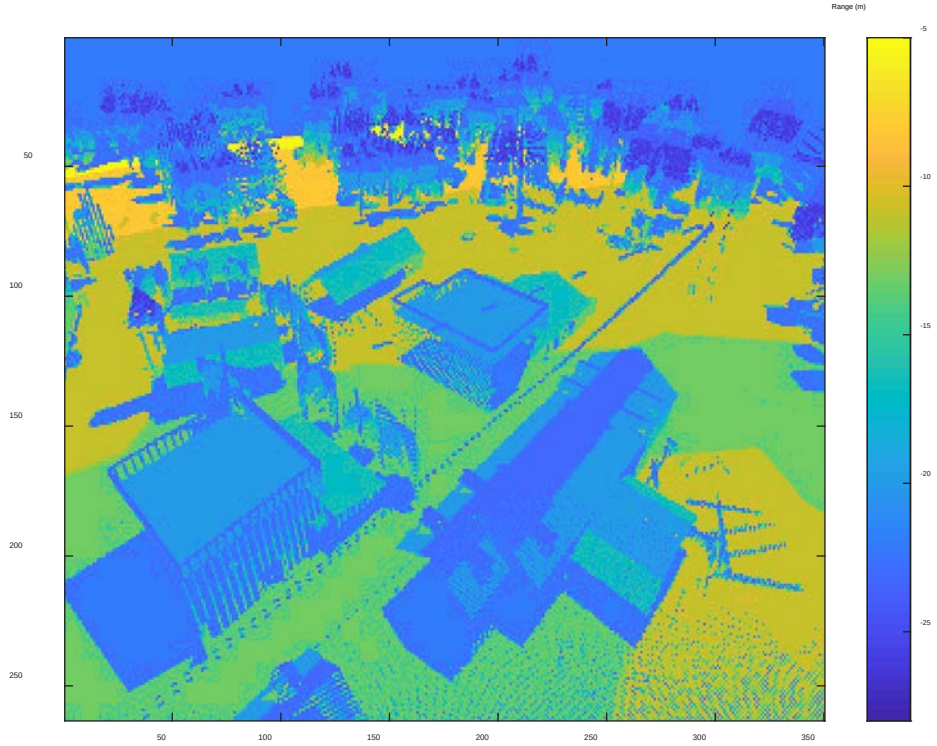


Figure 17: Detailed Target Environment

The same methodology was used in this example as the first. After one trial of this environment the RMSE errors were 13.895 meters for 3-D data only and 14.010 meters for the fusion. This result was surprising, showing that the error was high but also that there was no real difference between the two different methods. Running the entire dataset is extremely time and memory consuming with one run lasting approximately 100

hours. To curtail this issue, a 100x100 section of the environment was used to run multiple trials, shown in Figure 18.

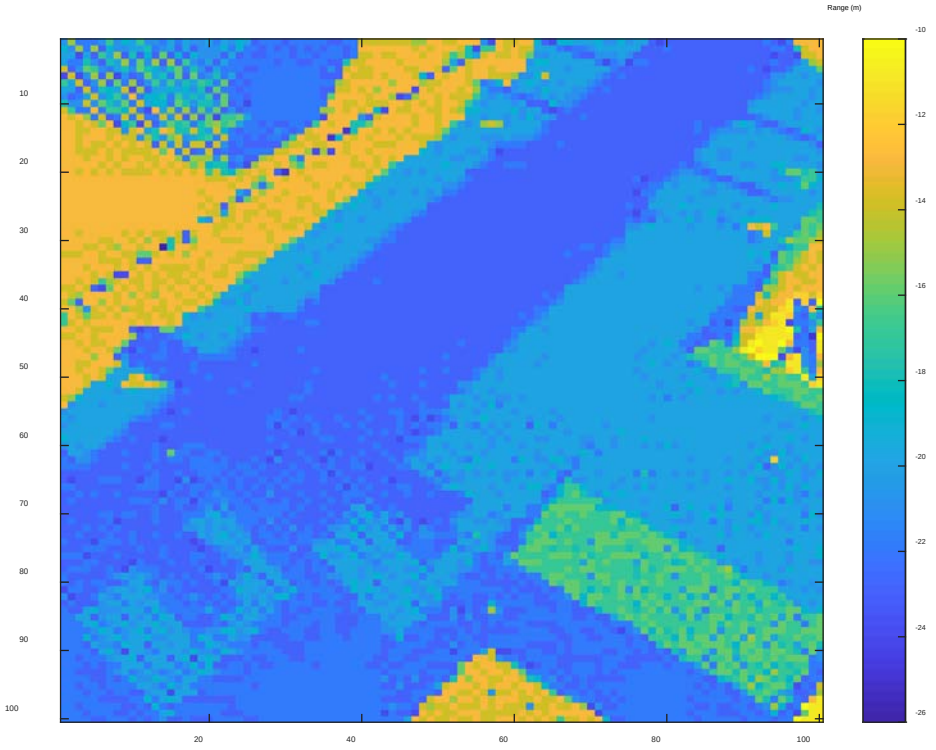


Figure 18: Cropped Target Environment

The results of these trials were similar to the entire dataset result. Even after removing the Gaussian white range noise, the results were still sub-optimal. Figures 19 and 20 shows the estimated range after the beam scans the entire scene and cropped scene, respectively. The figures show the large amount of noise present in this estimate.

Analysis suggests that the complexity and non-vertical point-of-view of the scene are the root causes of the error increase.

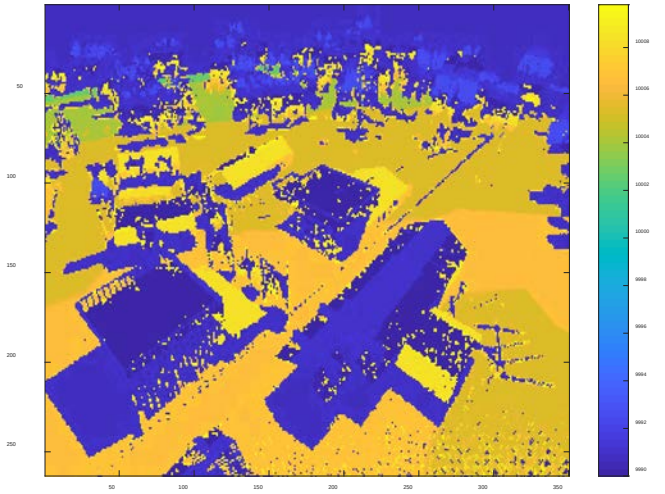


Figure 19: Entire 3-D scene

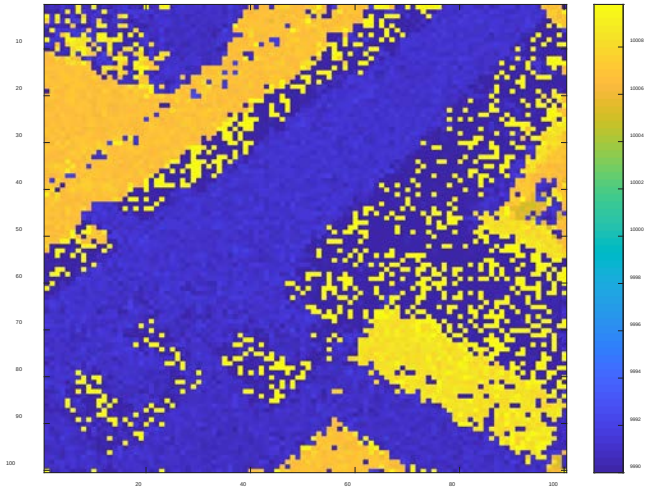


Figure 20: 100x100 section of 3-D scene

Results of Lab Data

A qualitative method was used to show the success of the fusion method in the lab data. A comparison between the initial target dataset with no 2-D correction and the dataset with the 2-D correction could not be accomplished quantitatively. This is because of registration issues between these images. To expand on this analysis, a collection of data took three days to complete. In that time, lights were turned on and off, the table that held the equipment was bumped, etc. Even slight changes in the test environment could cause a large change in the two collections. Therefore, when attempting to compare one set of data to the other, the result is not apples to apples. Figure 21 shows how the target area is slightly shifted from one dataset to the other. Even with only qualitative results, Figure 22 shows how the fused data corrected the noise in the

experiment and produced a cleaner image of the target area. The fused data was able to correct the error created by the unplanned error. Notice how even though the 3-D data and fused data were processed from the same dataset, the fused data produced an estimated range similar to the truth data while the 3-D data could not correct for the error.

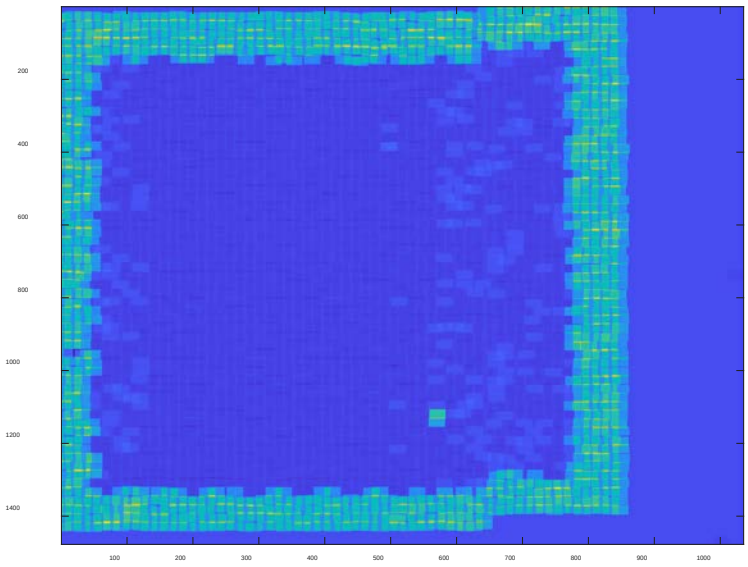


Figure 21: 3-D data only

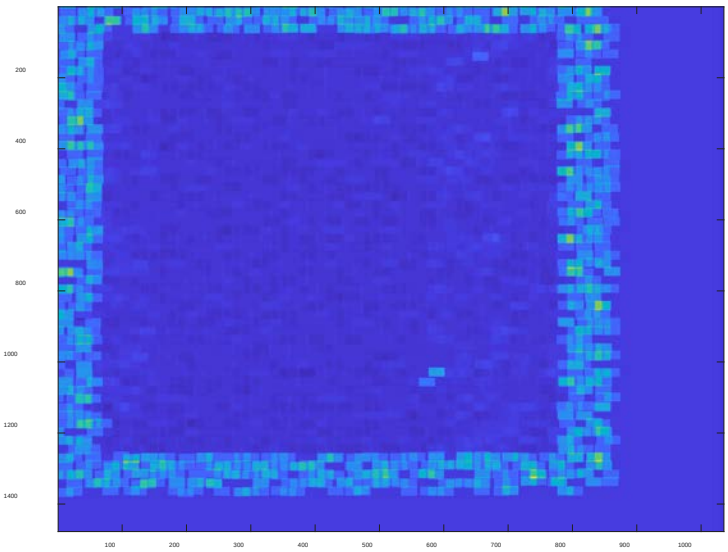


Figure 22: Fused data

Summary

This chapter discussed the results of the fusion of 3-D scanning LIDAR data and 2-D imager data. The main metric to determine the quality of data was the root mean square error. In the simulated data, five different resolution sizes with ten trials each all show that the fusion of the 3-D and 2-D data had a lower RMSE than the 3-D data alone, with an average of 8.92% lower RMSE. A more detailed target area was used to provide a higher complexity environment. The results of this detailed target area did not have the successful results as the simple target area. This is due to the complexity of the scene.

The lab data also proved to be difficult to quantify. External factors affected the integrity of the lab setup while data was being collected. These factors created registration issues within the data, making it impossible to compare the two datasets. Fortunately, image plots were produced that qualitatively show the improvement of the fused data over the 3-D data alone.

V. Conclusions and Recommendations

Chapter Overview

This section details conclusions drawn from this research and recommendations for future work that will add more robustness to this research.

Conclusions of Research

This research shows that the fusion of 3-D LIDAR data and 2-D imager data increases the accuracy of the data. Using the root mean square error provides quantitative evidence that this method is an improvement on scanning LIDAR alone.

Multiple resolution sizes and target areas were used in the simulations. Simple target areas proved to have the best results while more complex target areas prove difficult for the fusion to improve the error. The lab data showed, qualitatively, that this algorithm does provide accuracy improvements to scanning LIDAR systems.

Comparing these results with the results of Capt Dolce's research, his two simulated results displayed a range RMSE improvement of 40% and 103.825% for an average of 71.91% range improvement. The average range RMSE improvement from all five resolution sizes in this thesis was 33.60%.

Significance of Research

The benefits of increased LIDAR accuracy have significant impact on many areas of the Air Force and other military branches. LIDAR elevation data supports improved battlefield visualization, line-of-sight analysis and urban warfare planning. LIDAR can be used with a slew of different imaging platforms such as ISR, hyperspectral imagery,

and video (Walsh, 2011). Any increase of LIDAR accuracy is an increase in battlefield situational awareness.

Summary

This chapter discussed the conclusions that could be drawn from the results of this research and the significance of these conclusions.

VI. Future Work

Recommendations for Future Research

To further expand on this research, constructing an actual scanning LIDAR system would be beneficial. Although lab data is used, the laser system is still simulated because of lab limitations. If an actual laser system can be used to get real world data, that would only increase the validity of the research. Another research opportunity would be to expand the type of background and target varieties. Adding non-uniform targets of interest could produce interesting results. Also, as discussed in (Kashani and others, 2015:9), the amount of reflectance on a target surface can affect the effectiveness of a LIDAR system so varying target reflectance would be worth experimenting. Time constraints prevented a further look into the lab results. A continuation of the lab trials, without bumping the optics table, would prove beneficial in proving quantitatively that the fused data is an improvement over scanning LIDAR data alone. There are also opportunities to improve the algorithm to work with more complex environments with different attack angles.

Summary

This chapter discussed any recommendations for future research.

Appendix A: 20 x 20 Resolution Simulated Data MATLAB Code

Below is the MATLAB code for the initial simulation. A 20x20 resolution is used for time processing purposes but the premise of the code does not change with the resolution size. After each code section, the figures will be displayed. Any figures in a loop will be displayed in 4 time spots to show progression.

Creation of Gaussian Beam

```
stdevx=1;%standard deviation parameter in x direction
stdevy=1;%%standard deviation p arameter in y direction
sz=20; %standard deviation width of array
xx=-sz/2+1:sz/2; %creating the size of the arrary in the x coordinate
xx_mat=ones(sz,1)*xx; %creating an array of ones in the x coordinate
yy_mat=xx_mat'; %creating an array of ones in the y coordinate
beam=(1/(2*pi*stdevx*stdevy))*exp(-((yy_mat).^2)/(2*(stdevy^2))).*exp(-((xx_mat).^2)/(2*(stdevx^2)));
%Creating the gaussian beam using beam equation
beam=beam.*ones(sz,sz)/sqrt(sum(sum(beam.*beam))); %normalizing the beam
% figure(1)
% imagesc(beam) %displaying an image of the beam
% hcb=colorbar;
% set(get(hcb,'Title'),'String','Photon Intensity')
```

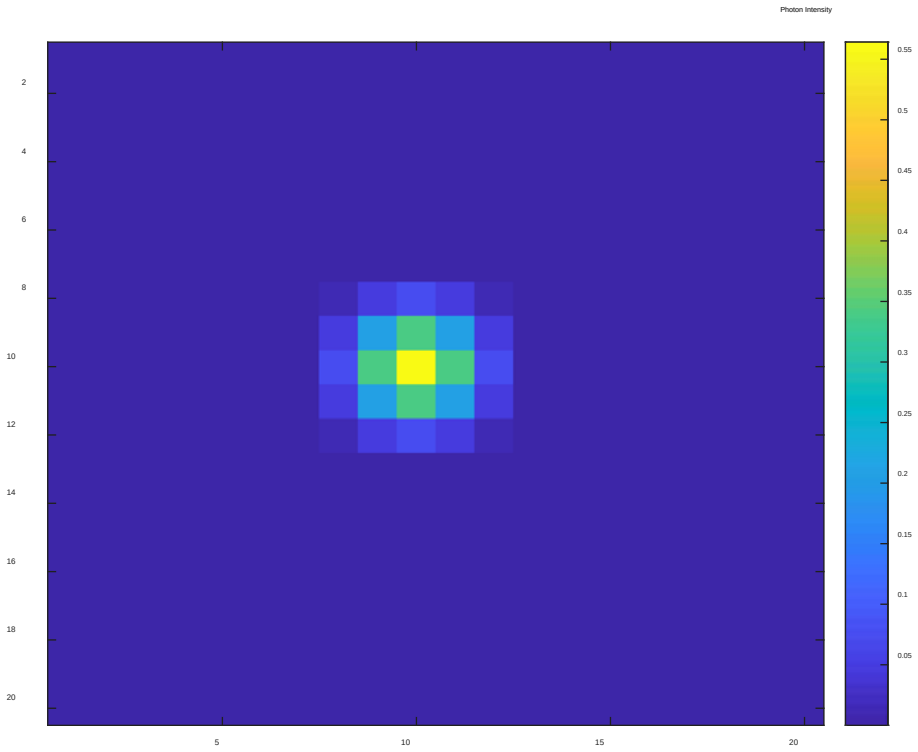


Figure 23: Figure 1 of MATLAB code, Gaussian Beam

Target profile

```
Sigma_w = 2e-9; % Pulse standard deviation in units of seconds
Rmin=9990; % Minimum range in the range gate
minT=Rmin*2/3e8; % first time that the receiver will measure the return
Rmax=10010; % Maximum range in the range gate
maxT=Rmax*2/3e8; % last time that the receiver will measure the return
deltat=Sigma_w; % Sample time in seconds.
t=minT:deltat:maxT; % Range of times in the range gate
target_area=ones(sz,sz)*5; % Define the area of the target at 10001.5 m
target_area(round(.25*sz):round(.75*sz)-1,round(.25*sz):round(.75*sz)-1)=zeros(round(sz/2),round(sz/2));% Define the area of the target at 10km
target_area_norm = (.3*target_area)+10000; %converts the target area coordinates to the same as the estimated range coordinates so that they can be compared
```

```
rho_t=ones(sz,sz)*0.1; % Target reflectivity at each pixel
T_p(sz,sz,:)=zeros(size(t)); %Creating the size of the target profile
for xn=1:sz  %loop to creating target profile
   for ym=1:sz
      T_p(ym,xn,:)=zeros(size(t)); % create a range vector per pixel
      indxx=target_area(ym,xn)+1; % Locate the range vector index
      T_p(ym,xn,indxx)=rho_t(ym,xn);% Assign a dirac based on target reflectivity and area of spatial sample
   end
end
% figure(2)
% imagesc(target_area) %display image of target area
%  hcb=colorbar;
%  set(get(hcb,'Title'),'String','Range(m)')
```

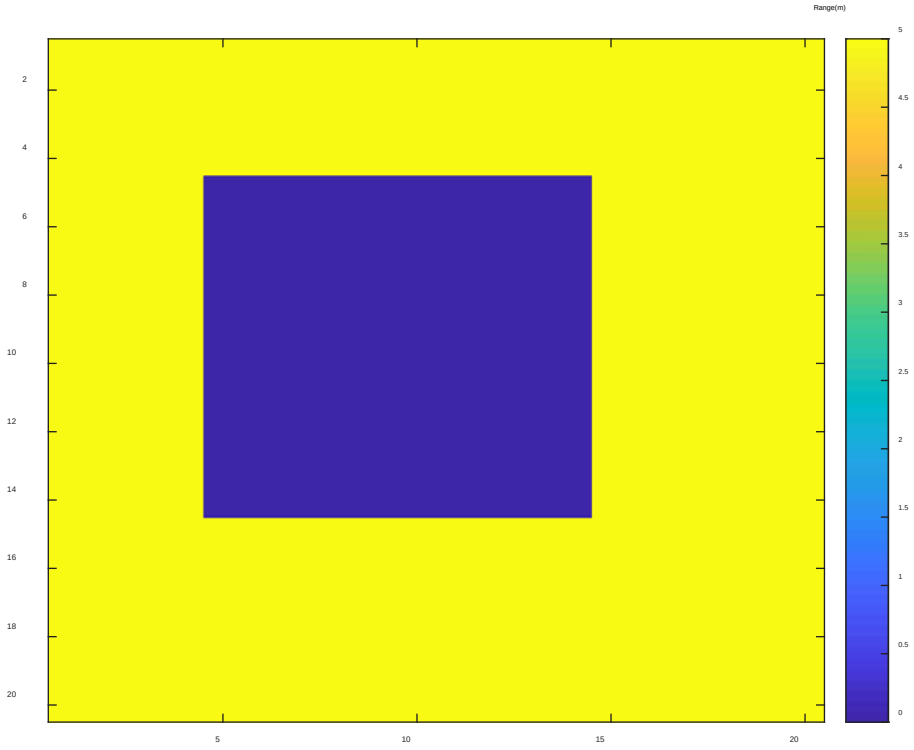


Figure 24: Figure 2 of MATLAB Code, Simulated Target Area

Creating scanning beam and receiver size creation

```
Z=10000; %distance to target
lam=1.55e-6; %wavelength of beam
tau_atm=1; % Atmospheric Transmission
tau_opt=1; % Receiver Optics Transmission
reciever_focal=1; % Focal length of the LIDAR receiver in meters
theta_r=pi; % Reflection angle for Lambertian targets
ap_diameter=.1; % Aperture diameter in units of meters
 for k = round(-sz/2)+1:round(sz/2) %loop to scan beam
    for m = round(-sz/2)+1:round(sz/2)

  S = circshift((beam),[round(k+rand) round(m+rand)]); %circshift shifts the beam and the rand commands randomly move the beam in a non linear direction
%    figure(3)
%    imagesc((1:sz),(1:sz),abs(S)) %displays the beam scanning across environment
%    hcb=colorbar;
%    set(get(hcb,'Title'),'String','Photon Intensity')
%    colormap
   pause(.1)
   E_t=.001*abs(S).^2; %1mJ pulse distributed by diffraction
   P_t=zeros(sz,sz,max(size(t))); % Allocate memory for 3-D pulse array.  P_t stands for power transmitted
      for tk=1:max(size(t)) % Setup loop to visit each time in the range gate
         P_t(:,:,tk)=(E_t/(sqrt(2*pi)*Sigma_w))*exp(-((t(tk)-Z*2/3e8).^2)/(2*Sigma_w^2)); % Images of the pulse at each range
      end
   I_target = tau_atm*P_t; %Intensity of target
   P_ref = I_target; % Reflected power from the target in units of Watts
   I_receiver=tau_atm*P_ref/(theta_r*Z^2);  % Intensity at the aperture
   P_rec = tau_opt*(ap_diameter^2)*pi*I_receiver/4; % Received signal power from a unit reflectance and area target at 1000 meters
   P_rec_tot=real(ifft(fft(P_rec,max(size(t)),3).*fft(T_p,max(size(t)),3),max(size(t)),3)); % Received signal power from every point in the target area at the correct range due to the convolution between the target profile and the waveform array.  The convolution is carried out using the convolution property of the Fourier transform.
%Function to add AWGN to a given signal
%Authored by Mathuranathan Viswanathan
%How to generate AWGN noise in Matlab/Octave by Mathuranathan Viswanathan
```

```
%is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0  International License.
%You must credit the author in your work if you remix, tweak, and build upon the work below
   SNR_dB = 10; % signal to noise ratio in DB, used to determine how much noise you want.  The higher the
SNR the lower the noise
   L=length(t); %the length of the time slices of the waveform
   SNR = 10^(SNR_dB/10); %SNR to linear scale
   Esym=sum(abs(P_rec_tot).^2)/(L); %Calculate actual symbol energy
   N0=Esym/SNR; %Find the noise spectral density
   if(isreal(P_rec_tot))
      noiseSigma = sqrt(N0);%Standard deviation for AWGN Noise when x is real
      n = noiseSigma.*randn(1,sz,L);%computed noise
   else
      noiseSigma=sqrt(N0/2);%Standard deviation for AWGN Noise when x is complex
      n = noiseSigma.*(randn(1,sz,L)+1i.*randn(1,sz,L));%computed noise
  end
   P_rec_tot_noisy = P_rec_tot + n; %received signal

   R_vec=t*3e8/2; %the range vector of the waveform

   for indxx=1:max(size(t)) %loop to create the waveforms from the power received at the aperture
      waveform1(k+round(sz/2),m+round(sz/2),indxx)=sum(sum(P_rec_tot_noisy(:,:,indxx)));
   end
      temp_img = sum(P_rec_tot_noisy(:,:,:),3);%2-D imager
%         figure(4)
%         imagesc(temp_img)
%         colorbar
      pause(.1)
       [yyc(k+round(sz/2),m+round(sz/2)),xxc(k+round(sz/2),m+round(sz/2))] =
find(temp_img==max(max(temp_img)));%coordinates of each pixel based on the 2-D imager
      idata=squeeze(waveform1(k+round(sz/2),m+round(sz/2),:)); %the intensity of the waveform
      Xx=find(idata==max(idata)); %the max intensity of the waveform, should be the middle of the beam.
This is how the 2-D imager knows where the beam truly is.

      Est_range(k+round(sz/2),m+round(sz/2))=mean(R_vec(Xx)); %This is the range that is calculated from
the range vector at the max intensity of the waveform
%         figure(5)
%         imagesc(Est_range)
%         colorbar
%         hcb=colorbar;
```

```
%          set(get(hcb,'Title'),'String','Range(m)')
        pause(.1)
    end
end
```

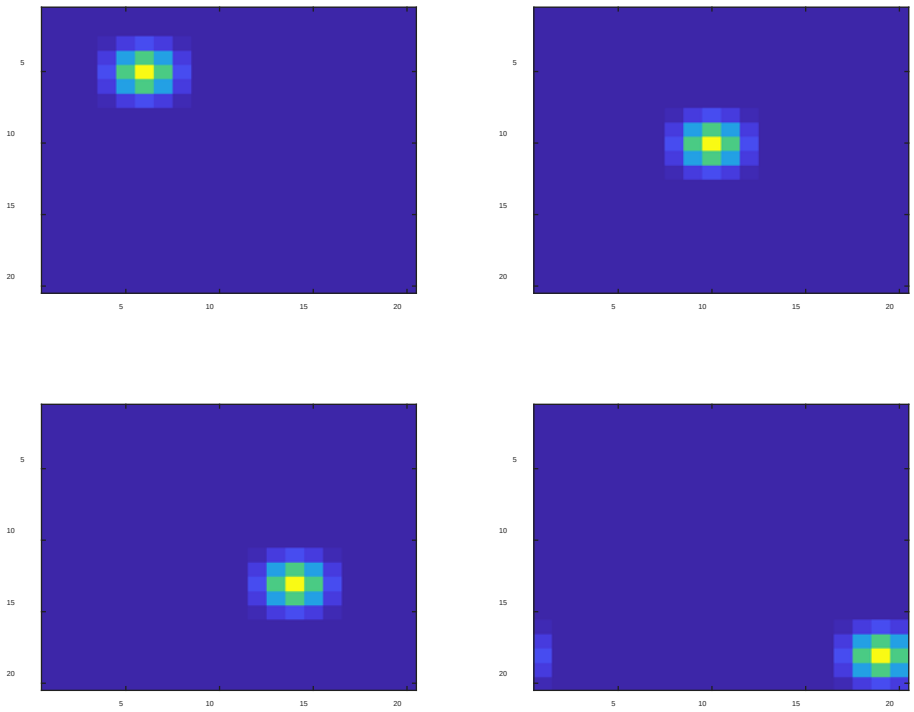


Figure 25: Figure 3 in MATLAB Code, Scanning Beam Time Progression

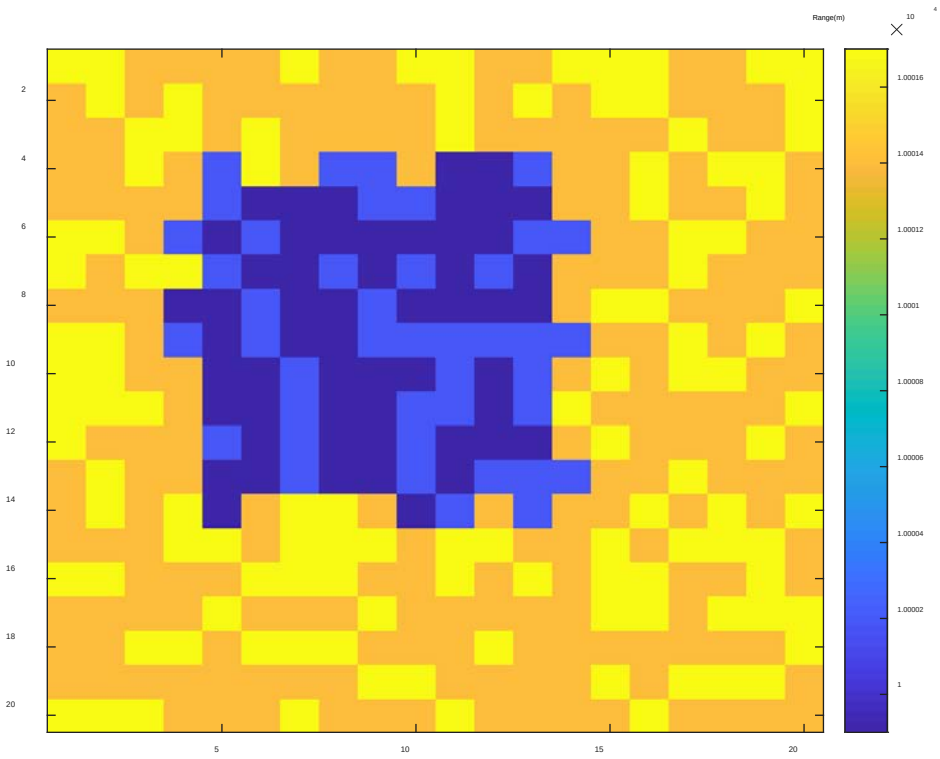


Figure 26: Figure 5 in MATLAB Code, Estimated Range from scanning LIDAR

Figure 4 in the MATLAB code, is not displayed because it is identical to Figure 3 in the MATLAB code (Figure 24) except for the value of the z-axis. Even though Figure 5 in the MATLAB code (Figure 25) is in a loop, the plot does not change enough to justify adding multiple plots.

Plotting waveforms

```
 wvfrm1=zeros(1,67); %creating a zero array for the waveforms in time
for frms=1:67 %loop to show the entire target area in time.  First the waveform hits the top of the building and then the ground.
% figure(6)
% image((10^8)*waveform1(:,:,frms)+1e-12); %image of waveform progression in time/range through target area
% colormap(gray)
% hcb=colorbar;
% set(get(hcb,'Title'),'String','Photons')
pause(.1)

 wvfrm1(frms)=sum(sum(P_rec_tot_noisy(:,:,frms))); %the waveform is the sum of the received power
%  figure(7)
%  plot(t,wvfrm1) %plot of the waveform intensity as it progresses through time
%  hcb=colorbar;
%  set(get(hcb,'Title'),'String','Intensity')
 pause(.1)
end
```

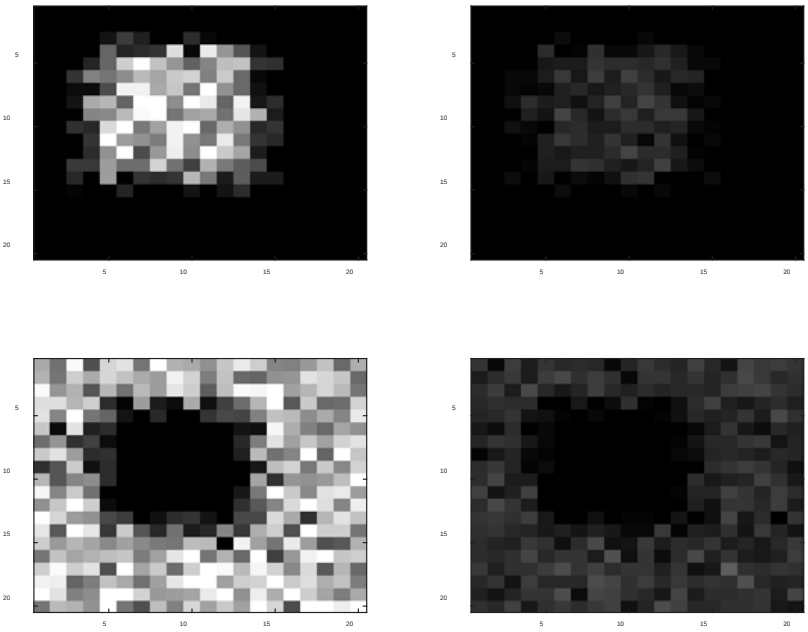


Figure 27: Figure 6 in MATLAB Code, Waveform propagation.
Top left: 68 ns, top right: 72 ns, bottom left: 78 ns, bottom right: 82 ns.

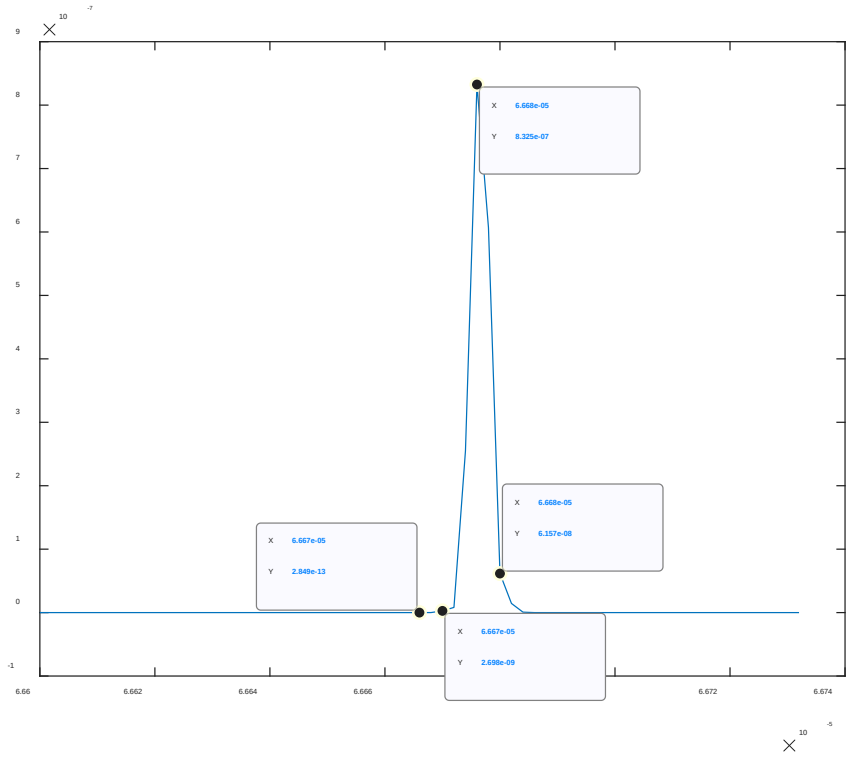


Figure 28: Figure 7 in MATLAB Code, Plot of Waveform. Each marker corresponds to Figure 27 plots.

Create point cloud

```
err1 = 0; %initializing the first error
err2 = 0; %initializing the second error
for p = 1:sz %loop to create scatter plot of estimated range
  for q = 1:sz
%       figure(8)
%       h = scatter3(xxc(q,p),yyc(q,p),Est_range(q,p),10,'k','.'); %Change xxc and yyc to q  and p
respectively to get noisy image. invert image by taking 10,000 minus Est_range,
%       pause(.1)
%       hold on
      err1 = err1 + ((Est_range(q,p)-target_area_norm(yyc(q,p),xxc(q,p))).^2); %Fused data error between
the estimated range and the actual range of the target area
      pause(.1)
  end
end
rmse1 = sqrt(err1/(sz^2)); %the root mean square error of the ranges.
for p = 1:sz
   for q = 1:sz
%       figure(9)
%       g = scatter3(p,q,Est_range(q,p),10,'k','.'); %Change xxc and yyc to q  and p respectively to get noisy
image. invert image by taking 10,000 minus Est_range,
%       pause(.1)
%       hold on
      err2 = err2 + ((Est_range(q,p)-target_area_norm(q,p)).^2); %3-D data only error between the estimated
range and the actual range of the target area
      pause(.1)
  end
end
rmse2 = sqrt(err2/(sz^2));
```

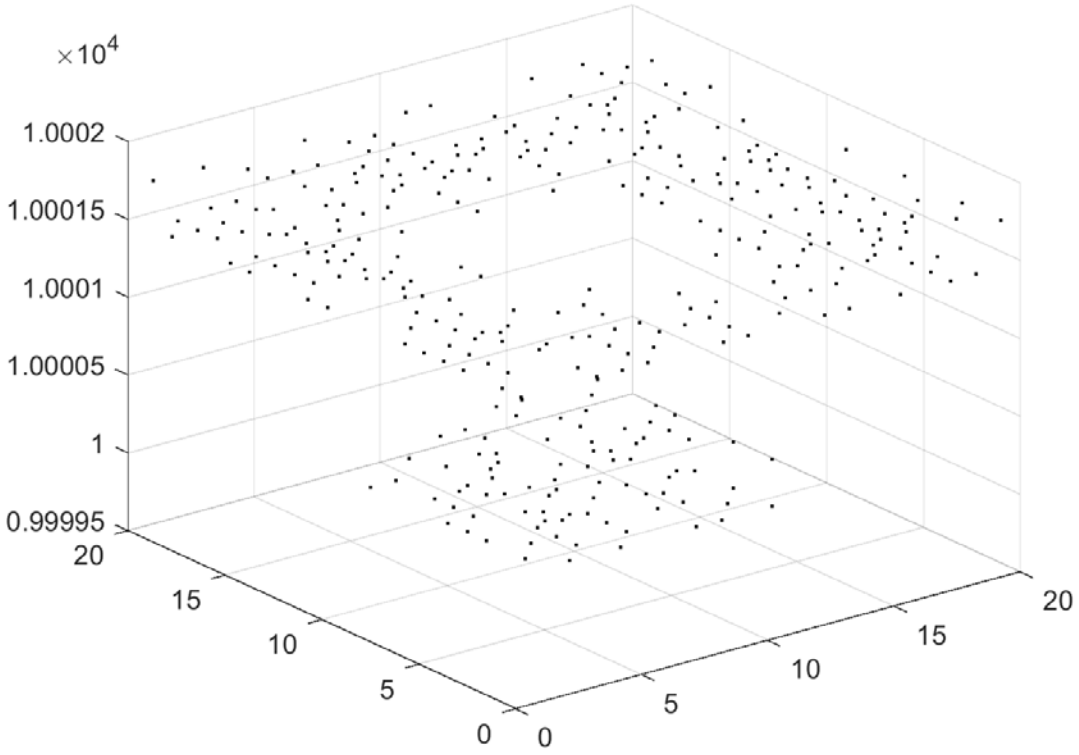


Figure 29: Figure 8 in MATLAB Code, Scatter Plot of Fused 3-D and 2-D Data.

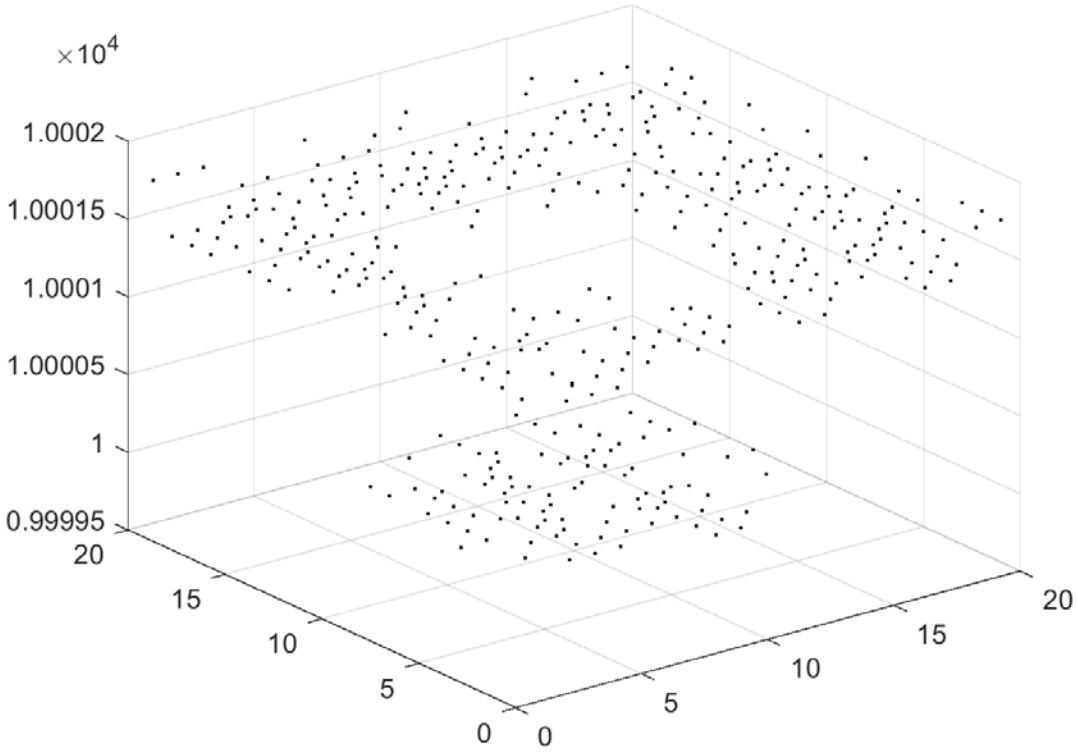


Figure 30: Figure 9 in MATLAB Code, Scatter Plot of only 3-D LIDAR Data

Appendix B: 100 x 100 Cropped Detailed Target Area MATLAB Code

The MATLAB code in this section is similar to the code in Appendix A. The main difference is that the target area is more detailed. Figure 4 in the code will not be displayed because it is identical to Figure 24 in Appendix A.

Creation of Gaussian Beam

```
stdevx=1;%standard deviation parameter in x direction
stdevy=1;%%standard deviation parameter in y direction
xx=-49:50; %creating the size of the array in the x coordinate
xx_mat=ones(100,1)*xx; %creating an array of ones in the x coordinate
 yy=-49:50;
 yy_mat=(ones(100,1)*yy)'; %creating an array of ones in the y coordinate
beam=(1/(2*pi*stdevx*stdevy))*exp(-((yy_mat).^2)/(2*(stdevy^2))).*exp(-((xx_mat).^2)/(2*(stdevx^2)));
%Creating the gaussian beam using beam equation
beam=beam.*ones(100,100)/sqrt(sum(sum(beam.*beam))); %normalizing the beam
% figure(1)
% imagesc(beam) %displaying an image of the beam
% hcb=colorbar;
%  set(get(hcb,'Title'),'String','Photons')
```

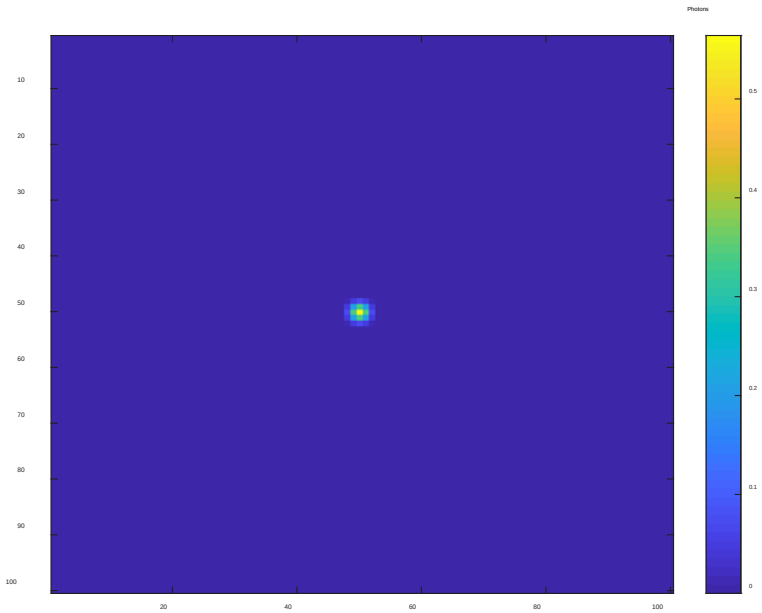


Figure 31: Figure 1 in MATLAB Code, Gaussian Beam

Target profile

```
load '3-D_dataset.mat' %dataset that contains 2-D range and reflectivity variables
Sigma_w = 2e-9; % Pulse standard deviation in units of seconds
Rmin=9990; % Minimum range in the range gate
minT=Rmin*2/3e8; % first time that the receiver will measure the return
Rmax=10010; % Maximum range in the range gate
maxT=Rmax*2/3e8; % last time that the receiver will measure the return
deltat=Sigma_w; % Sample time in seconds.
t=minT:deltat:maxT; % Range of times in the range gate
target_area=round((-range_img(125:224,175:274))/50); % Define the area of the target at 10km
target_area_norm = (.3*target_area)+10000; %converts the target area coordinates to the same as the
estimated range coordinates so that they can be compared
rho_t=reflect_img(125:224,175:274); % Target reflectivity at each pixel
T_p(100,100,:)=zeros(size(t)); %Creating the size of the target profile
for xn=1:100  %loop to creating target profile
   for ym=1:100
      T_p(ym,xn,:)=zeros(size(t)); % create a range vector per pixel
      indxx=target_area(ym,xn)+1; % Locate the range vector index
      T_p(ym,xn,abs(indxx))=rho_t(ym,xn);% Assign a dirac based on target reflectivity and area of spatial
sample
   end
end
% figure(2)
% imagesc(target_area) %display image of target area
% hcb=colorbar;
% set(get(hcb,'Title'),'String','Range (m)')
% figure(3)
% mesh(target_area)
% hcb=colorbar;
% set(get(hcb,'Title'),'String','Range (m)')
```

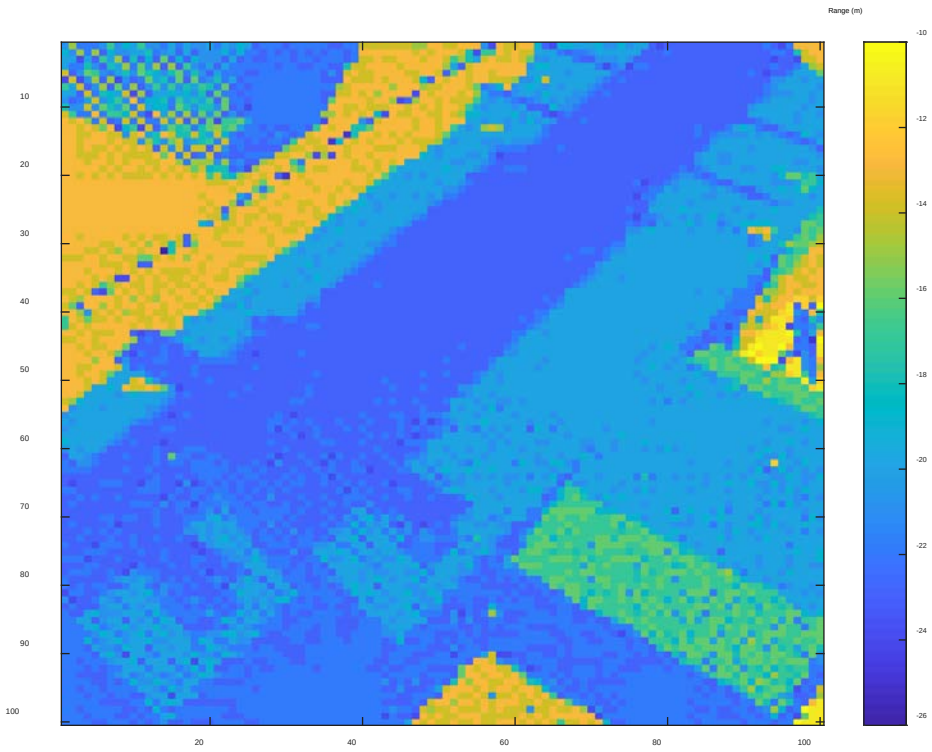


Figure 32: Figure 2 in MATLAB Code, Cropped Target Area

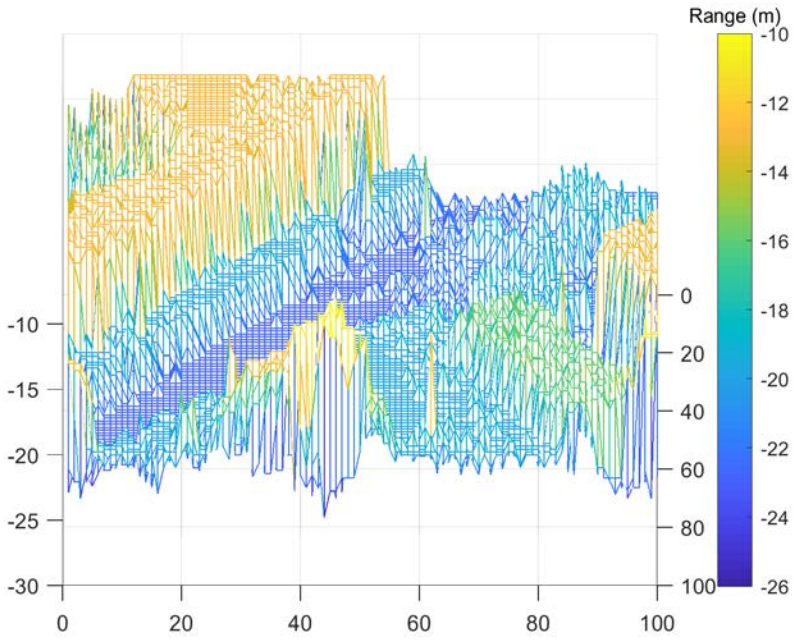


Figure 33: Figure 3 in MATLAB Code, Mesh Plot of Target Area

Creating scanning beam and receiver size creation

```
Z=10000; %distance to target
lam=1.55e-6; %wavelength of beam
tau_atm=1; % Atmospheric Transmission
tau_opt=1; % Receiver Optics Transmission
reciever_focal=1; % Focal length of the LIDAR receiver in meters
theta_r=pi; % Reflection angle for Lambertian targets
ap_diameter=.1; % Aperture diameter in units of meters
 for k = -49:50 %loop to scan beam
    for m = -49:50

  S = circshift((beam),[round(k+rand) round(m+rand)]); %circshift shifts the beam and the rand commands randomly move the beam in a non linear direction
%    figure(4)
%    imagesc((1:100),(1:100),abs(S)) %displays the beam scanning across environment
%    hcb=colorbar;
%    set(get(hcb,'Title'),'String','Photons')
%    colormap(gray)
   pause(.1)
   E_t=.001*abs(S).^2; %1mJ pulse distributed by diffraction
   P_t=zeros(100,100,max(size(t))); % Allocate memory for 3-D pulse array.  P_t stands for power transmitted
       for tk=1:max(size(t)) % Setup loop to visit each time in the range gate
          P_t(:,:,tk)=(E_t/(sqrt(2*pi)*Sigma_w))*exp(-((t(tk)-Z*2/3e8).^2)/(2*Sigma_w^2)); % Images of the pulse at each range
       end
   I_target = tau_atm*P_t; %Intensity of target
   P_ref = I_target; % Reflected power from the target in units of Watts
   I_receiver=tau_atm*P_ref/(theta_r*Z^2);  % Intensity at the aperture
   P_rec = tau_opt*(ap_diameter^2)*pi*I_receiver/4; % Received signal power from a unit reflectance and area target at 1000 meters
   P_rec_tot=real(ifft(fft(P_rec,max(size(t)),3).*fft(T_p,max(size(t)),3),max(size(t)),3)); % Received signal power from every point in the target area at the correct range due to the convolution between the target profile and the waveform array.  The convolution is carried out using the convolution property of the Fourier transform.
%Function to add AWGN to a given signal
%Authored by Mathuranathan Viswanathan
```

```
%How to generate AWGN noise in Matlab/Octave by Mathuranathan Viswanathan
%is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0  International License.
%You must credit the author in your work if you remix, tweak, and build upon the work below
   SNR_dB = 10; % signal to noise ratio in DB, used to determine how much noise you want.  The higher the
SNR the lower the noise
   L=length(t); %the length of the time slices of the waveform
   SNR = 10^(SNR_dB/10); %SNR to linear scale
   Esym=sum(abs(P_rec_tot).^2)/(L); %Calculate actual symbol energy
   N0=Esym/SNR; %Find the noise spectral density
   if(isreal(P_rec_tot))
      noiseSigma = sqrt(N0);%Standard deviation for AWGN Noise when x is real
      n = noiseSigma.*randn(1,100,L);%computed noise
   else
      noiseSigma=sqrt(N0/2);%Standard deviation for AWGN Noise when x is complex
      n = noiseSigma.*(randn(1,100,L)+1i.*randn(1,100,L));%computed noise
  end
   P_rec_tot_noisy = P_rec_tot + n; %received signal

   R_vec=t*3e8/2; %the range vector of the waveform

   for indxx=1:max(size(t)) %loop to create the waveforms from the power received at the aperture
      waveform1(k+50,m+50,indxx)=sum(sum(P_rec_tot_noisy(:,:,indxx)));
   end
      temp_img = sum(P_rec_tot_noisy(:,:,:),3);%2-D imager
      [tempfindy, tempfindx]=find(temp_img==max(max(temp_img)));
      yyc(k+50,m+50)=round(mean(tempfindy));
      xxc(k+50,m+50)=round(mean(tempfindx));
      idata=squeeze(waveform1(k+50,m+50,:)); %the intensity of the waveform
      Xx=find(idata==max(idata)); %the max intensity of the waveform, should be the middle of the beam.
This is how the 2-D imager knows where the beam truly is.
      Est_range(k+50,m+50)=mean(R_vec(Xx)); %This is the range that is calculated from the range vector
at the max intensity of the waveform
%        figure(5)
%        imagesc(Est_range)
%        colorbar
       pause(.1)
   end
 end
```

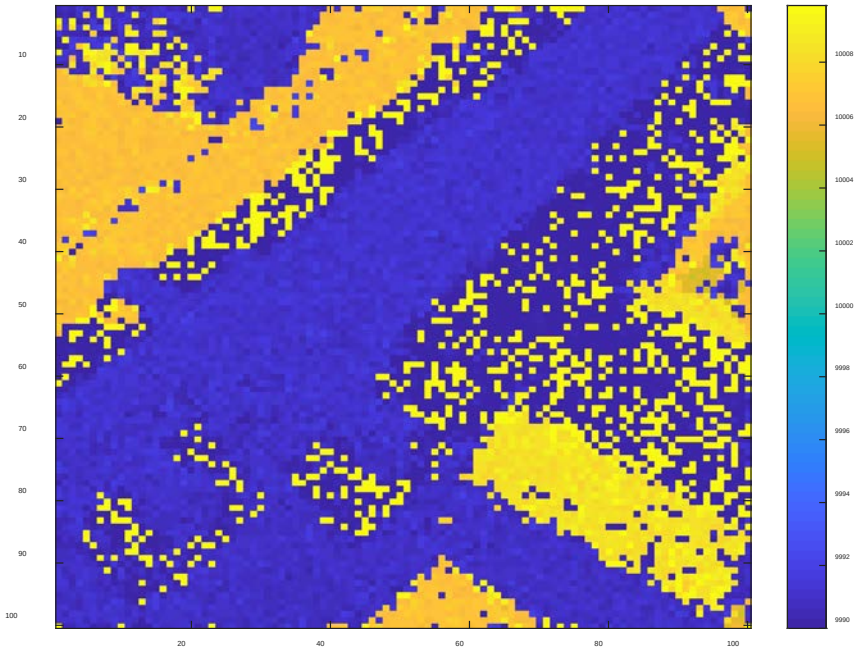


Figure 34: Figure 5 in MATLAB Code, Estimated Range

Plotting waveforms

```
 wvfrm1=zeros(1,67); %creating a zero array for the waveforms in time
for frms=1:67 %loop to show the entire target area in time.  First the waveform hits the top of the building and then the ground.
% figure(6)
% image((10^8)*waveform1(:,:,frms)+1e-12); %image of waveform progression in time/range through target area
% colormap(gray)
% hcb=colorbar;
% set(get(hcb,'Title'),'String','Photons')
pause(.1)

 wvfrm1(frms)=sum(sum(P_rec_tot_noisy(:,:,frms))); %the waveform is the sum of the received power
%  figure(7)
%  plot(t,wvfrm1) %plot of the waveform intensity as it progresses through time
 pause(.1)
```

```
end
```

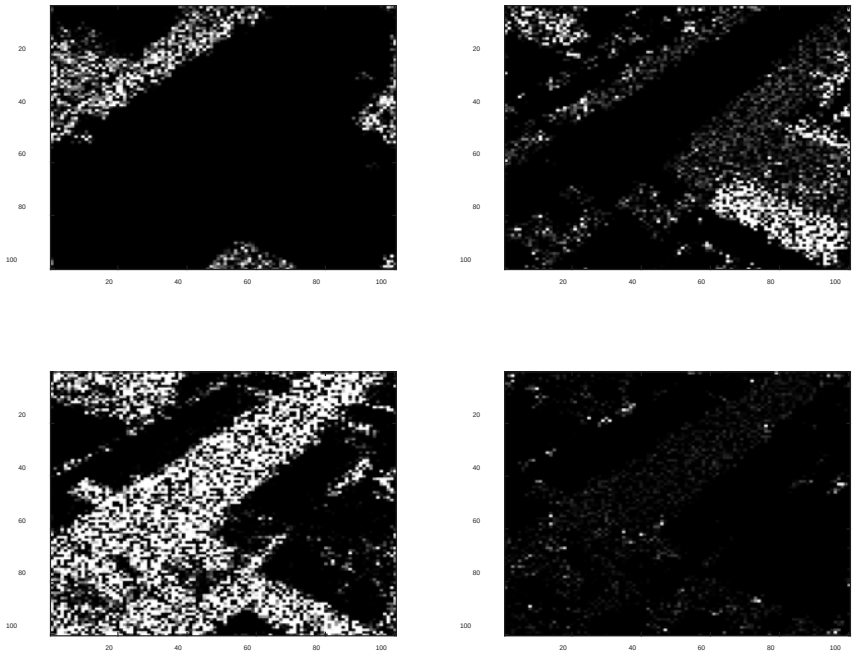


Figure 35: Figure 6 in MATLAB Code, Waveform propagation. Top left: 88 ns, top right: 100 ns, bottom left: 112 ns, bottom right: 116 ns.

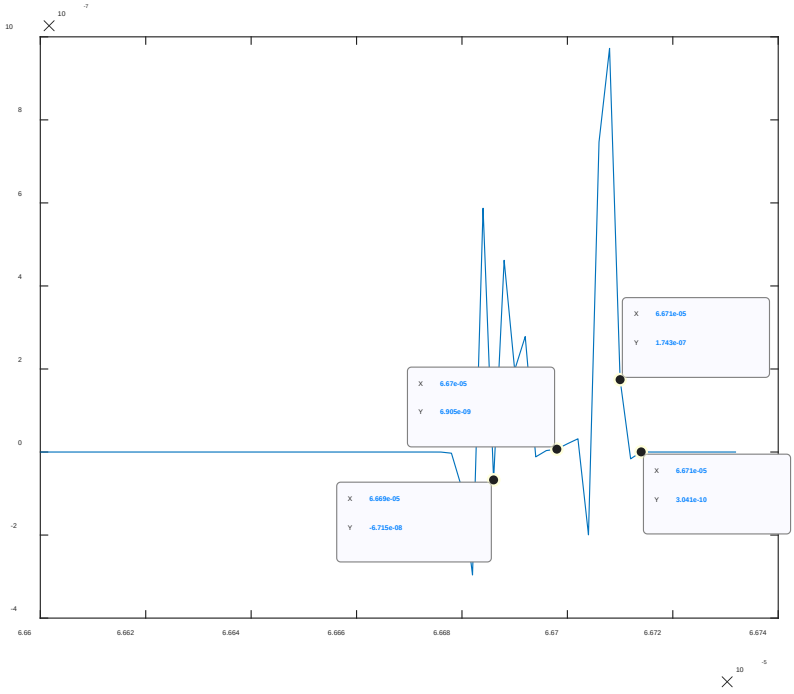


Figure 36: Figure 7 in MATLAB Code, Plot of Waveform. Each marker corresponds to Figure 35 plots.

Create point cloud

```
err1 = 0; %initializing the first error
err2 = 0; %initializing the second error
for p = 1:100 %loop to create scatter plot of estimated range
  for q = 1:100
%       figure(8)
%       h = scatter3(xxc(q,p),yyc(q,p),Est_range(q,p),10,'k','.'); %Change xxc and yyc to q  and p
respectively to get noisy image. invert image by taking 10,000 minus Est_range,
%       %h = scatter3(xxc(q,p),yyc(q,p),10000-Est_range(q,p),10,'k','.');
%       pause(.1)
%       hold on
      err1 = err1 + ((Est_range(q,p)-target_area_norm(yyc(q,p),xxc(q,p))).^2); %error between the estimated
range and the actual range of the target area
      pause(.1)
  end
end
rmse1 = sqrt(err1/(100*100)); %the root mean square error of the ranges.
for p = 1:100
   for q = 1:100
%       figure(9)
%       g = scatter3(p,q,Est_range(q,p),10,'k','.'); %Change xxc and yyc to q  and p respectively to get noisy
image. invert image by taking 10,000 minus Est_range,
%       %g = scatter3(p,q,10000-Est_range(q,p),10,'k','.');
%       pause(.1)
%       hold on
      err2 = err2 + ((Est_range(q,p)-target_area_norm(q,p)).^2);
      pause(.1)
  end
end
rmse2 = sqrt(err2/(100*100));
```

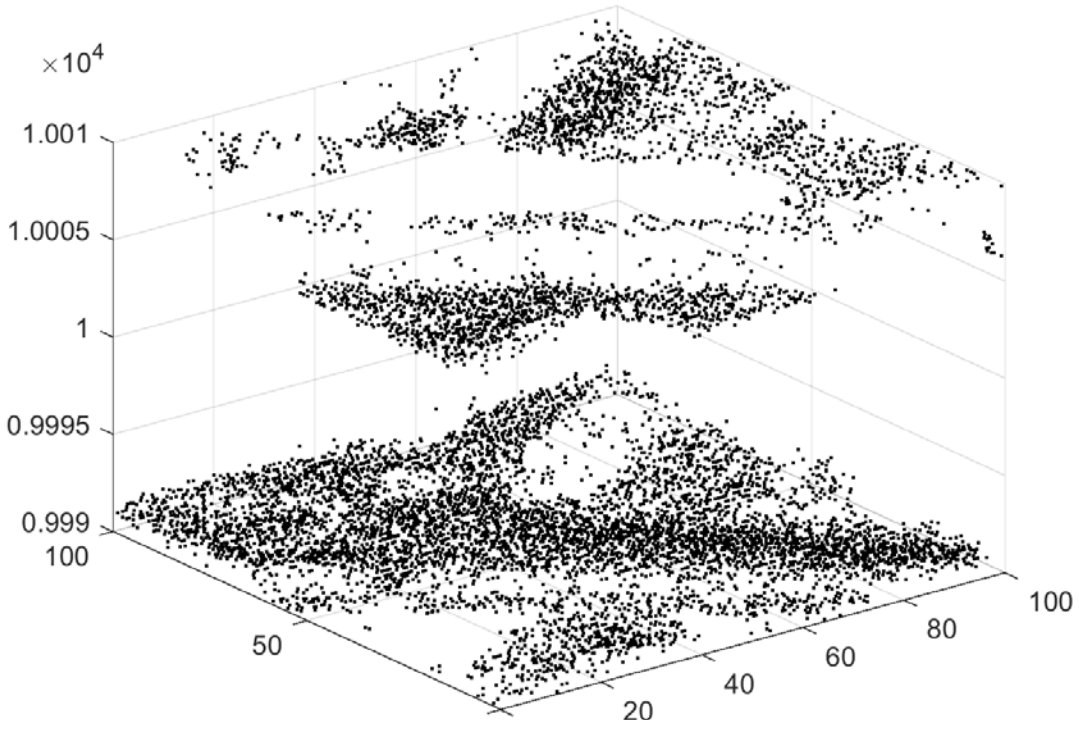


Figure 37: Figure in MATLAB Code, Scatter Plot of Fused 3-D and 2-D data.

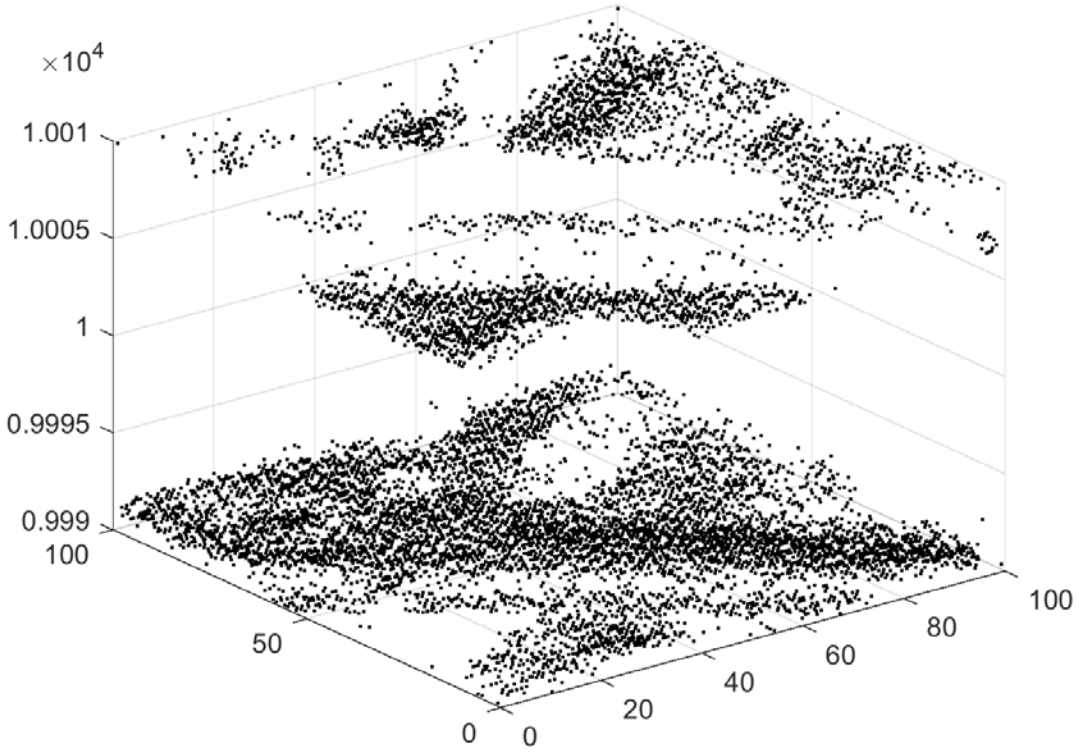


Figure 38: Figure 9 in MATLAB Code, Scatter Plot of Only 3-D LIDAR Data

Appendix C: Optics Lab MATLAB code

This section describes the MATLAB code used to perform the hardware-in-the-loop experiment to produce “real-world” LIDAR data. The Gaussian beam creation, target area, and scanning beam figures are identical to figures in previous appendices so they will not be added to this section. The Gaussian beam figure can be found in Appendix B, Figure 30. The target area figure can be found in Appendix A, Figure 23. The scanning beam can be found in Appendix A, Figure 24.

Camera Start-up Procedure

1. Turn on monitor close to computer off table

2. Start computer and login

3. Power on Camera

4. Turn on monitor on table

The following sequence will grab a single frame image from the Thor labs Camera

```
vid = videoinput('thorlabsimaq', 1, '8050m-ge-te (04999)');%creates object for camera input
src = getselectedsource(vid);%creates source object

%obj = videoinput('thorlabsimaq', 1);
figure(1)
imagesc
set(gcf,'MenuBar','none')%removes menu bar from figure 1

set(gca,'DataAspectRatioMode','auto')

set(gca,'Position',[0 0 1 1])
vid.ROIPosition = [900 500 1296 1972];%chooses region of interest for CCD array
for i=1:3600
```

```
   i
frame = getsnapshot(vid);%takes snapshot
figure(2)
imagesc(frame) %displays snapshot of image making sure the camera is initialized.
pause(.1)
end
```

Creation of Gaussian Beam

```
stdevx=1;%standard deviation parameter in x direction
stdevy=1;%%standard deviation parameter in y direction
sz=100; %standard deviation width of array
xx=-sz/2+1:sz/2;
xx_mat=ones(sz,1)*xx;
yy_mat=xx_mat';
beam=(1/(2*pi*stdevx*stdevy))*exp(-((yy_mat).^2)/(2*(stdevy^2))).*exp(-((xx_mat).^2)/(2*(stdevx^2)));
beam=beam.*ones(sz,sz)/sqrt(sum(sum(beam.*beam)));
% figure(1)
% imagesc(beam)
% colorbar
```

Target profile and receiver size creation

```
Z=10000; %distance to target
lam=1.55e-6; %wavelength of beam
Sigma_w = 2e-9; % Pulse standard deviation in units of seconds
tau_atm=1; % Atmospheric Transmission
tau_opt=1; % Receiver Optics Transmission
reciever_focal=1; % Focal length of the LIDAR receiver in meters
theta_r=pi; % Reflection angle for Lambertian targets
ap_diameter=.1; % Aperture diameter in units of meters
sz=100;
Rmin=9990; % Minimum range in the range gate
minT=Rmin*2/3e8; % first time that the receiver will measure the return
Rmax=10010; % Maximum range in the range gate
maxT=Rmax*2/3e8; % last time that the receiver will measure the return
deltat=Sigma_w; % Sample time in seconds.
```

```
t=minT:deltat:maxT; % Range of times in the range gate
target_area=ones(sz,sz)*5; % Define the area of the target at 10001.5 m
target_area(26:75,26:75)=zeros(50,50); % Define the area of the target at 10km
rho_t=ones(sz,sz)*0.1; % Target reflectivity at each pixel
T_p(sz,sz,:)=zeros(size(t));
for xn=1:sz
   for ym=1:sz
      T_p(ym,xn,:)=zeros(size(t)); % create a range vector per pixel
      indxx=target_area(ym,xn)+1; % Locate the range vector index
      T_p(ym,xn,indxx)=rho_t(ym,xn);% Assign a dirac based
            % on target reflectivity and area of the spatial sample.
   end
end
% figure(2)
% imagesc(target_area)
% colorbar
```

Creating scanning beam

```
 %obj = videoinput('thorlabsimaq', 1);
 for k = -29:30
    for m = -29:30

  %S = circshift((beam),[round(k+rand) round(m+rand)]);%beam scan with
  %random noise
  S = circshift((beam),[k m]);%beam scan with no noise
  figure(1)
  imagesc((1:sz),(1:sz),abs(S))
  %colorbar
  colormap(gray)
  pause(.5)
E_t=.001*abs(S).^2;
P_t=zeros(sz,sz,max(size(t))); % Allocate memory for 3-D pulse array
      for tk=1:max(size(t)) % Setup loop to visit each time in the range gate
P_t(:,:,tk)=(E_t/(sqrt(2*pi)*Sigma_w))*exp(-((t(tk) -Z*2/3e8).^2)/(2*Sigma_w^2)); % Images of the pulse at
each range
      end
I_target = tau_atm*P_t;
```

```
P_ref = I_target; % Reflected power from the target in units of Watts
I_receiver=tau_atm*P_ref/(theta_r*Z^2);  % Intensity at the aperture
P_rec = tau_opt*(ap_diameter^2)*pi*I_receiver/4; % Received signal power from a unit reflectance and area target at 1000 meters
P_rec_tot=real(ifft(fft(P_rec,max(size(t)),3).*fft(T_p,max(size(t)),3),max(size(t)),3)); % Received signal power from every point in the target area at the correct range due to the convolution between the target profile and the waveform array.  The convolution is carried out using the convolution property of the Fourier transform.
%Need a new loop the length of t
max_P=max(max(max(P_rec_tot)));%maximum value of the received waveform
clear frame
for q =31:45 %31:45 frames are the only frames with useful data
   figure(3)
   temp=(P_rec_tot(:,:,q));
   temp = temp*256/max_P;%normalizing waveform
   image(temp)
   frame(:,:,q) = getsnapshot(vid);%camera takes a picture of the frame
   pause(.1)
   figure(4)
   imagesc(frame(:,:,q))
   pause(0.1)
end
R_vec=t*3e8/2;

        for indxx=31:45
           waveform1(k+30,m+30,indxx)=sum(sum(frame(:,:,indxx)));
        end
        temp_img = sum(frame(:,:,:),3);%2-D imager
        [tempy ,tempx]=(find(temp_img==max(max(temp_img))));
        [yyc(k+30,m+30)] =mean(tempy);%coordinates of each pixel based on the 2-D imager
        [xxc(k+30,m+30)] =mean(tempx);%coordinates of each pixel based on the 2-D imager

        idata=squeeze(waveform1(k+30,m+30,:));
        Xx=find(idata==max(idata));
        Est_range(k+30,m+30)=mean(R_vec(Xx));

    end
   save wksp_collect xxc yyc Est_range k m waveform1
end
```

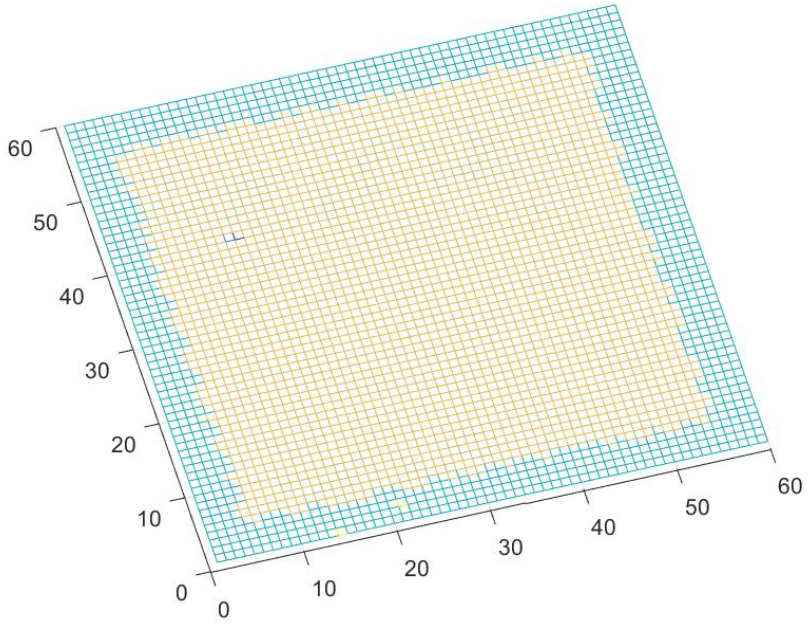


Figure 39: 3-D data only

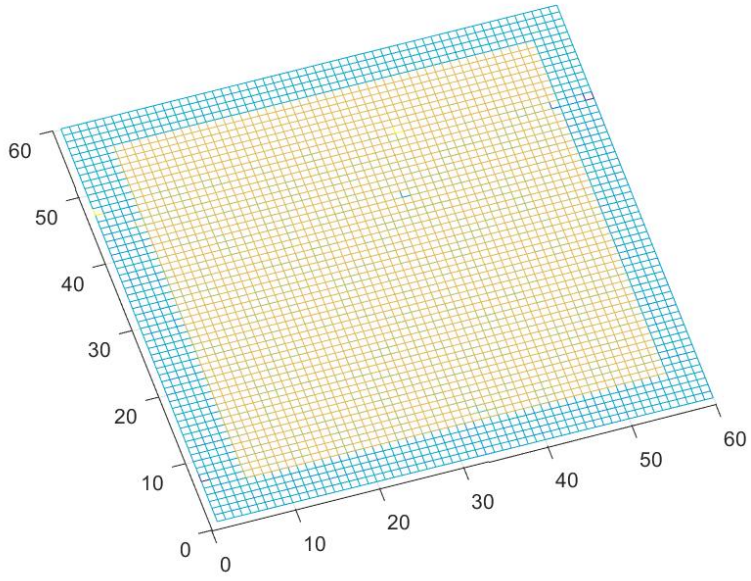


Figure 40: Fused Data

Interpolation Code

```
load wksp_collect %Estimated Range data with no noise
xxc_base=xxc;%x coordinates
yyc_base=yyc;%y coordinates
Est_range_base=Est_range; %Estimated range
xmin=min(min(xxc_base)); %minimum value in x coordinates
xmax=max(max(xxc_base)); %maximum value in x coordinates
ymin=min(min(yyc_base));%minimum value in y coordinates
ymax=max(max(yyc_base));%maximum value in y coordinates
xsize=xmax-xmin; %range of x coordinate values
ysize=ymax-ymin;%range of y coordinate values
winsize=25; %window size
target_area=10000*ones(ysize+winsize,xsize+winsize); %initializing the target area
for k=1:60
for m=1:60
target_area(round(yyc_base(k,m)-ymin+1),round(xxc_base(k,m)-xmin+1))=Est_range_base(k,m); %target
area from estimated range
end
end
interp_area=10000*ones(ysize,xsize); %Initialization of interpolated area
for ii=1:ysize
ii;
for jj=1:xsize
temp=target_area(ii:ii+winsize-1,jj:jj+winsize-1);
binmap=(temp>0);
if(sum(sum(binmap))>0)
interp_area(ii,jj)=sum(sum(temp))/sum(sum(binmap));
end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%
load wksp_collect2 %Estimated Range data with noise
target_area2=10000*ones(ysize+winsize,xsize+winsize);
interp_area2=10000*ones(ysize,xsize);
for k=1:60
for m=1:60
target_area2(round(yyc_base(k,m)-ymin+1),round(xxc_base(k,m)-xmin+1))=Est_range(k,m);
```

```
end
end
for ii=1:ysize
ii;
for jj=1:xsize
temp=target_area2(ii:ii+winsize-1,jj:jj+winsize-1);
binmap=(temp>0);
if(sum(sum(binmap))>0)
interp_area2(ii,jj)=sum(sum(temp))/sum(sum(binmap));
end
end
end
target_area3=10000*ones(ysize+winsize,xsize+winsize);
interp_area3=10000*ones(ysize,xsize);
for k=1:60
for m=1:60
if((yyc(k,m)-ymin)>=0)
if((xxc(k,m)-xmin)>=0)
target_area3(round(yyc(k,m)-ymin+1),round(xxc(k,m)-xmin+1))=Est_range(k,m);
end
end
end
end
for ii=1:ysize
ii;
for jj=1:xsize
temp=target_area3(ii:ii+winsize-1,jj:jj+winsize-1);
binmap=(temp>0);
if(sum(sum(binmap))>0)
interp_area3(ii,jj)=sum(sum(temp))/sum(sum(binmap));
end
end
end
end
```

Published with MATLAB® R2018b

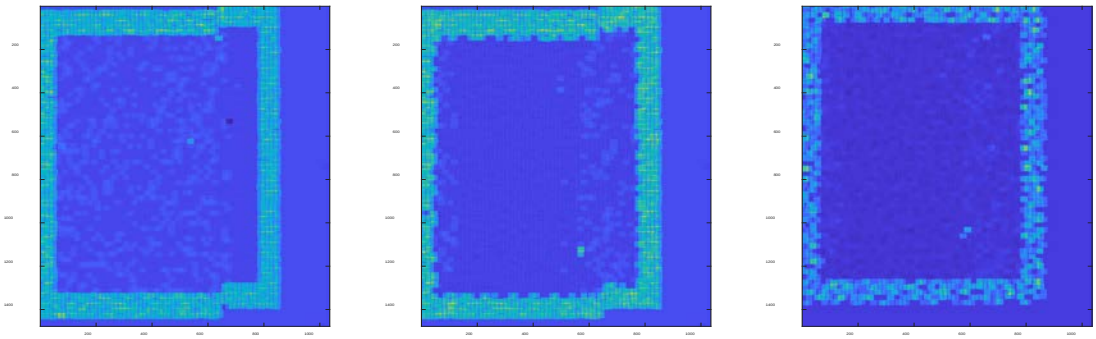


Figure 41: Images of Interpolated Area. Left – Original Target Area, Middle – Estimated Target Area with 3-D Data Only, Right – Estimated Target Area with Fused 3-D and 2-D data.

Bibliography

Carter, J and Keil Schmid, et al. "Lidar 101: An Introduction to Lidar Technology, Data, and Applications." National Oceanic and Atmospheric Administration (NOAA) Coastal Services Center. November 2012. Charleston, SC: NOAA Coastal Services Center. Retrieved on 28 November 2018.

Richmond, Richard D. and Stephen C. Cain. *Direct-Detection LADAR Systems.* SPIE, Bellingham, WA, 2010. ISBN 9780819480729.

Dolce, Paul F. *A Statistical Approach to Fusing 2-D and 3-D LADAR Systems.* MS Thesis, AFIT/GE/ENG/11-09. Graduate School of Engineering and Management, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, March 2011.

"Advantages and Disadvantages of LiDAR." *Lidarradar.com.* 19 January 2018. Retrieved on 10 December 2018.

Walsh, David. "Warfighters Reap Benefits of LIDAR Mapping Technology." *Defensesystems.com.* 26 July 2011. Retrieved on 15 December 2018.

Hoffmann, N., Weidner, F., Urban, P., et al. "Framework for 2-D-3-D image fusion of infrared thermography with preoperative MRI." *Biomedical Engineering / Biomedizinische Technik*, 62(6), pp. 599-607. (23 January 2017).

Winnemöller, Holger, Alexandrina Orzan, Laurence Boissieux, and Joëlle Thollot. "Texture Design and Draping in 2-D Images." *Computer Graphics Forum, Wiley, 2009, Proceedings of the Eurographics Symposium on Rendering 2009*, 28 (4): pp.1091-1099. (27 August 2010).

Li, Y., Zheng, Q., Sharf, A., Cohen-Or, D., Chen, B., Mitra, N. J., "2-D-3-D Fusion for Layer Decomposition of Urban Facades," *in IEEE International Conference on Computer Vision*. pp. 882-889. 2011.

Diebel, James and Sebastian Thrun, "An Application of Markov Random Fields to Range Sensing," *Proceedings of the Neural Information Processing Systems 2005 Conference.* 2005.

Kopf, J., Neubert, B., Chen, B., Cohen, M., Cohen-Or, D., Deussen, O., Uyttendaele, M., Lischinski, D. "Deep Photo: Model-Based Photograph Enhancement and Viewing." *ACM Transactions on Graphics*. 27 (5), Article 116, 10 pages, (December 2008).

Stamos, I., Liu, L., Chen, C., Wolberg, G., Yu, G., Zokai, S. "Integrating Automated Range Registration with Multiview Geometry for the Photorealistic Modeling of Large-Scale Scenes." *International Journal of Computer Vision*, 78(2-3), pp. 237-260. (July 2008).

Nairat, Mazen and David Voelz, "Performance characteristics of a scanning laser imaging system through atmospheric turbulence," *Optical Engineering* 51(10), 101708 (October 2012).

"How does LiDAR work?" *LiDAR-UK.com.* Retrieved on 10 December 2018.

"What is LIDAR?" National Oceanic and Atmospheric Administration (NOAA) Coastal Services Center. 25 June 2018. Retrieved on 10 December 2018.

"Understanding CCD Read Noise." *qsimaging.com.* Quantum Scientific Imaging. Retrieved on 2 January 2019.

Kashani, Alireza G., Michael J. Olsen, Christopher E. Parrish, and Nicholas Wilson, "A Review of LIDAR Radiometric Processing: From Ad Hoc Intensity Correction to Rigorous Radiometric Calibration," *Sensors*, 15: 28099-28128 (November 6, 2015).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	**2. REPORT TYPE**	**3. DATES COVERED** *(From - To)*
21-03-2019	Master's Thesis	Aug 2017 – Mar 2019

4. TITLE AND SUBTITLE

A Quantitative Analysis of the Fusion of 3-D Scanning LIDAR Systems and 2-D Imaging Systems

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Milton, Jr., Michael, F, Capt

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
Wright-Patterson AFB OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-19-M-044

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Intentionally Left Blank

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A. APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This research will demonstrate the feasibility of fusing the superior spatial resolution of a 2-D imaging system with the precise range to target information of a 3-D imaging system to create a LIDAR imaging system that can accurately find what and where a target is. The 3-D imaging system will use a scanning method as opposed to a flash method that has been used in similar research. The goal of this research is to improve performance of scanning LIDAR so it has better spatial resolution. The research in this thesis proves that incorporating 2-D imaging data into 3-D scanning LIDAR data improves the spatial resolution of the LIDAR system, at least for simplistic environments. This idea is introduced to improve LIDAR systems for missile seekers. Incorporating this system in missile seekers will allow improved target tracking compared to a 3-D scanning system alone.

15. SUBJECT TERMS

3-D Scanning LIDAR; Fusion; 2-D Imaging

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			Dr. Stephen C. Cain, AFIT/ENG
U	U	U	UU	95	**19b. TELEPHONE NUMBER** *(include area code)* (937) 255-3636 x4716 stephen.cain@afit.edu

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