CHANGE DETECTION IN LIDAR SCANS OF URBAN ENVIRONMENTS

A Dissertation

Submitted to the Graduate School
of the University of Notre Dame
in Partial Fulfillment of the Requirements
for the Degree of

Doctor of Philosophy

by

Alexandri Gregor Zavodny

________________________________________

Patrick J. Flynn, Director

Graduate Program in Computer Science and Engineering
Notre Dame, Indiana
November 2012
Abstract

by
Alexandri Gregor Zavodny

Light Detection and Ranging (LIDAR) is a popular tool for range sensing applications, with applications including environmental modeling and urban planning. Modern acquisition platforms facilitate data collection rates of over two billion points per hour, enabling the collection of massive datasets with ease. These datasets must typically undergo a large amount of processing before use, making maintenance a difficult issue. Additionally, the presence of transient objects can affect dataset quality, both as unwanted data and by obscuring the underlying surface model.

In this dissertation, we present a framework for enabling automatic change detection between large LIDAR datasets of urban environments. For two large scale datasets, we extract the relevant portions using information about the paths of the acquisition vehicles. However, acquisition inaccuracies can result in subset misalignment of up to 10 meters. To correct for this, we utilize a variety of point cloud alignment techniques, including a novel point descriptor, to bring overlapping pieces of data into alignment. Given the properly aligned data, we then use novel hierarchical and point-based techniques to extract regions of change between the two datasets. These regions can then be extracted and presented for further processing or filtering.
We present the results of our research executed on datasets totaling over 93 billion sample points. At over 1.5 terabytes in size, this represents by far the largest collection of ground-based LIDAR examined in the open literature. We quantify our results with a variety of objective metrics, investigate modes of failure, and recommend directions for future research.
To my parents, Ted and Esther.
### CONTENTS

**FIGURES** ........................................................................................................... v

**TABLES** ........................................................................................................... ix

**ACKNOWLEDGMENTS** .................................................................................... x

**CHAPTER 1: INTRODUCTION** ................................................................. 1
  1.1 Motivation ................................................................................................. 1
  1.2 Problem Statement .................................................................................. 2

**CHAPTER 2: DATA** .................................................................................... 5
  2.1 Data Acquisition ....................................................................................... 5
    2.1.1 LIDAR Scanners .............................................................................. 5
    2.1.2 Fusion Process ................................................................................. 9
    2.1.3 Acquisition Vehicles ....................................................................... 10
  2.2 Datasets .................................................................................................... 12
    2.2.1 2008 Datasets .............................................................................. 13
    2.2.2 2010 Datasets .............................................................................. 13
    2.2.3 NAVTEQ True Datasets .................................................................. 16
    2.2.4 Data Preprocessing ........................................................................ 20

**CHAPTER 3: POINT CLOUD ALIGNMENT** ........................................... 23
  3.1 Motivation ................................................................................................. 23
  3.2 Related Work ........................................................................................... 24
    3.2.1 Problem Definition ......................................................................... 26
    3.2.2 ICP-Based Approaches ................................................................... 28
    3.2.3 Non-ICP Approaches ..................................................................... 33
  3.3 Primary Methodology ............................................................................... 35
    3.3.1 Overlap Subsets ............................................................................. 35
    3.3.2 Problem Statement: Overlap Subset Alignment ................................ 36
    3.3.3 Alignment using ICP ....................................................................... 38
3.3.4 Alignment using RANSAC ........................................ 39
3.3.5 Domain Knowledge and Heuristics .......................... 42
3.3.6 Point Features .................................................. 45
3.3.7 Point Matching .................................................. 55
3.4 Metrics ............................................................. 56
3.5 Parameter Sweeps .................................................. 59
3.6 ICP versus RANSAC ................................................. 69
3.7 Discussion ............................................................. 72
  3.7.1 Sources of Misalignment ...................................... 72

CHAPTER 4: CHANGE DETECTION ........................................ 76
  4.1 Motivation / Problem Statement .................................. 76
  4.2 Related Work ....................................................... 78
    4.2.1 Aerial Approaches ............................................. 79
    4.2.2 Ground-Based Approaches ................................... 80
    4.2.3 Abstract Comparison Methods ............................... 83
    4.2.4 Evaluation Metrics ............................................. 84
  4.3 Methodology ........................................................ 85
    4.3.1 Octree Comparison .............................................. 86
    4.3.2 Point Comparison ............................................... 88
  4.4 Results and Discussion ............................................ 91
    4.4.1 Visibility ....................................................... 94
    4.4.2 Classification .................................................. 97

CHAPTER 5: CONCLUSIONS ................................................ 102
  5.1 Future Work ........................................................ 103

APPENDIX A: DATASETS .................................................. 104

BIBLIOGRAPHY ........................................................... 126
FIGURES

2.1 A single SICK LMS-291 scanner using during acquisition. Aside from an internal rotating mirror, the device is stationary during operation. ................................................... 7

2.2 A Velodyne HDL-64E S2 scanner using during acquisition. The cylindrical portion of the device rotates at 5 - 15Hz during operation. 8

2.3 The positioning of SICK scanners on the 2008 acquisition vehicle. 11

2.4 The positioning of SICK scanners on the 2010 acquisition vehicle, from above. The arrow points in the direction of motion. Side scanners are angled 45° away from the direction of motion; front and back scanners are angled up and down with respect to the horizon. ................................................... 12

2.5 Example scan data from ND_2008_00. The point density is not very high, but is relatively consistent over the face of buildings, and precision is high. ................................................... 14

2.6 Example scan data from ND_2008_00. Most of the path traversed by the van was along the campus loop, too far from buildings to sufficiently sample them; this is an example of some of the foliage that is present throughout most of the dataset. ............... 15

2.7 Example scan data from ND_2010_00. This subset is of the same building as in figure 2.5. Note the decreased density toward the roof of the building facade, but the color data and increased density elsewhere can prove beneficial. ................................. 16

2.8 Example scan data from ND_2010_00. As the acquisition vehicle was smaller, we were able to acquire more buildings with ease. . . 17

2.9 Example scan data from the NAVTEQ True system obtained in Cupertino, CA. Top: a residential street. Note the varied reflectance from street paint. Bottom: a shopping mall parking lot. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
3.1 Misalignment between two scans. The scan in blue (dark grey) was obtained two years after the scan in red (light grey) and they exhibit a strong misalignment, in spite of both being geo-registered.

3.2 A histogram of the altitudinal alignment error \( n = 50 \) obtained by comparing ground truth alignment with automatic ground plane pre-alignment. The mean value is 0.85 \( m \) with \( \sigma = 0.52 \) \( m \).

3.3 With respect to an point \( p \) and average normal \( \bar{n} \), SNADHs use a mapping that transforms each neighboring point into a pair of values \( (\theta, \phi) \), whose values capture the rotation of \( n_q \) about the reference normal and rotation away from the reference normal, respectively.

3.4 Example SNADHs showing distinctive patterns.

3.5 With respect to an oriented point \( p \), spin images use a mapping that transforms each neighboring point into a pair of values \( (\alpha, \beta) \), which correspond to the distance perpendicular to and parallel to the normal \( n_p \).

3.6 For each pair of points \( p_i \) and \( p_j \) within a neighborhood, Fast Point Feature Histograms are computed with respect to a Darboux frame as shown here.

3.7 A two-dimensional shape context, showing both radial and angular divisions. The 3D Shape Context is analogous. On the left, radial bins are divided evenly; on the right, they are divided such that each shell has equal volume.

3.8 Top: the initial misalignment between these two subsets. Middle: ICP + SNADHs with match threshold = 0.8, resulting in the best alignment \( \text{err} = 0.17 \) \( m \). Bottom: ICP + SNADHs with match threshold = 0.3, resulting in the worst alignment \( \text{err} = 0.28 \) \( m \). This dataset aligns well for all parameter values, resulting in a very small difference between the best and worst results.

3.9 Top: the initial misalignment between these two subsets. Middle: ICP + SNADHs with match threshold = 0.3, resulting in the best alignment \( \text{err} = 0.28 \) \( m \). Bottom: ICP + SNADHs with match threshold = 0.9, resulting in the worst alignment \( \text{err} = 7.69 \) \( m \). In this case, a high match threshold filtered out too many matches, causing ICP to overoptimize on the remaining few.

3.10 A histogram of the baseline values from the nearest-neighbor RMSE metric, executed using the ground truth alignments. Note that even the ground truth data does not result in perfect 0 scores.
3.11 A histogram of the point pair consistency metric given by equation 3.24. $\bar{X} = 0.51, \sigma = 0.30$. .................................................. 70

3.12 Box plots showing the 10% trimmed nearest neighbor RMSE results for the full-dataset RANSAC experiments. ..................... 73

3.13 A graph of the misalignment obtained from ground truth alignment of 31 overlap subsets obtained by intersecting the ND_2008_00 scan data with itself. .............................. 74

4.1 Two levels of an example octree structure fit to a set of points. Level 0 is the root of the octree, containing a single node which bounds all points. Level 1 is subdivided halfway along each axis. Unoccupied cells are ignored: in this diagram, they are lighter in color. ................................. 86

4.2 A visualization of the normalized point-based distance metric described in section 4.3.2. Top-left: one half of an overlap subset. Top-right: the other half of the overlap subset. Bottom-left: a visualization of the distance metric, computed with respect to the top-right cloud. Bottom-right: the same, computed with respect to the top-left cloud. ................................. 89

4.3 A visualization of per-point normalized distance in foliage. Dark blue indicates a nearby point in the other point cloud; red indicates large distance. ................................. 90

4.4 Top: category 1: structural changes. Middle: category 2: changes detected in the data, but the underlying model has not changed. Bottom: category 3: octree node detected as changed erroneously. 93

4.5 Top: segmented change regions as detected by the point-based technique. Bottom: the same regions, segmented by an octree-based technique. Note that since neighboring nodes were not flagged as change regions, the octree method does not cleanly contain the region. 95

4.6 Left: a small construction fence. Right: a single trash can. Both regions were detected only by the point-based technique. ........... 96

4.7 Pedestrians detected as change regions using the point-based approach. ................................................................. 96

4.8 Top: aligned datasets obtained with two different scanning platforms. Middle and bottom: the datasets, individually. Note the destructive effect of “shadowing” from trees that obscure the building. 98
4.9 Top: change detection using octrees on overlapping cross-time data. Bottom: change detection using the point based metric. While the point-based metric provides better segmentation, the objects themselves cannot be completely segmented without a more sophisticated approach.

4.10 An example of “smeared” data resulting from scanning a moving object. The car in the lower left was captured turning a corner and it stretched beyond recognition, making classification difficult.

A.1 ND_2008_00 - aerial photography ©Google Maps
A.2 ND_2008_01 - aerial photography ©Google Maps
A.3 ND_2010_00 - aerial photography ©Google Maps
A.4 ND_2010_01 - aerial photography ©Google Maps
A.5 chicago_00 - aerial photography ©Google Maps
A.6 chicago_01 - aerial photography ©Google Maps
A.7 chicago_02 - aerial photography ©Google Maps
A.8 chicago_03 - aerial photography ©Google Maps
A.9 chicago_04 - aerial photography ©Google Maps
A.10 chicago_05 - aerial photography ©Google Maps
A.11 cupertino_00 - aerial photography ©Google Maps
A.12 cupertino_01 - aerial photography ©Google Maps
A.13 cupertino_02 - aerial photography ©Google Maps
A.14 cupertino_03 - aerial photography ©Google Maps
A.15 cupertino_04 - aerial photography ©Google Maps
A.16 cupertino_05 - aerial photography ©Google Maps
A.17 sacramento_00 - aerial photography ©Google Maps
A.18 sacramento_01 - aerial photography ©Google Maps
A.19 sacramento_02 - aerial photography ©Google Maps
A.20 san_antonio_00 - aerial photography ©Google Maps
A.21 san_antonio_01 - aerial photography ©Google Maps
TABLES

2.1 A SUMMARY OF THE DATASETS USED IN THIS DISSERTATION. .................................................. 21

2.2 A SUMMARY OF THE DATASETS USED IN THIS DISSERTATION. .................................................. 22

3.1 PARAMETER SWEEP RESULTS: MEANS AND STANDARD DEVIATIONS OF 10% TRIMMED RMSE OF GROUND TRUTH POINT-TO-POINT DISTANCES .................................................. 60

3.2 PARAMETER SWEEP RESULTS: DATASET-NORMALIZED MEANS AND STANDARD DEVIATIONS OF 10% TRIMMED RMSE OF GROUND TRUTH POINT-TO-POINT DISTANCES ...................... 64

3.3 PARAMETER SWEEP RESULTS: DATASET-NORMALIZED MEANS AND STANDARD DEVIATIONS OF GROUND TRUTH ROTATION MATRIX GEODESIC .................................................. 65

3.4 BEST ESTABLISHED PARAMETERS $\tau_m$ FOR POINT MATCHING .................................................. 67

3.5 MEANS AND STANDARD DEVIATIONS COMPARING ICP AND RANSAC FRAMEWORKS UNDER GROUND TRUTH 10 % TRIMMED RMSE ................................................................. 71

4.1 CATEGORICAL RESULTS FROM AUTOMATIC CHANGE DETECTION ........................................ 92
Firstly, I would like to thank my advisor, Dr. Patrick Flynn, for his guidance, advice, and insight throughout my time in graduate school. I would like to thank Dr. Xin Chen for his mentorship and support, and for providing the opportunity to work on the research projects which have defined my graduate career. The Arthur J. Schmitt Foundation has my deep gratitude, for their financial support throughout my years at the university, as does NAVTEQ, for providing the unprecedented data used in this dissertation.

For my friends at Notre Dame and beyond, a heartfelt thank you, for your kindness, support, and company. You are what I miss most about my time as a Domer.

Lastly, and most importantly, I would like to thank my family for their endless love and encouragement, without which I would never have found myself here.
CHAPTER 1

INTRODUCTION

1.1 Motivation

Light Detection and Ranging (LIDAR) scanners are a popular tool for data acquisition in many research areas, such as urban planning, archaeological modeling, object modeling, and human-computer interaction. Improvements in scanning technology, as well as computers’ capability to process larger and larger amounts of data, has led to an increase of research in the area [43, 65, 66, 72, 84, 85], particularly in the field of environment modeling.

LIDAR scanners use optical sensing technology to measure distance to a target surface. To facilitate rapid environment modeling, many scanners consist of one or more sensors taking constant measurements while attached to a rotating apparatus. Given a return distance, as well as the orientation and location of a sensor, points are projected into a Cartesian coordinate space with real-world units before use. Mobile acquisition platforms add a layer of complexity to this model, as they require hardware to continually track location and orientation of the platform, but allow for large areas to be scanned with ease, leading to vast amounts of collected data and expansive environment models.

Collected data is frequently processed before being used in an application, however. Applications in building extraction or autonomous navigation, for instance,
will typically use an approach that attempts to extract structures that describe the scanned scene more succinctly than the point clouds. Applications in other domains, such as urban planning, environment modeling, or visualization, will typically aim to convert the point cloud data into a format more suitable for interaction, creating triangulated meshes of the data based on visibility constraints offered by the scanner or through surface extraction.

There is a non-trivial cost incurred for any of these processes because of the vast quantity of data in modern datasets. This is particularly problematic for urban modeling applications: even a small city, if scanned using current technology, would consist of many hundreds of billions of points. While modeling such a dataset is possible, depending on the techniques and processes employed, such an effort would likely be pointless due to the impracticality of maintenance. A real world dataset will become obsolete in time, as buildings are constructed, renovated, or demolished, and as roads and other items of infrastructure are created or moved. Updating the dataset would require either knowing which portions of the dataset are no longer up to date and reacquiring them specifically, or reacquiring the dataset entirely and comparing the new dataset and the old dataset to determine differences. In either situation, the process of manually examining a large dataset quickly renders maintenance impractical and necessitates a more scalable approach to the problem.

1.2 Problem Statement

The work presented in this dissertation aims to make maintenance of large-scale datasets feasible through the development of automatic change detection and classification techniques. With such a framework, information about changes
detected in the data can be fed into an automatic algorithm for model generation to update the database, or can be given to an artist for manual creation before being re-integrated into a large database. Multiple scans of an area will also serve to reduce unwanted features in the dataset, such as the presence of cars and pedestrians, which can otherwise be difficult to detect and remove in an automatic modeling pipeline. Furthermore, information obtained from alignment can lead us to characterize measurement accuracy drift over the course of one or multiple scans.

However, there are many challenges involved in creating an automated solution to this problem. Firstly, datasets acquired at different times, even using the same acquisition system, will likely suffer misalignment, which can be made worse by the duration of the scan, the amount of time since a strong GPS fix, or even weather conditions. Although the acquisition system we use produces geo-registered points as output, which gives an approximate initialization position for alignment, we find in practice that misalignment is a challenging problem without a simple solution.

Secondly, after the overlapping segments of scans have been successfully aligned, we must determine what has changed. Isolating changes can be problematic for a number of reasons. Scan density will not be uniform, either as a result of the scanner being at a different position and orientation with respect to the scanned surface (such as when driving on the other side of a road), or as a result of comparing scans made with different acquisition setups. In an urban environment, there can be many obstructions that prevent us from obtaining sample points of the environment: vehicles driving on the road or parked on the side of the street can obscure large sections of the streetfront, which may lead us to determine that a surface present in other scans has disappeared or been created erroneously.
Weather conditions can also adversely affect the relative quality of otherwise consistent scans.

The last primary challenge involves classifying the changes that we detect. As mentioned above, surfaces that are present may not be sampled due to obstructions, leading to erroneous change detection. Determining the presence of unimportant changes (between the presence or absence of cars, pedestrians, and other transitory objects) is a large difficulty that we face, but the information that we gain from this classification will help us to reduce noise: if we know that a transitory surface is detected, we should assume that some region changes are unimportant, as areas behind the surface may not have actually been observed. In addition to transitory surfaces, we hope to classify other region changes in broad but helpful categories: whether buildings have been constructed, demolished, or changed, as well as assigning some level of magnitude of significance to the change, to help direct the efforts involved in the updating process toward the most important examples.

In this dissertation, we will present research designed to address the issues of both point cloud alignment and change detection.
CHAPTER 2
DATA

In this chapter we will introduce the technology which enables this research, the platforms used for data acquisition, characteristics of the hardware involved, and information about the means by which this information is correlated to produce geo-registered datasets. We will also introduce terminology for referring to the various datasets and acquisition platforms involved.

In total, we have 21 datasets, obtained with three different acquisition platforms at multiple points in time and in such varied locations as the University of Notre Dame campus, Chicago, Cupertino, Sacramento, and San Antonio.

2.1 Data Acquisition

2.1.1 LIDAR Scanners

The primary technology behind this research is the remote sensing technology known as LIDAR (Light Detection and Ranging). LIDAR, analogously to radar, enables remote sensing by measuring the time of flight of a laser pulse, and using that measurement to determine the distance of the object off which the laser is reflected. An approximate equation for calculating the distance to an object in
meters is:

\[ d = \frac{t_1 - t_0}{2} \times c \]  

(2.1)

Where \( c \) is the speed of light in meters per second, \( t_0 \) is the time at which the pulse was emitted, and \( t_1 \) is the time at which the response is recorded, both in seconds. This equation is only an approximation; real-world systems adjust for error based on the intensity of the measured pulse return, environmental and atmospheric conditions, distance to the scanned object, and many more circumstances.

LIDAR is used in various forms in a wide variety of applications, including urban planning [59], archaeological modeling [45], agricultural monitoring [62], atmospheric research, autonomous navigation [31], and even for artistic purposes [4]. Generally, applications can be broken down into two major categories: aerial and ground-based. In aerial applications, including atmospheric research, it is more common to see higher energy systems to more accurately account for the effects of atmospheric scattering, while in ground-based systems, applications can utilize lower-energy eye safe pulses to obtain measurements.

The data used in this research was acquired with two different LIDAR scanners, both designed for ground-based applications, with different data characteristics and error profiles.

The first scanner, the SICK LMS-291 (henceforth, “SICK scanner”), is shown in figure 2.1. It consists of a stationary housing containing an infrared pulsed laser emitter, a rotating mirror, and photometric sensors for measuring the laser responses and calculating distances. It has a horizontal field of view of 180°, and its mirror can rotate at 75 Hz, scanning with half-degree resolution, resulting in a
maximum of 27,000 measurements per second. The device is capable of performing remote sensing measurements up to 80 meters away, with 10mm accuracy, although surface properties and environmental conditions can affect this accuracy [69].

Although the scanners have no vertical field of view, they are generally sufficient for modeling applications when mounted to a mobile acquisition platform. To compensate for their relatively low point density, and to maximize their visible range, acquisition vehicles typically mount multiple SICK scanners on a single vehicle at different angles, as described in section 2.1.3.
The second scanner, the Velodyne HDL-64E S2 (henceforth, “Velodyne scanner”), shown in figure 2.2, has a very different design than the SICK scanner. Particularly, it consists of 64 laser emitters and 64 laser receivers, organized at 64 evenly-spaced angular divisions along a 26.8° vertical field of view. These emitters and receivers are situated in a housing which rotates between 5 - 15 Hz, allowing for a full 360° horizontal field of view. The large number of lasers, along with a horizontal angular resolution of 0.09°, allows for a far greater point density than the SICK scanner, with collection rates in excess of 1.3 million measurements per second. The primary downside of this mode of collection is decreased precision,
with a reported accuracy of $< 2\text{cm}$ \[77\].

2.1.2 Fusion Process

LIDAR scanners by themselves only return range measurements as distances from the scanner. Using the rotational angle of the scanner (and, in the case of the Velodyne equipment, the particular laser used) a single measurement can be converted into a local coordinate frame with respect to the position and orientation of the scanner. As the acquisition platforms are mobile, the coordinate frame is constantly changing and these measurements are difficult to use in their raw format; to simplify analysis of the data, scan points are projected into a shared coordinate system. For our data, this is a geodetic coordinate system, transforming each measurement into a tuple consisting of latitude, longitude, and altitude (LLA).

This conversion is accomplished via a fusion process that correlates the raw range data with information from other sensors on the vehicle. The primary instrument is a global positioning system (GPS) on the vehicle, although its accuracy can be compromised under a number of conditions, such as dense urban coverage, unfavorable GPS satellite configurations, and weather conditions. To improve the accuracy of such a device, high-precision measurements are obtained at the start and end of each scan, and acquisition vehicles also use an inertial measurement unit (IMU) in conjunction with wheel sensors to reduce drift in the GPS data. Additionally, vehicles are equipped with a camera array that periodically obtains a color panorama of the surrounding environment. This panorama data can be correlated with the other sensory information to produce colored point clouds, resulting in each point containing an additional RGB color. However, as this color
information has not been explored for research purposes, most of the datasets used in this paper are uncolored.

2.1.3 Acquisition Vehicles

The datasets used in this thesis were acquired with three distinct acquisition vehicle configurations, utilizing the two LIDAR scanners mentioned previously and similar hardware for geolocation. The vehicles themselves represent different stages of research into data acquisition, and consequently have varying performance characteristics.

The first of these vehicles, used in 2008, was an early acquisition platform, consisting of three SICK LMS-291 LIDAR scanners mounted on a mobile acquisition platform. The vehicle was capable of obtaining scan data of acceptable density at highway speeds, although for our acquisitions it was driven between 5 and 20 MPH. The scanners are positioned on the vehicle as shown in figure 2.3: one scanner is rotated 16 degrees down from straight forward on the front, and the other two are located on the sides of the vehicle. The side scanners point away from the direction of movement by 45 degrees. The acquisition vehicle was also equipped with a panoramic multi-camera setup that captured color imagery throughout the duration of the scan, although this data was not used during fusion.

The second acquisition vehicle, used in 2010, was an improvement over the previous model; it was equipped with six SICK scanners, twice as many as the 2008 vehicle, in a configuration as shown in figure 2.4. Additionally, the vehicle itself was smaller and capable of being easily navigated down pedestrian sidewalks, thereby increasing its ability to cover ground and get closer to interesting structures. The downside of this decreased vehicle size is that the scanners them-
selves were closer to the ground, resulting in a decrease in scan density at larger distances from the vehicle although the overall scan density had improved as a result of the increased number of scanners.

Lastly, the third acquisition vehicle, referred to as “NAVTEQ True” or simply “True,” eschews the multi-scanner approach in favor of a single Velodyne scanner. This scanner is attached to the rear of the vehicle, and elevated such that it can obtain a full view of the surrounding environment as the vehicle moves. As this is the only setup which uses the Velodyne scanner, it has the highest point output rate at the cost of decreased accuracy.
Figure 2.4. The positioning of SICK scanners on the 2010 acquisition vehicle, from above. The arrow points in the direction of motion. Side scanners are angled 45° away from the direction of motion; front and back scanners are angled up and down with respect to the horizon.

2.2 Datasets

A number of datasets were used in this research, broken down into categories as described below, based on scan date and scan location. Each individual dataset is referred to as a ‘drive,’ referring to a continuous operation of the acquisition vehicle along a pre-planned route for data collection. We refer to the datasets here broadly by the acquisition vehicle with which they were obtained.
2.2.1 2008 Datasets

Two datasets were obtained with the 2008 acquisition vehicle, as described in 2.1.3. These scans, acquired on consecutive days of the Notre Dame campus, are referred to henceforth as “ND_2008_00” and “ND_2008_01.” The total size of the ND_2008_00 scan, from all three SICK scanners, is 95,809,671 points. The total distance traveled by the van is approximately 12 kilometers, and the drive time was approximately an hour. The ND_2008_01 scan followed a very similar path and lasted approximately the same duration; the total number of points in that dataset is 90,039,946.

Aerial views of the vehicle’s paths can be seen in appendix A, figures A.1 and A.2 and examples of the scan quality can be seen in figures 2.5 and 2.6.

2.2.2 2010 Datasets

In July 2010, another two scans of the Notre Dame campus were obtained, using the 2010 acquisition vehicle described in section 2.1.3. Thanks to the smaller size of this acquisition platform, the vehicle was capable of traversing the campus’ pedestrian sidewalk network, and consequently obtained much more comprehensive coverage of the area.

The 2010 data was captured in two sets on the same day: once in the morning, and then again in the afternoon after a recalibration. The first dataset is referred to as “ND_2010_00” and the second is referred to as “ND_2010_01.” ND_2010_00 consisted of 534,066,514 scan points obtained over a scan length of approximately 30km. ND_2010_01 consisted of 315,040,024 points obtained over a scan length of approximately 20km, and included some indoor scan data. An aerial view of the paths of the acquisition vehicle can be seen in appendix A, figures A.3 and
Figure 2.5. Example scan data from ND.2008.00. The point density is not very high, but is relatively consistent over the face of buildings, and precision is high.
Figure 2.6. Example scan data from ND_2008_00. Most of the path traversed by the van was along the campus loop, too far from buildings to sufficiently sample them; this is an example of some of the foliage that is present throughout most of the dataset.
and examples of the data obtained can be seen in figures 2.7 and 2.8. Note that the panoramic camera imagery from this vehicle was correlated with the scan data, resulting in colored point clouds, although this color was not used during processing.

2.2.3 NAVTEQ True Datasets

The vast majority of the datasets used in this research were obtained with the final acquisition vehicle as described in 2.1.3. These scans were obtained in mul-
Figure 2.8. Example scan data from ND_2010_00. As the acquisition vehicle was smaller, we were able to acquire more buildings with ease.
Figure 2.9. Example scan data from the NAVTEQ True system obtained in Cupertino, CA. Top: a residential street. Note the varied reflectance from street paint. Bottom: a shopping mall parking lot.
Figure 2.10. Example scan data from the NAVTEQ True system obtained in Chicago, IL. This is an extracted segment of the data; note the angular nature of the division (top) caused by the scanner position.
tiple urban areas, including Chicago, Sacramento, Cupertino, and San Antonio. In total, they contain 90,558,436,861 points, representing 98.8% of our total data. Although the acquisition vehicle obtained color camera imagery, the imagery was not used in this research project. The detailed breakdown of data sizes and point counts, as well as information on the drives themselves, can be seen in tables 2.1 and 2.2. Note that the dataset sizes are not the raw data sizes, but the size of the data in a partially compressed packet-based format. Examples of the data from the True system in particular can be seen in figures 2.9 and 2.10.

2.2.4 Data Preprocessing

As the majority of the datasets feature an overwhelming amount of point data, almost all point cloud processing techniques perform subsampling to reduce the processing burden. We perform some subsampling, but its primary purpose is to reduce point density in overscanned areas - primarily the ground, which is scanned in very high density by all scanners, owing to its consistent proximity to the vehicle. We perform subsampling by fitting a very fine grid to the data - each grid element is 2cm$^3$ in size. The points within each grid element are averaged together and the result of that averaging replaces all of the points inside the grid element.

A small amount of filtering is additionally used to remove some systematic noise: in the 2008 acquisition setup, two of the scanners obtained measurements on the vehicle itself, creating a small trace of points throughout the entire scan. These points were identified and discarded.
TABLE 2.1

A SUMMARY OF THE DATASETS USED IN THIS DISSERTATION.

<table>
<thead>
<tr>
<th>Drive Name</th>
<th>Drive Date</th>
<th>Distance (km)</th>
<th>Duration (m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>chicago_00</td>
<td>08/11/2011</td>
<td>0.84</td>
<td>52.27</td>
</tr>
<tr>
<td>chicago_01</td>
<td>08/12/2011</td>
<td>7.16</td>
<td>446.93</td>
</tr>
<tr>
<td>chicago_02</td>
<td>08/15/2011</td>
<td>2.47</td>
<td>121.60</td>
</tr>
<tr>
<td>chicago_03</td>
<td>10/27/2011</td>
<td>1.04</td>
<td>115.20</td>
</tr>
<tr>
<td>chicago_04</td>
<td>03/28/2009</td>
<td>8.83</td>
<td>364.8</td>
</tr>
<tr>
<td>chicago_05</td>
<td>08/16/2011</td>
<td>3.75</td>
<td>183.47</td>
</tr>
<tr>
<td>cupertino_00</td>
<td>08/31/2010</td>
<td>2.23</td>
<td>72.53</td>
</tr>
<tr>
<td>cupertino_01</td>
<td>08/31/2010</td>
<td>3.11</td>
<td>85.33</td>
</tr>
<tr>
<td>cupertino_02</td>
<td>09/06/2011</td>
<td>2.53</td>
<td>54.40</td>
</tr>
<tr>
<td>cupertino_03</td>
<td>09/06/2011</td>
<td>3.20</td>
<td>70.40</td>
</tr>
<tr>
<td>cupertino_04</td>
<td>09/06/2011</td>
<td>10.07</td>
<td>240.00</td>
</tr>
<tr>
<td>cupertino_05</td>
<td>09/07/2011</td>
<td>4.30</td>
<td>102.40</td>
</tr>
<tr>
<td>ND_2010_00</td>
<td>06/01/2010</td>
<td>2.77</td>
<td>222.93</td>
</tr>
<tr>
<td>ND_2010_01</td>
<td>06/01/2010</td>
<td>1.88</td>
<td>125.87</td>
</tr>
<tr>
<td>ND_2008_00</td>
<td>10/19/2008</td>
<td>1.24</td>
<td>60.98</td>
</tr>
<tr>
<td>ND_2008_01</td>
<td>10/20/2008</td>
<td>1.01</td>
<td>52.95</td>
</tr>
<tr>
<td>sacramento_00</td>
<td>07/16/2010</td>
<td>2.84</td>
<td>86.40</td>
</tr>
<tr>
<td>sacramento_01</td>
<td>07/16/2010</td>
<td>4.95</td>
<td>137.60</td>
</tr>
<tr>
<td>sacramento_02</td>
<td>05/29/2012</td>
<td>17.85</td>
<td>375.47</td>
</tr>
<tr>
<td>san_antonio_00</td>
<td>07/16/2010</td>
<td>2.87</td>
<td>105.60</td>
</tr>
<tr>
<td>san_antonio_01</td>
<td>10/11/2011</td>
<td>12.64</td>
<td>344.53</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td></td>
<td><strong>97.58</strong></td>
<td><strong>3421.66</strong></td>
</tr>
</tbody>
</table>
TABLE 2.2

A SUMMARY OF THE DATASETS USED IN THIS DISSERTATION.

<table>
<thead>
<tr>
<th>Drive Name</th>
<th>Number of Points</th>
<th>Size (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>chicago_00</td>
<td>1,232,523,368</td>
<td>18.88</td>
</tr>
<tr>
<td>chicago_01</td>
<td>9,556,401,268</td>
<td>146.27</td>
</tr>
<tr>
<td>chicago_02</td>
<td>3,087,970,619</td>
<td>47.30</td>
</tr>
<tr>
<td>chicago_03</td>
<td>1,118,321,640</td>
<td>17.13</td>
</tr>
<tr>
<td>chicago_04</td>
<td>10,813,414,431</td>
<td>165.65</td>
</tr>
<tr>
<td>chicago_05</td>
<td>5,292,997,352</td>
<td>81.06</td>
</tr>
<tr>
<td>cupertino_00</td>
<td>2,191,406,036</td>
<td>33.59</td>
</tr>
<tr>
<td>cupertino_01</td>
<td>3,292,437,831</td>
<td>50.46</td>
</tr>
<tr>
<td>cupertino_02</td>
<td>2,432,144,346</td>
<td>37.27</td>
</tr>
<tr>
<td>cupertino_03</td>
<td>3,110,894,488</td>
<td>47.67</td>
</tr>
<tr>
<td>cupertino_04</td>
<td>9,426,025,970</td>
<td>144.47</td>
</tr>
<tr>
<td>cupertino_05</td>
<td>4,239,351,387</td>
<td>64.97</td>
</tr>
<tr>
<td>ND_2010_00</td>
<td>534,066,514</td>
<td>8.41</td>
</tr>
<tr>
<td>ND_2010_01</td>
<td>315,040,024</td>
<td>4.98</td>
</tr>
<tr>
<td>ND_2008_00</td>
<td>95,809,671</td>
<td>6.60</td>
</tr>
<tr>
<td>ND_2008_01</td>
<td>90,039,946</td>
<td>6.10</td>
</tr>
<tr>
<td>sacramento_00</td>
<td>2,445,946,405</td>
<td>75.13</td>
</tr>
<tr>
<td>sacramento_01</td>
<td>4,078,690,532</td>
<td>125.21</td>
</tr>
<tr>
<td>sacramento_02</td>
<td>13,663,471,552</td>
<td>209.45</td>
</tr>
<tr>
<td>san_antonio_00</td>
<td>2,662,139,716</td>
<td>40.83</td>
</tr>
<tr>
<td>san_antonio_01</td>
<td>11,914,299,920</td>
<td>182.67</td>
</tr>
<tr>
<td>TOTAL</td>
<td>91,593,393,016</td>
<td>1514.10</td>
</tr>
</tbody>
</table>
This chapter details the research that we have performed in the area of point cloud alignment, sometimes referred to as point cloud registration, including detailed analyses of various experimentation techniques and evaluation metrics. We also discuss the importance of point cloud alignment in the context of a change detection framework, and make direct comparisons to existing research in the field.

Point cloud alignment is a widely-researched subject, but many research projects have unique constraints or assumptions that can or cannot be made about characteristics of the data. In the context of geo-registered environmental LIDAR data, we must make our own assumptions about visibility, scan density, and relative alignment. We compare and contrast these constraints and assumptions with a survey of related work in the field, and implement a number of existing research algorithms for direct comparison within our experimental framework.

We also conduct a number of experiments to determine optimal parameters for our alignment algorithms, evaluate different performance metrics, and utilize domain-based knowledge to improve performance and minimize error.

3.1 Motivation

There are many challenges that arise from real-world data that must be overcome to fully realize a functional change detection framework. Although our data
is geo-registered\footnote{See section \ref{sec:geo-registration}. geo-registered points are expressed as a tuple consisting of (latitude, longitude, altitude).} we observe significant registration error between scans of the same area (see figure \ref{fig:registration-error}). The causes of this misalignment are not obvious, although we examine this question in greater detail in section \ref{sec:registration-causes}. Of primary concern, however, is that without properly aligned datasets, it is difficult or impossible to perform a meaningful comparison for the purposes of change detection.

Without a known transformation between the coordinate spaces expressed by two scan datasets, we cannot directly compare their contents. If we eschew the notion of explicitly deriving a transformation between the two coordinate spaces, then we must essentially perform a “patch search” to perform change detection, and select subsets of points from one dataset to compare against the second dataset for an appropriate match. If no appropriate match is found, we might be tempted to assume that the subset represents an object which was not scanned in the other point cloud; but this assumption can be broken by any number of complications, including false assumptions about scan density, visibility, and the quality of our comparison algorithm. Essentially, difficult comparisons become vastly more difficult because we are limited in our ability to make assumptions about the relationships between the datasets; this is a result of conflating point cloud alignment and change detection as a single problem. To avoid this and other difficulties, we choose to focus on finding a solution to the alignment problem and then assuming a proper alignment during the change detection stage.

3.2 Related Work

The subject of point cloud alignment is the focus of a great amount of research, with publications ranging from the 1980s (\cite{10}) to the present (\cite{72}). While pro-
cessing power has increased greatly in that time, complexity of acquisition systems has also increased drastically, and fast, efficient performance is even more important.

Point cloud registration is necessary or desirable in many applications with varying constraints and requirements. Many papers utilize the technique to align multiple range scans from different viewpoints for object modeling ([1, 6, 8, 10, 12, 13, 18, 21, 22, 29, 32, 37, 38, 45, 65, 66, 74, 76, 83]); the camera position for viewpoints may be known ([1, 10]) or approximately known, or entirely unknown ([50, 65, 66]), with solutions increasing in complexity in the same order. In almost all cases, the approach taken is iterative in nature, and requires an ap-
proximate initial positioning to achieve good results. Although object modeling is a very popular application which requires scan data registration, it is required or beneficial in less obvious areas as well. For example, Cotting et al. [20] create a model watermarking technique that operates by applying low-frequency noise to surface patches; to recover this data, it is necessary to align a watermarked object precisely with a reference model. Johnson and Hebert [37] make reference to the necessity of registration in the context of cross-time dataset alignment, which is particularly relevant for our research.

Environment modeling ([43, 50, 72]) and object detection and recognition ([12, 37, 38, 57]) are also popular applications which require registration: in the case of stationary acquisition platforms in particular, a strong alignment technique can be used to bring scans into the same reference frame without human intervention. The wide variety of applications, as well as their input and output data formats, necessitate a more formal problem definition and taxonomy.

3.2.1 Problem Definition

We define an optimal transformation between two point clouds in terms of point correspondences between the two clouds. Given a point cloud \( P \) (the \textit{probe}) and a point cloud \( Q_p \) or surface \( Q_s \) (the \textit{model}), we establish for some subset of candidate points \( P_C \subset P \) a list of corresponding points \( Q_c \subset Q_p \) or \( Q_c \subset Q_s \), \( |Q_c| = |P_c| \), such that for each \( p_i \in P_c \), the corresponding \( q_i \in Q_c \) is by some metric the ‘best match’ to \( p_i \). We use these correspondences to generate a transformation that consists of a \( 3 \times 3 \) rotational matrix \( R \) and translational component \( t \) such that when applied to \( P \), an error metric \( E \) between this transformed source and
the model is minimized:

\[ E(R \ast p_i + t, q_i), \forall (p_i \in P, q_i \in Q) \]  

(3.1)

This error metric is typically, but not necessarily, Euclidean distance. We define point cloud alignment as the process of ultimately computing the components of this transformation, \( R \) and \( t \), which by definition constitute a rigid transformation. For simplicity, we generally combine \( R \) and \( t \) into a single \( 4 \times 4 \) homogeneous transformation matrix:

\[
T = \begin{bmatrix}
R_{00} & R_{01} & R_{02} & t_x \\
R_{10} & R_{11} & R_{12} & t_y \\
R_{20} & R_{21} & R_{22} & t_z \\
0 & 0 & 0 & 1
\end{bmatrix}
\]  

(3.2)

with points gaining a fourth component of 1, allowing the rigid transformation to be applied with a single matrix multiplication. For more information on homogeneous coordinate systems, see [63].

Computation of point correspondences is a challenging subproblem; for a point cloud model, variations in density between the probe and the model may mean that some points \( p_i \in P \) do not have true matches in \( q_i \in Q_p \), and sampling noise in \( P \) and \( Q_p \) further complicates matters. For a surface model this is simpler, as we often assume that \( P \) is a sampling of \( Q_s \), and consequently that every point \( p_i \in P \) can derive a matching point \( q_i \in Q_s \).

In object recognition applications ([12, 37, 38, 57]), the model is generally given as a surface description, since it is unlikely that one would not have a reference model for a query object. Any application that can use reference objects for
matching or registration has the distinct advantage of being able to derive noise-free low- or high-order surface approximations at any point along that surface. Many approaches that have point clouds as models can also be made to work with surface-based techniques; if the point cloud is given in a structured format (such as from a single range image), a solid surface can be generated simply using neighborhood information from that range image. Although surface models resulting from this technique are susceptible to noise, they are typically benefitted by the added flexibility of such structured representation.

In addition to information about point positions, some applications have color or luminance information that is added to the input dataset as extra dimensions. In [39] and [29], the authors utilize registered color camera imagery to improve alignment between datasets. They incorporate this information by searching for correspondences in 6D, with space constituting the first three dimensions and color constituting the last three. They obtained better results using HSV as a color space than with RGB. Weik [83] utilizes camera imagery as well, but only utilizes luminance; though saturation and value likely increase the discrimination of neighborhood computation, the author notes improved performance from just that one extra dimension.

3.2.2 ICP-Based Approaches

The most popular approach to point cloud registration is the framework provided by the Iterative Closest Point algorithm [9]. While it is very simple as a framework, it can be extended and tweaked in a variety of ways to handle more suitably various desired properties in its input or output. At a high level, it functions as follows: a set of point correspondences is established between the probe
and the model. Then a transformation is computed which minimizes the distance
between the corresponding points in each pair, and this process iterates until the
change in error between steps falls below a threshold or the total error between
the two sets is small enough, or a maximum number of iterations is exceeded.

Rusinkiewicz and Levoy [64] provide an excellent taxonomy for categorizing
the many ICP variations that exist, which we adopt for this discussion. They
divide the variety of different ICP approaches into 6 main steps:

1. Control point selection

2. Matching (searching for correspondences)

3. Weighting correspondences based on a goodness criterion

4. Rejecting outlying points

5. Computing an error metric

6. Minimizing that error metric

Step 1 is concerned with reducing the number of points from the probe for
which we wish to seek a corresponding point in the model data; these points are
commonly referred to as the control points. In many cases, this sampling is done
based on a regular grid based on range image structure to reduce computation
time ([6] [45]), but many approaches use a more complex approach to sampling
[18] [26] [76] by avoiding points which lie in flat or featureless regions. This is similar
to the use of features for the matching step, which we will address below. Zhang
[90] and Jost and Högli [41] use a multi-resolution approach, initially performing
alignment with thinned versions of the meshes and then using a denser version for
finer alignment.
Step 2 involves the selection of a corresponding point on the model for each of the control points selected from the probe in the previous step. This is arguably the most interesting step to examine in terms of variance, as many innovative algorithms are applied to find appropriate matching points from a model. Some earlier papers ([2, 9]) operate under the assumption that sampling density of an object will be high enough or similar enough between scans that a Euclidean nearest-neighbor search is sufficient to establish correspondence. Although many papers improve upon this, it is the simplest method for establishing correspondence, and the Euclidean distance between a point in the probe and a point in the model data is frequently used as part of a metric in many applications. Consequently, most implementations optimize this search through the use of acceleration structures; kd-trees [7] are a popular choice. It has also been shown [70] that point reordering and caching can be used jointly to further improve the computational complexity of such searching.

Instead of searching for a matching point in the model relying on just distance, however, most papers opt for a more sophisticated method in an attempt to match surface features. The simplest feature to match is a first-order approximation of the surface, in addition to the points themselves. Johnson et al. ([39, 40]) formulate a feature descriptor called a spin image, which has the distinct advantages of being rotation and scale invariant. The name comes from the technique used to intuitively describe how the feature is obtained: a two-dimensional histogram is formed by spinning a plane oriented perpendicular to the surface normal at a given point and accumulating points that pass through it. Another advantage of this technique is that it does not make any assumptions about the structure of the probe or the model data; it uses unorganized point clouds for both stages, although
a solid surface in the model could be used to allow for more accurate calculation of the surface normal at each model point. Many techniques, including recent papers such as [72] utilize range image structure directly for their technique. In [72], the authors present the use of Normal Aligned Radial Features (NARFs) for matching; these features rely on border extraction from the range image as part of their feature description. It is interesting to note that while most applications that utilize range imagery directly are oriented towards object modeling, this paper’s examples are all given in the context of indoor environment modeling, an application area in which initial estimates for point cloud registration can be very difficult to obtain.

Gelfand et al. [27] used a descriptor that attempts to capture curvature at a given point. Called the integral volume descriptor, its value for a given point is a function of the area of a sphere, centered at the point and with parametrized radius, which lies underneath the surface. Rusu et al. ([65, 66]) capture multiple features as a ‘feature histogram’ at each point, including estimates of surface curvature. They consider a point to be interesting, and therefore potentially useful as a feature, if its feature histogram is statistically significantly different from the feature histograms of all other points given the same search radius for neighbors when approximating curvature. They further enhance their method by varying this search radius over multiple discrete steps and saving points whose feature histograms have statistically significantly different values at all steps; they refer to these points as persistent and use them particularly for matching.

However, estimating surface curvature from point data requires what essentially amounts to a second derivative of the original data, and noise is typically amplified through these computations. Mitra and Nguyen [56] provide a sophis-
ticated analysis of the error bounds for normal estimation given potentially noisy data, which is a much more frequently used first-order surface approximation. Like many others, we have used a first-order surface approximation as part of our data processing, but we attempt to smooth this data by performing a second pass which improves the robustness of our normal estimation. We intend to examine a variation of this normal distribution analysis as a potential feature for point cloud alignment in future experiments.

The third and fourth steps in the given taxonomy address the issue of assigning importance to point correspondences and pruning unlikely point correspondences, respectively. Many papers provide variation on the weighting step, e.g. [45], but these choices are typically related to or rendered insignificant by an intelligent feature set selection. Johnson and Hebert [37] present an innovative approach to potentially conflicting point correspondences: once point correspondences have been created, they are grouped into conflicting sets, where points within a set can be aligned to some relatively small error. These alignment errors are computed for all sets, and the transformation and set that result in the smallest error are used at that step.

There is much variation in the fifth step, which chooses the error metric that optimization will attempt to minimize. The most popular metric is simply minimization of the Euclidean distance between corresponding points in the scan and model. However, as mentioned previously, this may not be optimal; noisy point matches can cause non-optimal solutions and prevent the algorithm from reaching convergence. Chen and Medioni [18] introduce the point-to-plane metric, which seeks to optimize the distance from the probe point to a first-order approximation of the model at the matching point. This technique is expanded upon by [57, 61].
although both of these papers insist that the model must be a solid surface as opposed to a point cloud to obtain better results. Masuda [51] follows neither of these approaches, and optimizes using a squared distance field defined with respect to an implicit model surface. [52] and [53] are both interesting as well, because they introduce an ICP variant that seeks to maximize not the least-squares error metric, but the least median of squares (LMedS) estimator. This has the advantage of making the algorithm much more robust to noise.

The last variable step in this ICP taxonomy involves the minimization of the given error metric. These error metrics are typically minimized with elegant solutions, such as the dual-quaternion rotation method used in [89], the closed-form quaternion transformation of [33] or a closed-form orthonormal solution ([34]). Some approaches, however, approach the issue of computing the optimal transformation differently. Makadia et al. [50] break the computation of the optimal transformation into two distinct transformations, one of which is purely rotational and the other purely translational. To compute the rotational component of the transformation, they compute extended Gaussian images (EGIs) for each point cloud involved and attempt to find a rotation by matching features in that domain. Fitzgibbon [24] applies the Levenberg-Marquardt algorithm, a non-linear minimization technique that interpolates between gradient descent and Gauss-Newton minimization, for its simplicity, and achieves competitive results.

3.2.3 Non-ICP Approaches

Although ICP is such a popular approach that it is almost a necessity for registration, some techniques do not use it. Random Sample Consensus (RANSAC), in particular, is a popular choice for alignment purposes. RANASC operates under
the principle that only a very limited subset of point need contribute to an optimal rigid transformation. By randomly selecting sets of three points and searching for correspondences, it is possible to identify the optimal transformation while ignoring the effects of outliers, presuming that the optimality of the transformation can be verified.

Chen et al. [12] perform an interesting alignment technique using RANSAC and a triangulated version of the input mesh obtained from the range image structure. If the source matches the data well after transformation (each source point has a nearby neighboring point), then the alignment is presumed to be good and is used. Otherwise, RANSAC continues sampling points that meet its selection criteria until such a transformation is identified. However, this paper operates under the assumption that scan density is consistent over the surface of an object and between scans – this is necessary for points to match nicely with correspondences after transformation – which is unlikely in practice, and particularly unlikely in the context of environment modeling.

Pottman et al. [60] formulate pairwise point-cloud registration as an instantaneous kinematics problem, computing a rigid body transformation that most closely approximates an optimal but non-linear transformation of data to the model. Ultimately, their method is similar to ICP, though: it establishes point correspondences using second-order surface approximations to compute point-to-surface distances, and iterates a rigid body transformation until a convergence is reached.
3.3 Primary Methodology

In this section we will describe in detail the primary methodology by which our experiments are conducted. This entails a description of how data subsets are extracted for specific experiments, details of the alignment frameworks used and the particular point features used in those frameworks, as well as a brief discussion of helpful heuristics derived from domain knowledge.

3.3.1 Overlap Subsets

Fully aligning the data obtained from two large LIDAR scans is a difficult problem, but it is further exaggerated by GPS drift, resulting in accumulated error over the course of a single scan. Because the drift may be affected by many factors, including properties of the GPS device, quality of the GPS satellite fix, weather, and calibration of the inertial measurement units, it is likely impossible to align two sufficiently long scans of data with a single linear transformation. To simplify our problem, we isolate specifically the overlapping areas between scans and align those individually using linear transformations. We define an overlap subset for two datasets, $P$ and $Q$, as the set of points $O_P \in P$ and $O_Q \in Q$ that surround a shared geodetic reference point, $r$:

$$O_P = \{ p \in P \mid \text{dist}(p, r) < \epsilon_{\text{overlap}} \} \quad (3.3)$$

$$O_Q = \{ q \in Q \mid \text{dist}(q, r) < \epsilon_{\text{overlap}} \} \quad (3.4)$$

We obtain these reference points $r \in R$ by using information about the path of the acquisition vehicle from each scan (see figures A.1 and A.2). For each
vehicle, we have or are able to extract a set of points representing the path of the vehicle throughout a scan, containing the current timestamp and the van’s latitude, longitude, and altitude. If a point from the path of the first scan is within $\epsilon_{\text{overlap}}$ of a path point from the second scan, the average of those points is added to our list of reference points, $R$. Once all overlap subsets between two scans have been extracted, an extra pass ensures that the set of overlapping subsets is expressed as compactly as possible: consecutive overlap subsets are combined. For efficiency reasons, large overlaps, which can occur when both paths follow the same long stretch of road, are split into smaller consecutive ‘runs’ to reduce the computational burden associated with individual experiments.

3.3.2 Problem Statement: Overlap Subset Alignment

The work that we present in this chapter deals primarily with our research in the area of overlap subset alignment. We begin with a formal definition of the overlap subset alignment problem.

Consider two sets of geo-referenced point cloud data, $P$ and $Q$, which together form an overlap subset as previously defined in equations 3.3 and 3.4. These data sets represent point samplings of underlying surface models, $\psi_P$ and $\psi_Q$ respectively, expressed in a geodetic reference frame, that have been transformed by an unknown rigid misalignment, $T_P$ and $T_Q$. In other words, $P$ is a point sampling of $T_P(\psi_P)$ and $Q$ is a point sampling of $T_Q(\psi_Q)$. Arbitrarily, we fix $\psi_Q$ (and consequently $Q$) as our reference surface model, and consider $P$ to be a sampling of $T_P(T_Q^{-1}(\psi_Q))$.

The overlap subset alignment problem, then, consists of finding the transformation $T = T_Q * T_P^{-1}$. We assume that the transformation between the two
overlap subsets is rigid\(^2\) and can be expressed with a translational component \(t\) and a rotational component \(R\), making its inverse trivial to compute. We compare Iterative Closest Point (ICP) (see section 3.3.3) and Random Sample Consensus (RANSAC) (see section 3.3.4) frameworks to approximate this transformation. One difficulty of applying this framework directly to our input data is that the surface models \(\psi_P\) and \(\psi_Q\) are not known; rather, they are implicit in the samplings we obtain as \(P\) and \(Q\).

As we do not have the underlying surface models \(\psi_P\) and \(\psi_Q\), we redefine our optimal transformation and error metric in terms of the point cloud samples themselves. If we assume that \(P\) and \(Q\) contain measurements of the same underlying surface model, we can define optimal alignment in terms of distance between points in \(P\) and \(Q\) which are samples of the same surface point. In other words, for a Euclidian distance function \(d\), we seek the transformation \(T\) that minimizes

\[
d(T(p_i), q_j) \quad \forall (p_i \in P, q_j \in Q) \tag{3.5}
\]

where \(p_i\) and \(q_j\) represent samples of the same point on a shared underlying surface model. Again, since we do not have knowledge of \(\psi_P\) or \(\psi_Q\), we attempt to estimate whether two points \(p_i\) and \(q_j\) represent the same surface point through the use of descriptors that encapsulate knowledge about the surface around these points. If the descriptors match, and other thresholds are met - such as the distance between the two points falling between reasonable bounds - then we say that these two points likely represent samples of the same surface point.

\(^2\)Visual inspection of overlap subsets reveals that a rigid transformation is sufficient to bring real-world data into alignment. For more discussion on the source of misalignment, see section 3.7.1
3.3.3 Alignment using ICP

We perform alignment using ICP with a version of the algorithm as described in section 3.2.2. Algorithm 1 contains high level pseudocode for ICP. Given two point clouds, \( P \) and \( Q \), a convergence threshold \( \tau_{\text{err}} \), an iteration limit \( \tau_{\text{iter}} \), the function computes a transformation, \( T \), which optimally aligns \( P \) to \( Q \) according to some error metric.

**Algorithm 1** A high level description of the Iterative Closest Point algorithm.

```plaintext
numIterations ← 0
T ← I

loop
  C ← findBestCorrespondences(\( T \ast P, Q \))
  T ← computeOptimalTransformation(C)
  err ← computeErrorMetric(\( T \ast P, Q \))
  numIterations ← numIterations + 1
  if err < \( \tau_{\text{err}} \) or numIterations > \( \tau_{\text{iter}} \) then
    return T
  end if
end loop
```

We select a set of point pair correspondences as follows. For each point in the probe, we look for its closest neighbor in the model closer than a maximum search distance, according to one of the point features described in section 3.3.6. This restriction is primarily for optimization purposes, since we assume a limit on misalignment of \( \approx 10 \)m, but it also helps to avoid erroneous correspondences. We then prune these correspondences by looking at the distribution of distances between points in a pair: if a distance is greater than two sigmas away from the
mean distance, that pair is removed from consideration.

After we have established a set of point correspondences we compute a rigid transformation that minimizes the distance between corresponding points in a pair. We compute this transformation using the dual-quaterion technique as described in [33], which computes a $4 \times 4$ transformation matrix $R$ such that the error between corresponding points in a pair $(p_i, q_i)$ is minimized:

$$\min_R \| R * p_i - q_i \|$$ (3.6)

We then apply this transformation to all points in the probe and repeat this process until we reach convergence. In our algorithm, converge is reached when the total number of iterations is greater than a threshold or the maximum distance any single point is moved by the transformation falls below a threshold. Given that our initial point clouds have relatively good starting positions, our experimental results show that ICP tends to converge in just a few iterations (see section 3.6).

3.3.4 Alignment using RANSAC

We also utilize Random Sample Consensus (RANSAC) for our alignment experiments. A very simple technique, RANSAC works using a guess-and-check method, as seen in the pseudocode in algorithm 2. We begin by choosing three or more points from our probe at random. For each of those points, we then search the model for its closest neighbor closer than a maximum search distance, using the point features described in section 3.3.6. As with ICP, this maximum distance restriction is primarily for optimization purposes, but serves to reduce the number of unhelpful attempts given our ability to estimate maximum misalignment.

Once we have established a small set of point correspondences, we compute
Algorithm 2 A high level description of the RANSAC algorithm.

bestT ← I

bestError ← inf

numIterations ← 0

loop

PC ← randomSubset(P, 3)
QC ← findBestCorrespondences(PC, Q)

if |QC| ≠ |PC| then
    continue
end if

T ← computeOptimalTransformation(PC, QC)
err ← computeErrorMetric(T * P, Q)

if err < bestError then
    bestError ← err
    bestT ← T
end if

numIterations ← numIterations + 1

if bestError < τerr or numIterations > τiter then
    return bestT
end if

end loop
the transformation that minimizes the distance between them. If we choose three point pairs, we can attempt to compute a rigid transformation explicitly, although it is likely that perfectly aligning point correspondences will result in a non-rigid transformation. For this reason, we generally use the dual quaternion transformation mentioned in section 3.3.3 to minimize the error between points in a pair under the rigidity constraint.

Once we have this transformation, \( T \) we apply it to all points in the probe and compute a metric that captures how well the transformation brings the two point clouds into alignment. For each transformed point in the probe \( T(p_i) \), we compute the distance to its nearest neighbor in the model. We trim the top and bottom five percent of this distribution to remove outliers, and use the root mean squared error of the remaining distance samples as our quality metric. If this value is below a threshold \( \tau_{err} \), we terminate RANSAC immediately. Otherwise, we keep track of the (error, transformation) pair with the lowest error and return that after we exceed a maximum number of RANSAC iterations, typically set around 500.

Although RANSAC has been used in large-scale point cloud alignment literature ([66]), it is generally less popular than ICP because at each iteration of the RANSAC algorithm, we must perform a costly point-to-point distance query to determine how well a given tuple brings two point clouds into alignment. For large datasets, like environmental scan data, this can quickly become a limiting factor, as choosing points at random may take hundreds or thousands of iterations to converge. To combat this, we only compute the error associated with a transformation for a random, small subset of points - around 10% - which we assume to be representative of the overall error.
3.3.5 Domain Knowledge and Heuristics

Although research techniques aim to be as general as possible, we find that we are able to vastly improve the performance and simplicity of our framework by acknowledging aspects of domain knowledge about our datasets. The first and most obvious of these observations relates to the most prevalent structure in all of our datasets: the ground. The ground is overscanned. In one representative dataset, we found that the average point density for points on the ground was \( \approx 2170 \frac{pt}{m^3} \), whereas for other points it was \( \approx 860 \frac{pt}{m^3} \). This is due to the ubiquity of the structure in the scanning environment, but also due to the placement and orientation of the scanners. See section 2.2.4 for more information on the point density of the ground region.

We use this information to improve our approach in two ways. First, we use the average altitude of the ground plane to quickly improve our initial alignment estimate between two datasets. We extract all points belonging to the ground plane via a simple clustering algorithm \[23\] executed on the set of points whose surface approximation normals point within a few degrees of the global up vector. In our experimental datasets, these extracted points exhibited a negligible amount of variation in altitude, leading us to term the set the ‘ground plane’ and associate with it an average altitude value. Before the actual alignment algorithm takes place, we apply the difference of these altitude values to the probe to improve our starting position. For an average ground altitude in the probe as \( A_p \) and average ground altitude in the model as \( A_q \), each LLA point in the probe is updated:

\[
\forall p_i \in P, \quad p_i = p_i + [0, 0, A_q - A_p]^T
\]

To assess the effectiveness of this approach, we measured the ground truth
(translational) vertical misalignment between two datasets for a number of overlap subsets, \( \delta_{GT} \). We then recorded the detected ground plane altitude differences for these subsets, \( \delta_A \). Figure 3.2 shows the distribution characterizing the difference between these two datasets, \( \text{abs}(\delta_{GT} - \delta_A) \). In a set of experiments conducted on 50 manually-labeled overlap subsets, the ground plane pre-alignment stage predicted vertical misalignment with an average error of 0.85m (\( \sigma = 0.52m \)) - this is impressive considering the magnitude of vertical misalignment seen in our data; the largest such misalignment was recorded at 9.55m. We term the application of this offset the \textit{ground plane pre-alignment step}, and given its effectiveness, we perform it before any and all alignment experiments.

Lastly, because after this step we know that the vertical misalignment between two point clouds is very small, all neighborhood queries used when searching for point matches in both the ICP and RANSAC variants of the alignment algorithm use a cylindrical neighborhood query instead of a spherical neighborhood query. This has the effect of minimizing false positives that are likely incorrect matches anyway. We typically choose our search cylinder to have a height of 10% of its radius (e.g., for a search radius of 10m, we would use a cylindrical search with a height of 1m and radius of 10m).

We also use the extracted ground plane to improve our efficiency in another manner. The ground plane, as previously mentioned, is largely flat and featureless. Since we expect points in the extracted ground region to have very similar point features, and therefore to not meaningfully produce point matches for alignment purposes, we remove the points in the ground region from consideration before performing point matching during both ICP and RANSAC alignment algorithms. In addition to reducing the number of points on which our algorithms are executed,
Figure 3.2. A histogram of the altitudinal alignment error ($n = 50$) obtained by comparing ground truth alignment with automatic ground plane pre-alignment. The mean value is $0.85m$ with $\sigma = 0.52m$. 
this also helps to reduce the likelihood of false matches with similar points in the
ground region, and therefore improve the quality of the resulting transformations.

3.3.6 Point Features

When finding point correspondences for alignment, we seek the ‘best match’ in
the model for a given point in the probe. Although the prototypical ICP approach
uses Euclidean distance alone to determine the best match for a given point,
we do not rely on this metric alone due to the potential for density differences
between datasets. Instead, we utilize similarity between point descriptors (or point
to features), which serve to capture the shape of a point’s neighborhood in a way that
is suitable for comparison. Primarily, we are concerned with both translational
invariance and density invariance in our selection of point features. Rotational
invariance is less important in our application, although generally speaking this
is only because our datasets exhibit a small amount of rotational variance.

Below we describe in detail the four different point features which we have
examined, the first of which is a novel point descriptor based on our prior research
in the area of object extraction [85]. The remaining point descriptors are selected
from popular literature and are used in the evaluation of our algorithms.

3.3.6.1 Surface Normal Approximation Distribution Histograms

The first point feature that we describe is a novel feature based on our prior
work in the area of object extraction [85, 86]. In that work, we sought to char-
acterize the ‘planarity’ of a neighborhood of points by examining the variance of
surface normal approximations with a two pass method: in the first pass, each
point is assigned a normal based on the best-fit plane to its neighborhood using
principal components analysis (PCA) [25]. In the second pass, each point is assigned a value based on the variance of angle differences between surface normals in the neighborhood.

Here, we use a slightly more sophisticated variation of this technique to develop an original point-based feature. Instead of a single value that measures the variance of surface normals in a neighborhood, we create a histogram that characterizes how the surface normals vary. For a set of points, $P$, and associated normals $N$, we compute a surface normal approximation distribution histogram (SNADH) for a point $p$ by first obtaining the average normal for the points in $p$’s neighborhood, $N_p$:

$$
\bar{n} = \frac{\sum \overrightarrow{n_i}}{|N_p|} \quad (3.8)
$$

Since the surface normals may not be aligned in a consistent direction, we arbitrarily set the direction to that of the first surface normal, and flip the sign of each consecutive normal if the dot product with that normal is negative. Then, for each point $q$ in the neighborhood, we compute two angles using its normal, $\overrightarrow{n_q}$: $\phi$, the angle between $\overrightarrow{n_q}$ and $\bar{n}$, and $\theta$, the angle representing its rotation about $\bar{N}$. See figure 3.3 for an illustration of these angles. We then use these to calculate an index into a two-dimensional histogram, using the knowledge that $\phi \in [0, \pi/2]$ and $\theta \in [0, 2\pi]$. In the majority of our experiments, we set the number of bins in the $\phi$ dimension ($\phi_N$) and the number of bins in the $\theta$ dimension ($\theta_N$) to 16, striking a balance between resolution and discriminative capacity.

After creating the histogram, it is normalized to counter the effects of varying point density. Additionally, we notice that for small values of $\phi$, the value of $\theta$ is less meaningful. To account for this, when performing matching we assign a
weight to each zero-indexed row of the $\phi$ dimension of the histogram:

$$w_i = \frac{2(i + 1)}{\phi_N(\phi_N + 1)},$$  \hspace{1cm} (3.9)

where $\phi_N$ is the number of bins along the $\phi$ dimension of the SNADH, and

$$\sum_{i=0}^{\phi_N-1} w_i = 1. \hspace{1cm} (3.10)$$

An important note to make is that this operator is not, by itself, rotationally invariant - the offset along the theta direction is dependent on the choice of in-plane axes perpendicular to the normal, which is relatively robust. To combat this, we select the axis corresponding to $\theta = 0$ which is the dot product of the surface normal and the positive X axis. Before performing matching, we shift the bins along the theta dimension slightly to account for a small amount of rotation; while sliding the bins completely could account for any amount of horizontal rotation, it is also likely to decrease the discriminative capacity of the point feature and increase computation time. To standardize the $\phi$ dimension of the histogram, we flip the normal if the dot product with the global up vector is negative.

Example histograms can be seen in figure 3.4. The first image, on the left, is an example SNADH for a point which lies in a planar region. The vertical axis corresponds to the $\phi$ dimension and shows that the magnitude of normal variance is typically low, although normals are spread evenly across the $\theta$ dimension. In the middle is an example SNADH for a point lying in foliage. The distribution is varied widely along both axes. Lastly is an example SNADH for a point on the intersection of two planar regions – certain neighborhood distributions can manifest as distinctive patterns in a SNADH.
Figure 3.3. With respect to an point $p$ and average normal $\bar{n}$, SNADHs use a mapping that transforms each neighboring point into a pair of values $(\theta, \phi)$, whose values capture the rotation of $n_q$ about the reference normal and rotation away from the reference normal, respectively.

Figure 3.4. Example SNADHs showing distinctive patterns.
3.3.6.2 Spin Images

Spin images are widely utilized point features that capture a translation- and rotation-invariant description of a point’s local neighborhood in a 2D image [37, 40], similar to SNADHs. The spin image for a given point is obtained via a function known as a spin map, which is defined with respect to that seed point. This mapping transforms each 3D point in its neighborhood into a pair of numbers, corresponding to distances along (β) and perpendicular to (α) the seed point’s surface normal, as seen in figure 3.5. Each point within a neighborhood is transformed using the spin map, and using bounds established as a function of the object’s sampling density, binned into a 2D histogram that constitutes the spin image. The original algorithm chooses the number of bins as a function of the sampling density and a user-specified bin size in each dimension, but since sampling density is unknown or inconsistent in our application, we provide the number of bins as a parameter and use that and the neighborhood size to determine the bin size.

3.3.6.3 Fast Point Feature Histograms

Fast Point Feature Histograms, introduced in [66] as an improvement over previous work [65], attempt to capture the shape of a neighborhood statistically, by creating a histogram based on the relationships between pairs of points in a neighborhood. For each pair of points in a neighborhood, \( p_i \) and \( p_j \), and their respective normal approximations, \( \hat{n}_i \) and \( \hat{n}_j \), their relationships are captured with respect to a Darboux frame \( \langle \hat{u}, \hat{v}, \hat{w} \rangle \) which is given by:
Figure 3.5. With respect to an oriented point $p$, spin images use a mapping that transforms each neighboring point into a pair of values $(\alpha, \beta)$, which correspond to the distance perpendicular to and parallel to the normal $n_p$. 
\(\vec{u} = \vec{n}_i\)
\(\vec{v} = (p_j - p_i) \times \vec{u}\)
\(\vec{w} = \vec{u} \times \vec{v}\)

The measured quantities are then:

\[\alpha = \vec{v} \cdot \vec{n}_j\]
\[\phi = (\vec{u} \cdot (p_j - p_i))/\|p_j - p_i\|\]
\[\theta = \arctan(\vec{w} \cdot \vec{n}_j, \vec{u} \cdot \vec{n}_j)\]

A visualization of the coordinate frame used for FPFH computation can be seen in figure 3.6. For each pair of points in a neighborhood, a tuple \((\alpha, \phi, \theta)\) is computed and binned in a 3-dimensional histogram. For \(n\) bins along each dimension of the histogram, we have \(n^3\) total histogram elements. Generally speaking, we set \(n\) to be 5 for most of our experiments, resulting in histograms with \(5^3 = 125\) bins.

Generally speaking, though, the strict computation of the FPFH for each point is intractable: for \(n\) total points, and \(k\) points in a neighborhood, the complexity is \(O(n \ast k^2)\), not counting the cost of neighborhood queries. To counter this, the authors suggest first computing a simple point feature histogram (SPFH) for each point, and then combining SPFHs within a neighborhood to produce an FPFH approximation. We follow this approach.

Whereas the FPFH for each point \(p\) is obtained by obtaining a histogram of the \((\alpha, \phi, \theta)\) tuples for all pairs of points in \(p\)’s neighborhood, the SPFH only examines the relationship of \(p\) to every other point in its neighborhood. Thus
Figure 3.6. For each pair of points $p_i$ and $p_j$ within a neighborhood, Fast Point Feature Histograms are computed with respect to a Darboux frame as shown here.

The computational complexity of the SPFH is $O(n \times k)$, the cost of neighborhood queries again notwithstanding. Once the SPFH has been computed for every point, a second pass over the dataset computes the FPFH approximation for each point $p$ as a weighted sum of the SPFHs in its neighborhood:

$$FPFH(p) = SPFH(p) + \frac{1}{k} \sum_{i=1}^{k} \frac{1}{w_i} \cdot SPFH(p_i)$$

(3.11)

where $w_i$ is the Euclidean distance between $p$ and $p_i$. We use this final approximation as the FPFH for every point in a dataset.

3.3.6.4 3D Shape Contexts

The last point descriptor we use is the 3D Shape Context, as defined in [44]. Simply, the 3D Shape Context captures the spatial distribution of points in a
neighborhood in spherical coordinates. A two-dimensional analogue of the 3D Shape Context can be seen in figure 3.7. The range of spherical coordinates is divided into sectors and shells as a 3D histogram; to diminish the effect of points further from the center of the neighborhood, the shells are spaced according to an equal-volume distribution. For a maximum radius \( r_n \) and \( \rho_k \) equal-volume shells, the radius of zero-indexed shell \( i \) is given as:

\[
    r_i = r_n \times \sqrt[3]{\frac{i + 1}{\rho_k}}
\]  

(3.12)

The indices of the sectors along \( \phi \) and \( \theta \), for a point in spherical coordinates \( p = (\phi_p, \theta_p, \rho_p) \) and \( \phi_k \) and \( \theta_k \) bins are then:

\[
    \phi_i = \left( \frac{\phi_p + \pi/2.0}{\pi} \right) \times \phi_k
\]  

(3.13)

\[
    \theta_i = \frac{\theta_p}{2 \times \pi} \times \theta_k
\]  

(3.14)

As with FPFH, this is a 3D histogram, and we will have \( (\theta_k \times \phi_k \times \rho_k) \) total bins. We typically set the bin sizes to 5 to keep overall bin count manageable. As with SNADHs, 3D shape contexts are not rotation-invariant; in order for two shape contexts to match, their neighborhoods must be defined with respect to the same coordinate frame. We account for this in the same way as SNADHs: we choose a unified normal direction via a dot product with the global up vector, and we fix the first in-plane vector with a cross product of the normal and the positive X axis. To account for some small rotation, we slide the 3D shape contexts along their \( \theta \) axis slightly when performing matches.
Figure 3.7. A two-dimensional shape context, showing both radial and angular divisions. The 3D Shape Context is analogous. On the left, radial bins are divided evenly; on the right, they are divided such that each shell has equal volume.
3.3.7 Point Matching

Although SNADHs and Spin Images result in two-dimensional histograms, and FPFHs and 3D shape contexts result in three-dimensional histograms, we can concatenate the bins of the histograms along each dimension to obtain a single one-dimensional histogram that is representative of any of the point features. As we will always be comparing the histogram computed using one feature against other histograms computed using that same feature and same parameter set, visualizing the comparison of two point features as a comparison between two one-dimensional histograms simplifies the reasoning behind our approach. To account for differences in point density, all histograms are normalized before comparison so that the sum of the histogram bins totals to 1.

To determine how well two points match given their precomputed feature histograms, we utilize the normalized cross-correlation coefficient [30]. For two feature histograms, $F_1$ and $F_2$, of length $n$, we compute their match score as:

$$M(F_1, F_2) = \frac{1}{n-1} \sum_{i=1}^{n} \left| \frac{F_1[i] - \bar{F}_1}{\sigma_{F_1}} \right| \left( \frac{F_2[i] - \bar{F}_2}{\sigma_{F_2}} \right|,$$

(3.15)

where $\bar{F}_1$ and $\bar{F}_2$ are the sample means of $F_1$ and $F_2$, respectively, and $\sigma_{F_1}$ and $\sigma_{F_2}$ are their sample standard deviations. This value lies within the range of $[0, 1]$, representing the linearity of the relationship between the two histograms. If this score is above a certain threshold $\tau_m$, we say that the two points match based on their point features. Naturally, we expect an effective threshold to differ based on the discriminatory capacity of the point feature used; we explore the derivation of such a threshold in section 3.5.
3.4 Metrics

To determine the effectiveness of our algorithms and the various point features that we use, we explore the use of various metrics for evaluating the quality of an alignment. There is a wide variety of approaches toward computing error metrics for point cloud alignment. One popular metric is computed by quantifying the distribution of point to point nearest neighbor distances between the clouds; i.e., for a probe cloud $P$, model cloud $Q$, and transformation $A$, one might seek to characterize the distribution:

$$D = \{d(A*p, Q), \forall p \in P\},$$  \hspace{1cm} (3.16)

where $d(p, Q)$ is the Euclidean distance between $p$ and the nearest point from $Q$. To reduce this distribution to a single value, we take the root mean squared error (RMSE) of the distribution:

$$\text{RMSE}(D) = \sqrt{\frac{1}{|D|} \sum_{i=0}^{|D|} D_i^2},$$  \hspace{1cm} (3.17)

To improve the robustness of this metric, we use a trimmed estimator to remove outliers – we typically remove values which lie in the top and bottom 5% of the distribution.

The primary difficulties with this nearest neighbor point-to-point (NN-PP) metric are twofold: it is affected by sampling density differences between two scans, and it assumes that underlying structures are present in both scans. For enviromental scan data, especially cross-time datasets, neither of these assumptions is necessarily true. However, in absence of ground truth data, this is still likely the best metric we can use: the RMSE will in general be minimized for a
correct alignment. Furthermore, as the units of this metric are in meters, we can gain some intuition about the meaning of the value.

To improve the quality of our evaluation, we also compute metrics that utilize a ground truth transformation between two point clouds. For a probe cloud $P$ and model cloud $Q$, we manually compute a ground truth alignment that best aligns the two point clouds, $G$. We can calculate a Euclidean metric by examining the difference between point locations after the ground truth transformation and after an automatically derived transformation, $A$:

$$D = \{d(A \ast p, G \ast p), \forall p \in P\} \quad (3.18)$$

We term this metric the *ground truth point-to-point (GT-PP)* value or *supervised point-to-point* value, and typically reduce it to a single value by using the RMSE and a trimmed estimator as described above. Although this technique does not compare the generated transformations directly and instead computes a metric based on the effects of the transformations, it has the benefit of having units of meters, making it as intuitive to interpret as the unsupervised point-to-point metric described above.

We also use methods that directly compare the automatically derived transformation and the ground truth transformation themselves. These transformations consist of an orthonormal 3x3 rotation matrix $R$ and a translation vector $\vec{t}$, such that a point $p$ can be transformed by a transformation $A$ as

$$p' = R_A \ast p + t_A \quad (3.19)$$

We can compute the difference between the translational components of two trans-
formations $A$ and $B$ easily, using Euclidean distance:

$$d(\vec{t}_A, \vec{t}_B) = \|\vec{t}_A - \vec{t}_B\|$$  \hspace{1cm} (3.20)

but calculating a distance metric between two matrices is not necessarily as straightforward. The Euclidean norm of the difference between two matrices, also known as the Frobenius norm, is one method:

$$\|A - B\|_F = \sqrt{\sum_{i=0}^{n} \sum_{j=0}^{m} (A_{ij} - B_{ij})^2}$$  \hspace{1cm} (3.21)

although it is difficult to formulate an intuition as to the meaning of such a difference. Wang [82] derives a distance metric that is specifically formulated to express a more meaningful difference between two orthonormal rotation matrices. This distance metric uses the observation that 3x3 orthonormal rotation matrices form an embedding of $\mathbb{R}^3$ in $\mathbb{R}^9$, and computes a geodesic along that embedded hypersurface. The geodesic distance between two rotation matrices $R_A$ and $R_B$ is expressed as:

$$d_G(R_A, R_B) = \|\log (R_A^TR_B)\|_F$$  \hspace{1cm} (3.22)

where $\|\|_F$ is the Frobenius distance mentioned above.

In summary, this gives us two metrics for evaluating the difference between two rotation matrices when a ground truth transformation matrix is available to us. Additionally, when a ground truth transformation is present, we can also utilize the supervised point-to-point metric, and in the absence of any ground truth we can use the unsupervised point-to-point metric described above. Unless otherwise stated, we use a 10% trimmed estimator in conjunction with RMSE for all of these
distributions for our results.

3.5 Parameter Sweeps

There are a number of parameters involved in the alignment process, such as point thinning and neighborhood size. Many of them can be set empirically using sensibly established values: for instance, knowing the maximum amount of misalignment we can expect to see allows us to set an upper bound on the distance we expect to search to find point matches. The primary parameter involved in our experimentation for which there is no clear answer is the threshold used when determining point matches, $\tau_m$ as mentioned in section 3.3.7. Different point features offer differing discriminatory capabilities which are not obvious and which can depend heavily on the qualities of the data used (e.g., point density and scan noise) or the parameters used to calculate the feature itself (e.g., number of bins). To account for these potential variances, we perform a parameter sweep using the aforementioned alignment metrics to determine optimal thresholds for each of the specified point features.

We perform our parameter sweep on a subset of 50 overlap subsets with manually established ground truth alignments. The presence of ground truth allows us to compute a more accurate error as mentioned in equation 3.18 with a perfect alignment having an error of 0m. Additionally, we can utilize the rotation matrix geodesic as in equation 3.22.

Table 3.1 contains the results of the parameter sweep, presented as the mean and standard deviation of the 10% trimmed RMSE of ground truth point-to-point distances (equation 3.18). Immediately apparent is the wide overlap between results for varying parameters, indicating an inability to choose an effective param-
TABLE 3.1

PARAMETER SWEEP RESULTS: MEANS AND STANDARD DEVIATIONS OF 10% TRIMMED RMSE OF GROUND TRUTH POINT-TO-POINT DISTANCES

<table>
<thead>
<tr>
<th>$\tau_m$</th>
<th>3DSC</th>
<th>FPFH</th>
<th>Spin Images</th>
<th>SNADH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>5.12 ± 2.62</td>
<td>5.36 ± 4.30</td>
<td>0.56 ± 0.30</td>
<td>0.57 ± 0.24</td>
</tr>
<tr>
<td>0.3</td>
<td>6.78 ± 2.88</td>
<td>5.67 ± 4.30</td>
<td>0.57 ± 0.30</td>
<td>0.55 ± 0.24</td>
</tr>
<tr>
<td>0.4</td>
<td>3.54 ± 1.47</td>
<td>5.58 ± 4.62</td>
<td>0.58 ± 0.30</td>
<td>0.64 ± 0.43</td>
</tr>
<tr>
<td>0.5</td>
<td>5.28 ± 2.87</td>
<td>7.23 ± 6.02</td>
<td>0.58 ± 0.29</td>
<td>0.54 ± 0.38</td>
</tr>
<tr>
<td>0.6</td>
<td>4.87 ± 2.71</td>
<td>6.37 ± 6.28</td>
<td>0.53 ± 0.25</td>
<td>0.42 ± 0.23</td>
</tr>
<tr>
<td>0.7</td>
<td>4.31 ± 3.13</td>
<td>4.43 ± 3.94</td>
<td>0.51 ± 0.27</td>
<td>0.38 ± 0.23</td>
</tr>
<tr>
<td>0.8</td>
<td>4.85 ± 2.66</td>
<td>7.06 ± 6.74</td>
<td>0.46 ± 0.26</td>
<td>0.74 ± 1.12</td>
</tr>
<tr>
<td>0.9</td>
<td>5.90 ± 2.64</td>
<td>7.02 ± 6.70</td>
<td>0.54 ± 0.51</td>
<td>0.48 ± 0.79</td>
</tr>
</tbody>
</table>
This can be explained by more closely examining the data. Simply put, some subsets are harder to align than others. Consider the examples in figures 3.8 and 3.9. Both of these results come from the parameter sweep using ICP with SNADHs, varying the threshold for point matches from 0.2 to 0.9 in steps of 0.1. For the dataset shown in figure 3.8, the distribution of errors across all parameter steps is $3.64m \pm 1.72m$, whereas for the dataset shown in figure 3.9, the distribution of errors across all parameter steps is $0.24m \pm 0.04m$.

To account for this variance, we attempt to normalize based on the overall score distribution for a given subset, across parameter values for a given point feature. For each dataset, we have a distribution of $N$ values (for $N=8$ parameter steps). We convert these values into z-scores:

$$z_i = \frac{x_i - \bar{X}}{\sigma_X}$$

Using the means and standard deviations of the distributions on a per-dataset basis. Since the z-scores more accurately capture the relative performance of different parameter values for that dataset, we look at the means and standard deviations of the z-score distributions instead of the raw data itself. We refer to this process as dataset normalization, since it normalizes for the idiosyncratic effects of alignment on a per-dataset basis. We do this for each point feature separately. These results are presented in tables 3.2 and 3.3 with table 3.2 containing the 10% trimmed ground truth RMSE (equation 3.18) and table 3.3 containing the rotation matrix geodesic (equation 3.22).

Since these values are the means and standard deviations of z-scores, they no
Figure 3.8. Top: the initial misalignment between these two subsets. Middle: ICP + SNADHs with match threshold = 0.8, resulting in the best alignment (err = 0.17m). Bottom: ICP + SNADHs with match threshold = 0.3, resulting in the worst alignment (err = 0.28m). This dataset aligns well for all parameter values, resulting in a very small difference between the best and worst results.
Figure 3.9. Top: the initial misalignment between these two subsets. Middle: ICP + SNADHs with match threshold = 0.3, resulting in the best alignment (err = 0.28m). Bottom: ICP + SNADHs with match threshold = 0.9, resulting in the worst alignment (err = 7.69m). In this case, a high match threshold filtered out too many matches, causing ICP to overoptimize on the remaining few.
TABLE 3.2

PARAMETER SWEEP RESULTS: DATASET-NORMALIZED MEANS AND STANDARD DEVIATIONS OF 10% TRIMMED RMSE OF GROUND TRUTH POINT-TO-POINT DISTANCES

<table>
<thead>
<tr>
<th>$\tau_m$</th>
<th>3DSC</th>
<th>FPFH</th>
<th>Spin Images</th>
<th>SNADH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.22 ± 0.11</td>
<td>0.06 ± 0.12</td>
<td>0.63 ± 0.12</td>
<td>0.35 ± 0.29</td>
</tr>
<tr>
<td>0.3</td>
<td>0.18 ± 0.20</td>
<td>-0.08 ± 0.12</td>
<td>0.62 ± 0.09</td>
<td>0.47 ± 0.19</td>
</tr>
<tr>
<td>0.4</td>
<td>0.19 ± 0.17</td>
<td>0.12 ± 0.08</td>
<td>0.50 ± 0.09</td>
<td>0.30 ± 0.09</td>
</tr>
<tr>
<td>0.5</td>
<td>-0.00 ± 0.10</td>
<td>0.05 ± 0.07</td>
<td>0.46 ± 0.06</td>
<td>0.17 ± 0.08</td>
</tr>
<tr>
<td>0.6</td>
<td>0.18 ± 0.11</td>
<td>0.31 ± 0.20</td>
<td>0.24 ± 0.06</td>
<td>0.06 ± 0.10</td>
</tr>
<tr>
<td>0.7</td>
<td>0.12 ± 0.09</td>
<td>0.34 ± 0.19</td>
<td>-0.04 ± 0.19</td>
<td>0.12 ± 0.14</td>
</tr>
<tr>
<td>0.8</td>
<td>-0.15 ± 0.09</td>
<td>0.31 ± 0.13</td>
<td>-0.50 ± 0.52</td>
<td>-0.40 ± 0.33</td>
</tr>
<tr>
<td>0.9</td>
<td>0.06 ± 0.12</td>
<td>0.27 ± 0.14</td>
<td>-1.09 ± 0.65</td>
<td>-0.71 ± 0.44</td>
</tr>
</tbody>
</table>
TABLE 3.3
PARAMETER SWEEP RESULTS: DATASET-NORMALIZED
MEANS AND STANDARD DEVIATIONS OF GROUND TRUTH
ROTATION MATRIX GEODESIC

<table>
<thead>
<tr>
<th>$\tau_m$</th>
<th>3DSC</th>
<th>FPFH</th>
<th>Spin Images</th>
<th>SNADH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>-0.18 ± 0.07</td>
<td>-0.18 ± 0.18</td>
<td>0.00 ± 0.20</td>
<td>0.15 ± 0.29</td>
</tr>
<tr>
<td>0.3</td>
<td>0.09 ± 0.24</td>
<td>0.14 ± 0.24</td>
<td>0.13 ± 0.17</td>
<td>0.12 ± 0.16</td>
</tr>
<tr>
<td>0.4</td>
<td>0.00 ± 0.15</td>
<td>-0.08 ± 0.06</td>
<td>0.17 ± 0.15</td>
<td>0.15 ± 0.10</td>
</tr>
<tr>
<td>0.5</td>
<td>-0.27 ± 0.11</td>
<td>-0.22 ± 0.16</td>
<td>0.15 ± 0.10</td>
<td>0.03 ± 0.08</td>
</tr>
<tr>
<td>0.6</td>
<td>-0.08 ± 0.04</td>
<td>0.04 ± 0.11</td>
<td>0.08 ± 0.13</td>
<td>-0.07 ± 0.20</td>
</tr>
<tr>
<td>0.7</td>
<td>-0.11 ± 0.17</td>
<td>-0.03 ± 0.24</td>
<td>-0.18 ± 0.20</td>
<td>0.04 ± 0.20</td>
</tr>
<tr>
<td>0.8</td>
<td>-0.06 ± 0.11</td>
<td>-0.16 ± 0.11</td>
<td>-0.25 ± 0.18</td>
<td>0.29 ± 0.55</td>
</tr>
<tr>
<td>0.9</td>
<td>-0.03 ± 0.08</td>
<td>-0.08 ± 0.14</td>
<td><strong>-0.45 ± 0.34</strong></td>
<td><strong>-0.58 ± 0.72</strong></td>
</tr>
</tbody>
</table>
longer correspond to absolute error values (and numbers less than zero are to be expected), but lower is still ‘better.’ Specifically, a value of less than zero means that that point feature, at that parameter threshold, performed on average better than the mean for each involved dataset when compared to other parameter thresholds for the same point feature. Since this experiment is concerned with establishing an ideal parameter for the point features separately, relative performance across point thresholds is what is ultimately important.

For the results in table 3.2, computed from the 10% trimmed ground truth point-to-point RMSE, there is still some overlap between experiments but choosing an ideal parameter is much more straightforward. The “best” results from each column are highlighted in boldface, establishing the parameters which we use for the rest of the experiments - see table 3.4. Table 3.3 presents the corresponding results, using the rotation matrix geodesic instead of the ground truth RMSE. While the RMSE distance captures a direct measurement of how closely points’ final positions match their ideal final positions, the geodesic measures a distance between the rotation matrices as automatically derived and provided by the ground truth transformation. This metric appears correlated with the RMSE results, but differs in some notable ways. Ultimately, it is not as hard of a metric as the point-to-point distance, because it is not affected by point density or distribution.

It is important to note that we cannot expect a parameter sweep to cleanly correspond to improved results; the parameter involved here affects which points get labeled as matches, but it is possible that bad matches can still help the overall alignment and good matches can fail to meaningfully improve it. For example, if enough bad matches occur in foliage, as is common with most of the algorithms,
TABLE 3.4

BEST ESTABLISHED PARAMETERS $\tau_m$ FOR POINT MATCHING

<table>
<thead>
<tr>
<th></th>
<th>3DSC</th>
<th>FPFH</th>
<th>Spin Images</th>
<th>SNADH</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_m$</td>
<td>0.8</td>
<td>0.3</td>
<td>0.9</td>
<td>0.9</td>
</tr>
</tbody>
</table>

these matches can have a stabilising effect on the resulting transformation, preventing ICP from sliding away from an ideal result.

We also note that we cannot directly compare cross-feature performances from this data, since the z-scores measure relative performance across parameters for a given feature only. The full experimental results are discussed in section 3.7.

All of the metrics used for the parameter sweeps are derived from comparisons with a manually created ground truth. In the absence of ground truth, we must use the point-to-point nearest neighbor metric described in 3.4 which, in addition to being susceptible to point density issues, will likely return inflated values whenever points from one scan are not present in the other (i.e. will be higher in scans which exhibit change). For example, even when using the ground truth alignments, the NN RMSE metric will return values greater than zero. Figure 3.10 shows a histogram of the NN RMSEs obtained using the ground truth transformations: it is likely that automatic experiments will not perform better than these values on their corresponding datasets.
Figure 3.10. A histogram of the baseline values from the nearest-neighbor RMSE metric, executed using the ground truth alignments. Note that even the ground truth data does not result in perfect 0 scores.
3.6 ICP versus RANSAC

One observation based on the ICP experiments is that point pair assignments are likely to remained fixed over time. After applying the initial ground plane prealignment, point clouds are generally misaligned less than 10 meters, meaning that with a sufficiently large neighborhood radius, we are likely to find a point’s best match in the other point cloud either immediately or within a few iterations. To determine how frequently this occurred, we saved the list of point matches determined after the ground plane prealignment, $M_0$, and the list of point matches determined after the final transformation, $M_n$. We compute the percentage of point pair assignments which remain constant over all iterations of ICP:

$$\frac{|M_0 \cap M_n|}{|M_0 \cup M_n|}$$

(3.24)

For the experiments performed using the optimal parameters established above, the histogram of this metric can be seen in figure [3.11]. Given that the mean value is over 50%, these results make a strong case for RANSAC as an alignment framework: if we can establish most of our final point matches from our initial position, we do not need to perform multiple iterations. We also examined the number of iterations performed by ICP before reaching an optimal solution. We find that on average, they converge within 7 iterations ($\bar{X} = 7.04, \sigma = 5.73$).

For RANSAC, we also examine the number of iterations until convergence, and find $\bar{X} = 354.53, \sigma = 324.28$. Although the number of iterations for RANSAC convergence is much greater than the number of iterations for ICP convergence, each RANSAC iteration requires the computation of the NN RMSE, whereas each iteration of ICP requires a full search for point matches between the two point clouds.
Figure 3.11. A histogram of the point pair consistency metric given by equation 3.24: \( \bar{X} = 0.51, \sigma = 0.30 \).
TABLE 3.5

MEANS AND STANDARD DEVIATIONS COMPARING ICP AND RANSAC FRAMEWORKS UNDER GROUND TRUTH 10 % TRIMMED RMSE

<table>
<thead>
<tr>
<th></th>
<th>3DSC</th>
<th>FPFH</th>
<th>Spin Images</th>
<th>SNADH</th>
</tr>
</thead>
<tbody>
<tr>
<td>RANSAC</td>
<td>0.30 ± 0.07</td>
<td>0.33 ± 0.09</td>
<td>0.23 ± 0.07</td>
<td>0.18 ± 0.06</td>
</tr>
<tr>
<td>ICP</td>
<td>4.85 ± 2.66</td>
<td>5.67 ± 4.30</td>
<td>0.54 ± 0.51</td>
<td>0.48 ± 0.79</td>
</tr>
</tbody>
</table>

To compare the performance of both algorithms, we utilize the ground truth RMSE metrics for the optimal parameters as determined during the parameter sweep. The results, seen in table 3.5, show that RANSAC meaningfully outperforms ICP for all of the point features, and produces a vast improvement for both 3D Shape Contexts and Fast Point Feature Histograms. Comparing the algorithms in action gives clues as to why this is: when ICP fails to reach a satisfactory alignment, it is frequently due to the overinfluence of bad point matches. If enough bad matches are made, they can pull the point cloud beyond the desired transformation, and cause huge RMSE errors due to bad alignments. For RANSAC, an individual transformation is generated with just a few point matches; as long as there are three or four helpful pairs of points in the list of generated matches, then given enough iterations RANSAC will produce the most optimal solution according to the given metric. For ICP, if the algorithm is led astray, it is likely that it will not complete with satisfactory results.
3.7 Discussion

The results of alignment on the full dataset of 1,493 overlap subsets can be seen as a set of boxplots in figure 3.12. Note that since we do not have ground truth alignments for the majority of the data, the results given are computed using the nearest neighbor 10\% trimmed RMSE metric. Notably, there is a large overlap among the various experiments. This is partly for the same reason as mentioned in the previous section - that some datasets are more difficult to align than others - but is also because the NN-RMSE metric is much more affected by point density. Unfortunately, there’s not much that can be done to ameliorate this effect - since we only have four alignment results for each dataset, we don’t have sufficient data to normalize data points into z-scores as we have previously done. However, all of the point features have strong performance, with an average alignments of 0.30m, 0.33m, 0.23m, and 0.18m, respectively, with SNADHs obtaining the lowest overall mean.

3.7.1 Sources of Misalignment

The causes of misalignment between geo-registered point clouds in our datasets are not immediately clear. We suspect that the misalignment is caused by some form of drift, present in either the GPS or IMU trace, or both, as opposed to errors with the acquisition process, but isolating the source of this drift remains an open problem. Importantly, we note that misalignment arises even when a scan is overlapped with itself, suggesting that drift accumulating over the course of a single scan may be the primary cause.

Accordingly, we have executed experiments to try to quantify this drift with respect to the amount of time elapsed in a scan. Specifically, we sought to quan-
Figure 3.12. Box plots showing the 10% trimmed nearest neighbor RMSE results for the full-dataset RANSAC experiments.
Figure 3.13. A graph of the misalignment obtained from ground truth alignment of 31 overlap subsets obtained by intersecting the ND_2008_00 scan data with itself.

Unfortunately, one of the first observations is that the availability of overlaps is limited: the majority of overlaps occur within 15 minutes of each other, with...
a small number occurring between 50 and 60 minutes of each other. This can
be explained intuitively; as the ND_2008_00 scan forms a closed path, ending in
the roughly the same position as it started, the majority of self-overlaps occur
near both the start and the end of the scan. For efficiency purposes, scan drivers
try to avoid scanning the same area multiple times, and so in this scan, as with
any other scan, the only points of overlap tend to be in bottlenecks or on major
thoroughfares. For a scan of a closed area, such as a college campus, this is the
most common; for scans of urban areas and city streets, this is far less so.

Unfortunately, while the data is suggestive of a trend, it does not provide a
clear picture of how drift is related to the time difference between two scans. We
can see that the drift is larger with time elapsed, which matches our intuition, but
there is too little data to draw more specific conclusions.
CHAPTER 4

CHANGE DETECTION

In this chapter, we describe the research that we have done in the area of change detection. We begin with a formal description of the problem as well as a discussion of the main challenges involved. We then detail the experiments we have performed so far in the context of change detection as a whole, and conclude with a discussion and future directions for work.

4.1 Motivation / Problem Statement

LIDAR-based acquisition systems are very popular for environmental modeling, and the prevalence and size of such datasets is increasingly large. However, LIDAR data by itself is generally not sufficient for an environment model; the point-based representation requires sophisticated rendering techniques to avoid holes, and can contain much more data than is necessary. Consequently, solid representations of scans are typically created from such raw point data, either automatically or through artist intervention [86]. Unfortunately this methodology is infeasible at larger scales of operation, for instance when creating a comprehensive city model. By the time such a project is completed, the real-world reference will likely have changed in meaningful ways - either through addition, modification, or removal of buildings or major infrastructure. For any approach to be feasible and effective, it must be supported by a maintainable framework.
The work presented in this dissertation enables such research by providing a framework for automatically detecting changes between repeated LIDAR scans of an area. Our work will focus on a distinction between detecting changes in the data and changes in the underlying model that the data represents, as the point-cloud representation is ultimately just a sampling of a true surface model. By realizing this distinction, we hope to eliminate a reliance on scan structure or density, both of which are highly variable in mobile acquisitions.

Additionally, a primary difficulty that arises when attempting to create a comprehensive environmental model, particularly in urban areas, is the presence of unwanted obstruction as caused by transient objects - frequently pedestrians and vehicles. We note that this problem can be mitigated via a similar application of change detection, on a different scale. Change detection for the purposes of database maintenance involves the comparison of scans taken with large timescale differences, and differences over a certain size can be classified as meaningful, structural changes. However, if we compare scans taken with a short time difference - a few days, or even a few hours - we can instead look for smaller changes, which can be interpreted as the presence or absence of transient objects. By filtering these transient changes, we can increase the fidelity of a scan dataset, as well as the relevance of the recorded data. Ideally, this can all be performed without human intervention.

We formally define the change detection as follows. Given two point clouds, $P$ and $Q$, which are samplings of underlying surface models $\psi_P$ and $\psi_Q$, respectively, we seek to identify a number of change regions $R$ which correspond to differences between $\psi_P$ and $\psi_Q$. A region $R_i$ may be expressed as a series of geodetic points obtained from one scan which represent a structure absent in the other scan ($R_i =$
\{p_0, ..., p_k\}). Alternately, a region \(R_i\) may be expressed as a bounding box in geodetic coordinates \((R_i = \{LLA_{min}, LLA_{max}\})\), denoting a region in space which where \(\psi_P\) and \(\psi_Q\) differ.

4.2 Related Work

Change detection is a very well researched area, but with many varying applications, the phrase can take on different meaning in different contexts. Generally speaking, “change detection” refers to the identification of meaningful differences in the state of a measurable property or set of properties over multiple observations [71]. Change detection is frequently employed in the context of Geographical Information Systems (GIS) for the purposes of monitoring or characterizing urban growth [11, 15, 16, 49, 54, 59, 73, 78, 80, 88], for monitoring ecological conditions (including canopy cover [62] and landslide monitoring [35]), for assessing damages after natural disasters [47, 75], to assist in monitoring construction projects [28, 42, 67, 68, 87], or for infrastructure maintenance [17, 46, 58].

In the context of this dissertation, we are interested in evaluating change detection research based on LIDAR data. The literature can be divided into two rough categories: aerial applications and ground-based applications. The former category contains the vast majority of research, and while it can be difficult to draw direct comparisons to our approach, which utilizes ground-based sensors, we can gain a useful understanding of the state of the field through examination of these approaches, as well as a vocabulary with which to frame our analysis. Change detection from ground-based LIDAR is a relatively unexplored area, with only a few works directly addressing the challenges involved [28, 35, 42, 58, 67, 68, 87], making it a particularly interesting topic for development. We begin by examining
these separately.

4.2.1 Aerial Approaches

The majority of change detection work in the context of GIS is based on data obtained aerially. While many approaches work exclusively with multispectral imagery [5, 49, 88], a large amount of research benefits from the inclusion of Digital Surface Models (DSMs) obtained via LIDAR technology [11, 15, 16, 47, 48, 54, 59, 62, 67, 73, 75, 78–81], either exclusively or in conjunction with multispectral imagery.

While the motivations for individual research papers range from urban planning [88] to disaster response [75], they are all broadly concerned with identifying meaningful changes between registered datasets observed at different time intervals. For urban applications, many research projects first tackle segmentation, and then attempt to monitor how segments change over time. In [54], the authors primarily utilize LIDAR to improve building segmentation over traditional image processing techniques, clustering data points based on height and applying a region-growing approach to isolate building roofs. “Change” can then be computed as the amount of overlap between segments obtained during one time period and segments obtained during another. This method of segmentation and segment change detection is very popular [15, 16, 75, 78, 80] and allows for a meaningful, object-based approach to the analysis [81].

However, some simpler methods choose to construct change images in a structure-agnostic manner, deferring interpretation of the output to a human operator. [59] functions in this way, presenting a user with a difference image of filtered DSMs. [62] notes that for applications where segmentation may be less meaningful, such
as forest canopy analysis, utilization of LIDAR data can better distinguish meaningful change regions from noise.

Most approaches that utilize LIDAR data first convert the raw scan data into DSMs, allowing the application of more traditional image processing operators and filters. At least one paper surveyed, [11], does not use DSMs, opting instead to use the raw LIDAR point clouds themselves. The authors construct a triangulated irregular network (TIN) over the data for a reference scan, and then measure per-point distances to this structure to determine whether a query measurement differs substantially. Although quantification of the overall change is still left to visual observation, this point-to-surface distance metric helps to make change detection more robust to noise.

4.2.2 Ground-Based Approaches

Ground-based LIDAR frequently finds applications in surveying [35], construction monitoring [28, 42, 58, 67, 68, 87], or infrastructure assessment [17, 46]. Although some systems incorporate data from mobile acquisition platforms [67], the majority utilize stationary LIDAR scanners as part of terrestrial laser systems (TLS). The benefit of using a stationary system is the ability to work with scan data as range images, similar to the DSMs frequently utilized in aerial systems. Most papers follow this approach, as it allows for the application of traditional image processing techniques for filtering and fast analysis.

Given the physical constraints placed on data acquisition, applications using LIDAR in a TLS frequently gravitate towards building or construction monitoring, although specific geographical regions can be surveyed accurately with effort [35]. Sequeira et al. [67] combine ground-based and aerial LIDAR systems in a building
verification scenario specifically tasked with ensuring compliance with Nuclear Proliferation Treaty obligations. In such an application, accurate measurement of individual building components necessitates the detail of a ground-based system. Girardeau-Montaut et al. [28] monitor an entire construction site from day-to-day to track building and excavation progress. In this application, precision is not necessarily a key requirement, since the results are interpreted visually by an operator. Some applications, as in [17, 46, 58], are concerned less with locating the changes and more with using LIDAR data as a means of quantifying change or deformation, for surveying purposes.

In [87], the authors utilize a stationary scanner with a 360° horizontal field of view for a building model change detection application. The scan points obtained are transformed into a panoramic range image before analysis. To detect changes between scans obtained at different known positions, the scan data for one scanner is reverse-projected into the panoramic range image of the other and the depths are directly compared. This allows the authors to classify changes based on whether a point is closer than the previous depth buffer value (indicating a change), whether it is approximately equal to the previous depth buffer value (indicating no change), or whether it is further than the previous depth buffer value (indicating either the absence of the previous obstructing object or additional data that was previously unseen). [42] expands upon this idea, following the same panoramic range image reverse projection, but introducing optimizations to avoid false positives in foliage, and using a region growing technique to capture coherent groups of changes.

The most direct comparison to our research can be found in [28]. In this paper, the authors compare multiple LIDAR scans of a construction site from a fixed viewport to identify day-to-day changes. Although the data from the stationary
TLS scanner could be interpreted as a range image, the authors process the data in 3D cartesian coordinates, making their approach much more generally applicable, although they do use the location of the scanner to derive visibility information for classification purposes. An octree \cite{14} is fit to the data, and corresponding cells across both observations are compared with varying metrics to determine if their contents have considerably changed. Cells that exhibit change are linked via a connected components algorithm, and the resulting set of changes is presented to a user for visual analysis.

One of the metrics used to calculate distance between corresponding octree cells in this paper, and used analogously in \cite{42}, is the \textit{Hausdorff distance} \cite{36}. The Hausdorff distance is a distance between two point sets, \( A = \{a_0, ..., a_n\} \) and \( B = \{b_0, ..., b_m\} \), defined as:

\[
h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\| \tag{4.1}
\]

For a given point \( a \in A, b \in B \) is the nearest point from the other point set; the pair of these that maximize their separation is used when computing the Hausdorff distance. This can be thought of intuitively as the “worst choice of \( a \)” in terms of the distance between \( a \) and the point set \( B \). However, this distance is not symmetric: it is unlikely that \( h(A, B) = h(B, A) \). Consequently, the \textit{symmetric Hausdorff distance} is most frequently used when referring to the concept:

\[
H(A, B) = \max(h(A, B), h(B, A)) \tag{4.2}
\]
4.2.3 Abstract Comparison Methods

There are also, in literature, research projects which aim to characterize differences between point clouds and models in an application-agnostic fashion. Mémoli and Sapiro [55] derive a computationally viable version of the Gromov-Hausdorff distance, which itself is a variant of the Hausdorff distance for comparing manifolds embedded in higher-dimensional spaces. One of the primary limitations of this technique is that it requires surfaces to be densely sampled, and to have similar sampling density. For applications involving mobile acquisition platforms in particular, this is a requirement that cannot be enforced; it also presents a challenge for stationary scanners, although resampling range images is more well-defined. Lastly, this distance is computed between two entire point clouds, giving only a quantification of the degree of change, and no information about the location of such change. This limitation is generally true of other techniques, however; [28] circumvent this by computing the Hausdorff distance on a cell-by-cell basis of their octree, but they note that the metric is still very computationally intensive.

Aspert and Ebrahimi [3] and Cignoni et al. [19] both provide variants of the Hausdorff distance appropriate for comparing distances between a point cloud and a surface model, or between surface models. The former is frequently used in verification scenarios, when one wishes to compare scan data with a preexisting CAD model, but is also used to compare points from one cloud with local surface approximations from another [28]. Both of these techniques are generally suitable for a wide variety of applications, as point-to-point and point-to-surface metrics are both frequently employed.
4.2.4 Evaluation Metrics

The wide range of applications for change detection research occasion an equally wide variety of evaluation metrics. A large number of research projects do not attempt to quantify the success or failure of their approach, opting instead to present a visualization of the results to a human operator for further analysis [11, 28, 35, 42, 59, 67, 68, 87]. This is particularly helpful in situations where the quality of the change may be difficult to meaningfully quantify, such as the landslide monitoring application in Hsiao et al. [35]. In [11], the authors create a tool for visually identifying meaningful changes in aerial LIDAR data: this allows an operator to draw his or her attention to the most relevant or meaningful aspects of the dataset for a wide variety of applications. Further, because of the size of large land surveys in modern datasets, this serves to make analysis of such data feasible. This ability to meaningfully examine a previously intractable dataset is in many ways similar to our motivations for change detection research.

Matikainen et al. [54] quantify the amount of change between cross-time DSMs by comparing the presence or absence of segmented building outlines between the two processed datasets. Specifically, they establish a set of heuristics that classify segments as ‘new building,’ ‘enlarged building,’ or ‘old building,’ based on both per-pixel overlap between segmentations and the height of the segmentations. Additionally, they quantify the reliability of their segmentation algorithm by comparing coverage with a per-pixel ground truth, obtaining accuracy rates of up to 96.7%. [15, 16, 75, 79, 81] all follow similar approaches, segmenting DSMs into smaller regions, and then computing the amount of change as a correlation between region areas. In [42], the authors only used a singular scan dataset for their evaluation; to emulate change for the purposes of change detection, parts of the
scan were isolated, artificially affected with noise, and then used as a comparison dataset.

If a reference model for the scanned region is available, typically in the form of a computer-aided design (CAD) model, it can be used to directly provide an evaluation metric \[42, 47, 67, 73\]. Some approaches use a point-to-surface metric as described in \[11\], although this does not necessarily mean that a hard metric is used to determine the effectiveness of the approach. Sequeira et al. \[67\] use CAD models to obtain error on a per-sample basis, and then present those visualizations to a trained operator to ensure that the model meets specifications. Liu et al. \[46\] and Chen \[17\] compare scan data to existing infrastructure models to determine whether defects are present, and are therefore concerned with quantifying the amount of deviance from the expected model.

If it is feasible to generate such models from the data automatically, change detection could be used to compare new scan points against previously established segmentations, analogously to the segmented aerial approaches mentioned in section \[4.2.1\].

4.3 Methodology

After two point clouds in an overlap subset have been brought into alignment, we seek to identify regions of meaningful change between the two. We compare two different paradigms for performing change detection on these subsets. The first of these methods analyzes small corresponding patches of the point clouds, in an attempt to isolate broad regions of space in which the represented surfaces likely differ. The second method, by contrast, examines the individual points themselves, and seeks to ascertain whether those points represent samplings of
Figure 4.1. Two levels of an example octree structure fit to a set of points. Level 0 is the root of the octree, containing a single node which bounds all points. Level 1 is subdivided halfway along each axis. Unoccupied cells are ignored: in this diagram, they are lighter in color.

surfaces which have changed between the two scans.

4.3.1 Octree Comparison

To obtain a coarse set of differences between the point clouds, we base our comparison on an octree structure that is fit to the bounding box of both subsets. An octree is a data structure for hierarchical organization of data, represented as a hierarchy of nested bounding boxes which contain the full dataset. An example in three dimensions can be seen in figure 4.1. The root of the structure encompasses the entire point cloud: each subsequent level of the tree is split evenly along each dimension, resulting in $2^3 = 8$ children nodes, which are similarly subdivided. We refer to the root node, which encompasses the entire dataset, as level 0, such that
the number of nodes at any level $k$ is $8^k$.\footnote{For efficiency purposes, octree nodes are generally not subdivided if they contain no children. This helps to dramatically reduce the memory requirements associated with implementation.}

To perform a comparison between two point clouds, we first fit an octree structure to both point clouds, up to a maximum specified subdivision level. To ensure that the bounds of the children at each level are identical between both point clouds involved in a comparison, the bounding box for the root is set to the bounding box of the union of both point clouds. Then, we perform a direct cell-to-cell comparison and identify all cells which appear to differ substantially. Once we have identified a list of change cells we group them with a connected components algorithm to identify coherent change regions.

We define a cell difference function, $F(A, B)$, which takes two point sets consisting of points contained within corresponding octree cells, and returns a boolean value indicating whether there is substantial change between the two cells. We experiment with a number of cell difference functions, designed to account for different potential challenges.

The first cell difference function we use is a thresholded Hausdorff distance function, as defined in section 4.2.2. Intuitively, this distance will be high either the two point sets are dissimilar or if their contents represent samplings of different surfaces. However, the drawback of the Hausdorff distance is that it is affected by varying sampling density. In our data, this can prove problematic, particularly as different acquisition platforms can result in differing density variations with distance to the vehicle.

The second cell difference function we use is based on the point descriptor we introduce in section 3.3.6.1. For each cell, we compute the SNADH for the points within that cell. The cell difference function $F$ is a simple thresholded comparison
of the two resulting SNADHs. We expect the normalized nature of the histograms to protect against variances in density, and for its discriminative nature to help avoid false negatives (type II errors).

After the cell difference function has been executed on all cells, we perform a connected components algorithm to group change cells into more coherent change regions.

4.3.2 Point Comparison

One of the difficulties of performing the hierarchical comparison mentioned in the previous section is that small misalignments between the point clouds, which are omnipresent in real-world data, affect cell membership for points and structures, and therefore can undesirably affect comparison, since the structures contained within corresponding cells may no longer be shared. To avoid this difficulty, we also implement comparators based solely on the points themselves.

Our point based comparison system is simpler in nature: for each point in the probe, we record the distance to the nearest point in the model. We examine the distribution of distances, remove outliers, and then normalize all distances for the remaining points in terms of their z-score. If this z-score is greater than a threshold, we flag that individual point as being a change point, since there is no obvious match in the other point cloud. One of the benefits of the normalized distance over the octree method is that if two point clouds are slightly misaligned, the overall distribution for point distances will shift, but the normalized point distances will remain roughly the same. In other words, this metric is more robust to misalignment. A visualization of this metric can be found in figure 4.2.

One of the benefits of this approach is that thresholding based on the distri-
Figure 4.2. A visualization of the normalized point-based distance metric described in section 4.3.2. Top-left: one half of an overlap subset. Top-right: the other half of the overlap subset. Bottom-left: a visualization of the distance metric, computed with respect to the top-right cloud. Bottom-right: the same, computed with respect to the top-left cloud.
Figure 4.3. A visualization of per-point normalized distance in foliage. Dark blue indicates a nearby point in the other point cloud; red indicates large distance.

The distribution of cross-time interpoint distances means that small misalignments between the clouds are less likely to affect the result: the distribution of point distances should be shifted, but should not be substantially changed in shape. One of the downsides of this approach is that it tends to flag points in foliage with higher frequency, as seen in figure 4.3. In theory, it is also susceptible to sampling density errors; in practice, this has not been observed to be particularly problematic.

After points above a certain threshold have been identified and flagged, we propagate these point flags to nearby points. This can be thought of as a form of hysteresis thresholding. Generally speaking, the transition between points with high distance values and points with low values will be smooth. Once we set a threshold strict enough to avoid false positives, however, points near the boundary of the changed object tend to fall outside that bound. By dilating the points that pass a stricter threshold, we are more likely to obtain all points that fully belong
to the change region.

Once we have flagged all points which do not have a likely match in the other point cloud, we perform a clustering algorithm known as DBSCAN [23] to identify coherent regions, similarly to the connected components algorithm used for combining octree nodes.

4.4 Results and Discussion

To evaluate the performance of the change detection algorithms, we manually inspected 50 overlap subsets and classified the individual regions subjectively. The observation that a large amount of foliage regions are flagged as change led us to classify regions based on a) whether or not they were foliage regions, and b) whether or not they exhibited meaningful change. Additionally, we remove all detected change regions and manually examine the remaining data to determine whether our change detection algorithm failed to identify any cross-time changes.

Firstly, we notice that the majority of detected change regions lie in foliage. For the octree and point methods, detected regions are primarily foliage 48.88% of the time and 42.49% of the time, respectively. In terms of the total number of change points, octree based methods detect 57.49% of total change points as foliage, and point-based methods detect 63.23% of total change points as foliage. One of the difficulties of dealing with foliage regions is that they are non-trivially difficult to detect: although the point distribution within such regions may seem essentially random, the same can be said for smaller, but ultimately more structured regions, such as pedestrians.

We classify the non-foliage changes into three categories to communicate both the effectiveness of the algorithms and the inherent difficulty involved in process-
TABLE 4.1

CATEGORICAL RESULTS FROM AUTOMATIC CHANGE DETECTION

<table>
<thead>
<tr>
<th>Category</th>
<th>Category 1</th>
<th>Category 2</th>
<th>Category 3</th>
<th>Missed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Octree - SNADH</td>
<td>285</td>
<td>336</td>
<td>171</td>
<td>36</td>
</tr>
<tr>
<td>Octree - Hausdorff</td>
<td>301</td>
<td>352</td>
<td>198</td>
<td>43</td>
</tr>
<tr>
<td>Point-based</td>
<td>303</td>
<td>373</td>
<td>92</td>
<td>77</td>
</tr>
</tbody>
</table>

ing the data. The first category contains regions of legitimate change - points sampled from objects or structures which are present in one scan and not the other. The second category contains classifications which do not represent true structural change, but for which the algorithm cannot be faulted - these are the inherent shortcomings of the data, and include regions blocked due to occlusion, structures which are seen in one scan but not the other, and extreme density variances. Without additional information, an algorithm cannot differentiate these changes from the first category, but a human observer would likely identify them as superfluous. Lastly, the third category consists of any detections which have occurred erroneously. Examples of these three categories can be seen in figure 4.4.

We executed change detection experiments on 50 subsets and manually classified the detected regions into each of the three categories as described. After classifying each of the resulting regions, we remove all affected points and visually inspect the overlaps for any regions which were missed by the change detection algorithm. These results are presented in table 4.1.

The results are fairly similar across metrics and paradigms, with the most sig-
Figure 4.4. Top: category 1: structural changes. Middle: category 2: changes detected in the data, but the underlying model has not changed. Bottom: category 3: octree node detected as changed erroneously.
significant departure between the octree and point approaches coming in the number of category 3 errors, the incorrectly classified regions. This is likely due to the adaptability of the point-based technique: whereas the octree algorithms flag an entire leaf node of points at a time, the point-based method flags only the individual points themselves. If, after clustering points, the point-based region is too small, it is filtered out. The octree algorithms, however, are incapable of distinguishing a further granularity than the size of a leaf node, so a single flagged node is more likely to pass through a size filter and trigger false positives.

One of the primary strengths of the point-based metric which is not necessarily borne out in the statistics is its ability to cleanly segment regions of change, particularly for small regions. Figure 4.5 compares the results of point-based and octree-based change detection for relatively small regions (trash cans in a residential area) as an example of this. The point-based technique is also well-suited to grab small regions that other techniques would miss, as seen in figure 4.6, and generally results in very clean segmentations, as in 4.7.

4.4.1 Visibility

One of the difficulties of the categorization of the detected change regions used above is the inability to meaningfully distinguish between categories 1 and 2. In order to truly determine whether a detected change represents a change in the underlying scanned structure, we have to be able to infer that the change isn’t merely a new observation. In other words, we need to incorporate information about visibility. Deriving and incorporating visibility information presents a host of challenges by itself, including accuracy concerns and classification of seen and unseen space. Ultimately, though, the availability of visibility information will
Figure 4.5. Top: segmented change regions as detected by the point-based technique. Bottom: the same regions, segmented by an octree-based technique. Note that since neighboring nodes were not flagged as change regions, the octree method does not cleanly contain the region.
Figure 4.6. Left: a small construction fence. Right: a single trash can. Both regions were detected only by the point-based technique.

Figure 4.7. Pedestrians detected as change regions using the point-based approach.
allow us to make more meaningful conclusions based on the detection of change regions.

Visibility information will also enable us to better handle occlusions in the data. Consider the data shown in figure 4.8 the presence of trees between the sidewalk and the building result in a large series of “shadows” being cast on the building, which severely degrades the quality of the dataset. Furthermore, the areas of occlusion are different for both scans, resulting in a slew of false positive detections under our present framework. With visibility information, we could readily avoid this situation. Additionally, we can include that missing data to create a more comprehensive set of scan data, compensating for scan density issues that naturally arise due to acquisition conditions.

4.4.2 Classification

Given the output of this change detection framework, classification can be performed on the resulting regions to extract more application-specific information about the changes themselves. Looking at the output regions, one can easily pick out cars and pedestrians, and other classes of objects, but there are challenges that complicate a simple classification scheme.

First, transient objects will likely only be scanned from one direction. A car may have a well-defined shape, but the acquisition vehicle is likely to only scan one half of the car, resulting in a series of profiles which much be matched to a template for accurate classification. Additionally, we frequently encounter overlapping data across multiple scans. In other words, one object may occupy a space in the first scan, and a separate object may occupy or partially occupy that space in a second scan, making the change detection problem much more complicated. Figure 4.9
Figure 4.8. Top: aligned datasets obtained with two different scanning platforms. Middle and bottom: the datasets, individually. Note the destructive effect of "shadowing" from trees that obscure the building.
Figure 4.9. Top: change detection using octrees on overlapping cross-time data. Bottom: change detection using the point based metric. While the point-based metric provides better segmentation, the objects themselves cannot be completely segmented without a more sophisticated approach.
Figure 4.10. An example of “smeared” data resulting from scanning a moving object. The car in the lower left was captured turning a corner and it stretched beyond recognition, making classification difficult.

contains an example of the most frequent situation in which this occurs, on a city street. The different change detection paradigms may segment the overlapping data to a greater or lesser degree, but without a more intelligent object-detection framework, it will be difficult to extract, for instance, a cleaned version of the dataset without any of the transient objects.

Lastly, the structure of the scan data for any transient object is complicated by the fact that they are frequently in motion. As the acquisition vehicle captures a continuous stream of data points, the object may get scanned multiple times, smearing a trace into an unrecognizable shape, intersecting with scan data for other objects, and generally greatly distorting any properties that make it distinguishable. Figure 4.10 contains a very common example in real-world data: as the scanned vehicle turns a corner, it is recorded across multiple locations, and
becomes indistinguishable from a multitude of more complex objects. Heuristics and domain knowledge may help in the retrieval and filtering of such objects, but their uses are very application-specific, which serves to hinder the effectiveness of a general-purpose classification framework.
CHAPTER 5

CONCLUSIONS

This dissertation addresses the primary difficulties involved in the creation of a change detection framework capable of handling tremendous amounts of environmental scan data, with applications both to database maintenance and data filtering. Although the work presented in this dissertation is generally applicable, it is also capable of handling the problems that arise from a real-world data acquisition process, including wildly varying scan densities, comparisons between data obtained by different acquisition platforms, and strong misalignment issues. In addition to these challenges, we have shown that such a pipeline is applicable to a dataset orders of magnitude larger than previous datasets in the literature, which demonstrates its utility.

We provide a satisfactory solution to the point cloud alignment problem for large environmental overlaps, and provide a thorough comparative analysis which includes both prominent research in the field and a novel point descriptor, which is shown to provide improved performance in some circumstances. We compare and contrast a number of novel techniques for meaningful change detection between aligned point clouds. In addition to a manual performance analysis of the aforementioned techniques, we also provide commentary on the unique challenges that arise for application-driven scenarios and provide suggestions for future improvements.
5.1 Future Work

Although we achieve satisfactory results with point cloud alignment, and our point features are shown to detect meaningful correspondences between point clouds, we may be able to achieve results of comparable accuracy more quickly by intelligently pruning our data. One of the primary challenges of point cloud alignment is that removing points increases speed but decreases match accuracy, making a tradeoff between the two non-trivial to analyze, but potentially very beneficial.

Change detection work is itself open to a wide variety of potential research directions. Application-driven approaches face many difficulties associated with real world data, including the challenges mentioned in section 4.4.2, but specific and general approaches can benefit greatly from the incorporation of visibility information. Not only will visibility improve filtering and classification possibilities, but it can also be used to identify unseen areas in a large database, and consequently direct future data acquisition efforts as efficiently as possible.

Change detection itself, particularly with the inclusion of visibility information, can be used in a wide variety of application areas, including urban planning, personal navigation, and disaster response. With industrial hardware, it is already possible to capture data at an overwhelming rate. With the future inclusion of LIDAR technology in consumer level automobiles, and the further proliferation of commodity range scanning hardware, the possibilities for future applications are endless. The work presented in this dissertation improves upon existing research topics required for such applications, provides a framework for future dataset comparative research, and proves the utility of such a framework on a vast amount of real-world data.
APPENDIX A

DATASETS

This appendix contains a visual listing of the drives used in this dissertation. Each drive is represented as a yellow line overlaid on relevant aerial imagery. In total, the drives represent 97.58km (60.63mi) of scan data.
Figure A.1. ND_2008_00 - aerial photography ©Google Maps
Figure A.2. ND_2008_01 - aerial photography ©Google Maps
Figure A.3. ND_2010_00 - aerial photography ©Google Maps
Figure A.4. ND_2010_01 - aerial photography ©Google Maps
Figure A.5. chicago_00 - aerial photography ©Google Maps
Figure A.6. chicago_01 - aerial photography ©Google Maps
Figure A.7. chicago_02 - aerial photography ©Google Maps
Figure A.8. chicago_03 - aerial photography ©Google Maps
Figure A.9. chicago_04 - aerial photography ©Google Maps
Figure A.10. chicago_05 - aerial photography ©Google Maps
Figure A.11. cupertino_00 - aerial photography ©Google Maps
Figure A.12. cupertino_01 - aerial photography ©Google Maps
Figure A.13. cupertino_02 - aerial photography ©Google Maps
Figure A.14. cupertino_03 - aerial photography ©Google Maps
Figure A.15. cupertino_04 - aerial photography ©Google Maps
Figure A.16. cupertino_05 - aerial photography ©Google Maps
Figure A.18. sacramento_01 - aerial photography ©Google Maps
Figure A.19. sacramento_02 - aerial photography ©Google Maps
Figure A.20. san_antonio_00 - aerial photography ©Google Maps
Figure A.21. san_antonio_01 - aerial photography ©Google Maps
BIBLIOGRAPHY


34. B. K. P. Horn, H. M. Hilden, and S. Negahdaripour. Closed-form solution


45. M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginz-


56. N. J. Mitra and A. Nguyen. Estimating surface normals in noisy point cloud


69. *SICK LMS-291 Technical Information Datasheet*. SICK, Inc., SICK, Inc. 6900 West 110th Street, Minneapolis, MN 55438 USA.


80. T. Vu, M. Matsuoka, and F. Yamazaki. Lidar-based change detection of build-


