SIGNED DISTANCE REGISTRATION FOR DEPTH IMAGE SEQUENCE

BY

DANIEL B. KUBACKI

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Adviser:

Associate Professor Minh N. Do

ABSTRACT

New depth camera technology has potential to make a significant impact on computer systems interaction with 3D objects; yet, it is currently limited due to its poor noise and resolution characteristics. In this thesis we propose to use depth camera's strongest characteristic, its video rate capture speeds, to overcome these limitations. Previous work to utilize sequences of depth images used 2D super-resolution techniques to combine chunks of depth images that are close in time in order to increase the resolution and noise characteristic. This technique took advantage of the consistency of the scene with respect to small changes in viewpoint. But, while increasing the resolution and decreasing unbiased noise, this algorithm increased biased noise. Thus, we are motivated to consider an algorithm that can first register all depth images to a common 3D coordinate system and then utilize a 3D superresolution technique, known as surface reconstruction, to increase the resolution and decrease both biased and unbiased noise.

This thesis considers the first part of this problem, which is the registration of a sequence of depth images to a common coordinate frame. Previous registration methods were developed for high resolution, low noise point clouds and perform poorly for noisy sequences of depth data. Thus, from our analysis of ideal signed distance functions, we propose a new method for finding the closest point to the surface given a signed distance function and its gradient. Utilizing Implicit Moving Least Squares (IMLS) and our analysis, we propose a new algorithm that computes the registration of a set of points to the surface defined by the IMLS function of another set of points. We also propose a grid based implementation that allows for bounded computations per time step. Our results demonstrate that the proposed algorithm is more robust in the presence of realistic depth noise.

To my Lord Jesus Christ
and my wife Sandy

ACKNOWLEDGMENTS

I would like to show my gratitude to my adviser Minh Do for providing me with the opportunity and encouragement to work with state-of-the-art depth camera technology. His instruction and encouragement provided the base for my research. I am also indebted to my many colleagues for their generous assistance and beneficial discussions, especially Joshua Blackburn, Raman Singh, Andre Targino, Tom Comberiate, Quang Nguyen, Hien Nguyen, and Huy Bui. I would also like to thank my parents for their many years of love and support, and for encouraging me to be my best. I would like to thank my wife Sandy for all her support and understanding. Finally, I would like to thank Jesus for all of the strength, encouragement, and will power to continue on in research and achieve something.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Motivation

A new technology is currently hitting the market that has potential to radically change the way humans record the world and interact with 3D virtual environments. This technology, known as a depth camera, has the ability to capture raw 3D snapshots at video rate speeds. It is predicted to become as common as the digital camera, which is found on virtually all modern phones and laptops. At first, one might wonder what significant advantage depth information provides. While color images provide information similar to what our eye records and sends to the brain, depth images are not intuitively interpretable by the human mind. For starters, in order to visualize a depth image, false coloration must be applied to differentiate depth values. Figure 1.1 contains a color and depth image of the same scene. In the color image, one can read the words on the board and recognize the person, while the depth image only gives the information that a person is sitting on a table. By itself depth information adds little to a person's perception of a scene. This is because humans generally estimate the depth to most objects in their scene from the color information alone through a complex combination of depth cues. In Figure 1.1, it is clear that there is a human *on top* of a table. This knowledge is derived from previous knowledge of the general 3D shape of humans and tables and the ability of one to occlude and support the other. This same ability is very difficult and not robust for computer systems. Thus, the main advantage that depth cameras bring is added information to computer vision algorithms.

These algorithms that previously had to estimate depth through computationally expense techniques now have access to it at up to 30 frames per second. But, this fast 3D sensing comes as a cost. There exists a

(a) Color (b) Depth

Figure 1.1: Examples of color and depth images taken of the same viewpoint.

trade-off between the speed and the quality of the results. Previous 3D scanners exemplify the excellent resolution and quality of 3D information captured very slowly. Depth cameras trade these characteristics for speed. This speed enables new applications, in that objects are no longer constrained to be static for 3D sensing. The fact that video rate 3D information can be obtained is very promising, but the noise and resolution pose very real obstacles to incorporating the new 3D information into computer vision techniques. Take for example the case of segmenting a foreground object from the background. This task is often difficult and computationally expensive for color information alone, but given depth information, a threshold on the depth can be used to distinguish foreground from background. The problem is that the depth information is noisy and of much lower resolution than the color image. Thus, edges in the color image may exist between information in the depth image. Due to noise, these edges may be hard to predict as either foreground or background and edges of the segmentation may change arbitrarily from frame to frame, causing a flickering of the boundary even for stationary objects. Thus, processing of the depth information is necessary to overcome the limitations of noise and resolution.

1.2 Problem Statement

From the previous section we motivate the necessity of processing depth image sequences for their incorporation into computer vision algorithms. We propose to overcome the limitations of noise and resolution by combining 3D information over time, utilizing the characteristic that is strongest for the depth cameras, namely, that they can collect 3D information at real time speeds. For this paper we make three assumptions that simplify the scope of problem: (1) The objects we are sensing are rigid, implying that 3D geometry will be preserved over time and that registration from one frame to the next will only require estimation of a rigid transformation, which in 3D has 6 degrees of freedom. (2) The relative movement between the scene and depth camera is small between consecutive frames. (3) The surfaces being sampled have significant 3D geometry and are sufficiently smooth.

Ideally we would like to consider registration with arbitrary motion such that information can be accumulated even in the presence of typical human body motion. It has been demonstrated in the computer vision tracking community that human bodies can be well approximated by a set of jointed rigid bodies. Thus we consider the rigid body case as a first step towards non-rigid registration. We also restrict the relative speed of the movement such that we can make the assumption that the previous frame's registration is a good initial guess for the current registration. As seen in Section 3.1 this is a necessary assumption for use with ICP registration techniques.

Our goal is to design a novel registration and integration algorithm that has the following characteristics:

Incremental: The algorithm should produce usable results after each frame of data, enabling its use in real time applications.

Utilizes all Data: All data previously collected should be utilized. Properly used redundancy should reduce noise and increase effective resolution.

Efficient: Data collected by a depth camera grows roughly linearly with time. Therefore, the computational cost and data storage

requirements of the algorithm should not grow with size of total collected data.

1.3 Related Work

While there is a large body of research dedicated to the study of 3D reconstruction from high quality point sets, due to the recent emergence of depth cameras and their limitations there is only one known attempt to integrate a sequence of depth images.

In [1], the authors begin by noting that real depth images are too noisy to perform accurate registration by standard ICP techniques; thus, they propose to first perform superresolution by a technique known as LidarBoost [2] on chunks of the depth image sequence. LidarBoost is a 2D superresolution technique that has been tailored to depth image data. The assumption is that in small time windows the depth camera will not change its view significantly, thus allowing 2D image registration. With registered images, standard image superresolution techniques are employed with a cost function that is tailored to Time of Flight depth data. Each superresolution depth image then provides a denser, less noisy point cloud which is then more amiable to registration. One issue that is encountered in this approach is that LidarBoost enhances the resolution and decreases unbiased noise, but it also enhances the biased noise. Frames that are close in time will have similar biased noise, which typically is caused by artifacts around the edges. Cui et al. [1] propose to overcome this enhanced biased noise by a semi-rigid global registration technique that assumes the biased noise can be modeled such that all depth pixels with the same radial distance from the center of the depth image have the same bias. This limits the number of unknowns and makes their registration tractable. In the process of making this assumption, they also assume that reflectance, edge, and distance biases can be neglected, which is not usually the case.

One of the issues with the approach of [1] is that they only utilize information from similar perspectives to reduce noise. Due to the reflective nature of most depth cameras, the angle at which a surface is oriented with respect to the camera greatly impacts the noise variance of the sample. Therefore, surfaces measured at oblique angles will be noisier than surfaces

measured at perpendicular angles. Thus, information collected around edges in one frame may present itself again later as a flat surface in subsequent frames. Based on this intuition, it seems profitable to first register all frames to a common coordinate reference, and then perform 3D superresolution, a.k.a. surface reconstruction.

1.4 Thesis Summary

The remainder of this thesis is structured as follows. In Chapter 2, we present an overview of 3D sensing technologies. Previous technologies focused on high quality 3D point clouds at the cost that objects to be scanned were required to be static such that the 3D shape and calibration would not change over the duration of the scanning process. These high quality point datasets motivated algorithms that were designed to fully reconstruct 3D objects given as few viewpoints as possible since each scan was expensive in time. This leaves room for new techniques that are motivated by the relatively inexpensive ability of depth cameras to obtain new viewpoints in real time, but are limited due to their noise and resolution. This section also provides a detailed look at Time of Flight depth camera technology.

Chapter 3 examines variations of the dominant registration method, Iterative Corresponding Point (ICP). It considers the main operation of point matching and the choice of the error metric used to determine the optimal rigid transformation each iteration. We also consider the issue of registration drift for sequences of point datasets. Previous methods require global optimizations which are not well suited toward incremental registration and integration.

Chapter 4 considers the problem of integrating depth data over time through surface reconstruction. It first considers the choices for representing a surface and then presents two influential surface reconstruction techniques that both utilize signed distance functions to implicitly represent the surface.

Chapter 5 contains our analysis of ideal signed distance functions. Through the analysis we propose a way to define the closest point on the surface based on the signed distance function and its gradient.

Chapter 6 details our proposed algorithm from input to output. The basic framework is that given a new set of points and a representation of the surface by its signed distance function, new points are registered to the surface by a novel variant of ICP and then incorporated into the surface representation. We propose our algorithm based on the signed distance function described by Implicit Moving Least Squares.

Chapter 7 provides empirical evidence to support our algorithm. Comparisons are made with respect to dominant variants of ICP on synthetic depth image sequence derived from a known 3D mesh. The synthetic sequences allow for ground truth comparisons between methods and demonstrate that our proposed algorithm is more robust than previous methods in the presence of noise and low resolution.

Finally, Chapter 8 presents concluding remarks as to the results of the proposed method.

CHAPTER 2

3D SENSING TECHNOLOGIES

Depth cameras are not the first devices to provide an output of the 3D structure of real world objects. This chapter provides an overview of most 3D sensing techniques (see Figure 2.1) and a detailed look at Time of Flight (ToF) depth camera technology in order to highlight the differences between the data upon which most previous registration and reconstruction methods are built and the data received from depth cameras. These differences motivate the need for novel algorithms particularly suited to depth cameras.

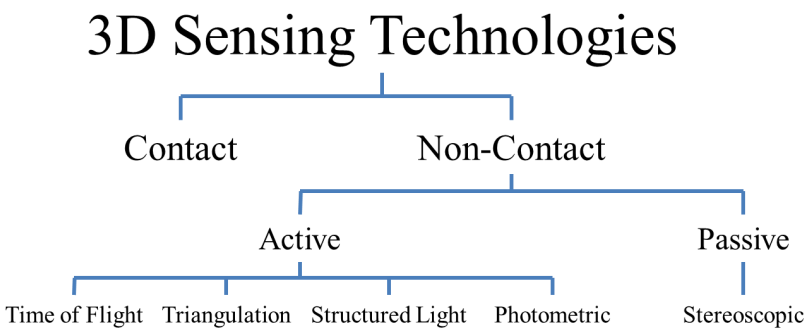


Figure 2.1: Characterization of current 3D sensing technologies.

2.1 Overview

The first order characterization of 3D sensing technologies is whether or not they require contact with the object to be sensed. Contact based techniques require a sensing instrument to touch the object, which in turn generates one point in 3D. Thus, by touching the object all over, a 3D point cloud is generated. Techniques to generate points upon contact are varied, but in general contact methods tend to produce very high quality (i.e. low noise and dense) 3D point clouds, but their applicability is limited in that

contact is a very intrusive and time-consuming process. Contact methods are not suited for large objects or pliable surfaces. Depth cameras are a form of non-contact 3D sensing; therefore, we will further characterize non-contact techniques.

The second level of characterization for non-contact techniques is based on whether the scene is actively changed in order to sense 3D structure or if it is passively observed and 3D information is deduced. Radar is an example of active sensing in which a signal is broadcast into an area and the received reflected signals are processed to determine object positions. On the other hand, non-flash photography is an example of passive sensing in that no illumination is added to a scene but images are generated from visible light naturally reflected off objects. The following sections detail the major active and passive 3D sensing technologies.

2.1.1 Time of Flight

Time of Flight techniques measure distance to an object by measuring the time it takes for a sensing signal to travel from the source to an object and back. Provided that the speed of the sensing signal is known, the distance to the object can be calculated by

$$distance = \frac{1}{2}\, rate \times time$$

The $\frac{1}{2}$ corresponds to the fact that the signal had to travel over the distance to the object twice. This is the basis for light detection and ranging (LIDAR) and radar. Differences among Time of Flight techniques focus on the method upon which the time of flight is estimated. For estimates that require only large distance scales, or for single estimates, it is possible to measure the time change directly with modern hardware. As a frame of reference, to estimate distance to 1 mm accuracy requires measuring the time of flight to picosecond accuracy. Hardware capable of measuring changes on that scale is very sensitive to noise and thus must be sufficiently cooled and shielded. This makes densely packed sensing on a small scale impractical using direct measurement.

An alternative method is to use a modulated sensing signal and indirectly measure the time of flight from the difference in phase between

the sent and received signals. This method benefits from the fact that the phase difference estimation can be made numerous times in a short interval given a high frequency sensing signal. Thus, by averaging over many estimations, noise is reduced. This averaging allows for less precise individual estimates and thus smaller and more inexpensive sensors. In fact, phase ToF sensors are small enough that they can be packed into a 2D camera pattern and operate at video rate speeds to produce a depth camera. As mentioned, ToF depth cameras utilizing phase differencing will be expounded in greater detail in Section 2.2. A major limitation of phase difference time of flight is that the distance it can measure must be limited. If an object were to be sufficiently far away that the sensing signal's phase wraps around to zero before being measured, then the estimate would alias to a much shorter distance. In practice, phase differencing ToF cameras typically operate in a range up to 7.5 meters. Two major manufacturers of ToF depth cameras are PMD Technologies and Canesta.

2.1.2 Triangulation

Given a source and a camera detector that are positioned with known calibration, triangulation uses the detected position of the source in the camera image to determine angles in the triangle formed between the source, detector and object point. Ultimately, based on the calibration of the source and detector, the distance and direction to the point on the object can be calculated. The resolution of the triangulation is determined by how accurately the detector can localize the source reflection in the image. Using lasers and high resolution imaging devices these scanners provide millimeter accuracy with high spatial resolution. Scanners utilizing this technique employed with lasers are often referred to as laser scanners. Scanners such as these were employed in the Digital Michelangelo Project [3] and for many of the available models found in the Stanford 3D Repository [4]. A manufacturer that provides laser scanners is Cyberware.

One limitation to triangulation techniques is that they require a scan over any object in order to collect 3D points. This is accelerated by scanning with a laser line over just a solitary point, but at any one point in time only a local part of any object can be scanned. Multiple scanning devices can be

employed simultaneously, but in general, scanning requires many seconds to many minutes depending on desired accuracy and resolution. In comparison to depth cameras, laser scanners have much higher resolution and better noise characteristics, but they operate at speeds that require static objects.

2.1.3 Structured Light

Structured light is similar to triangulation in that light is projected onto a surface and is detected by a camera. But, where triangulation utilizes precise localization of the source in the image, structured light infers 3D structure from the deformations of the projected light pattern. The light pattern can vary from points to 1D stripes to complex 2D patterns. A rapid succession of these patterns allows algorithms to disambiguate the sensed distorted patterns and infer depth. The advantages of this method are that it utilizes CCD cameras which can have very high resolution and that it can estimate 3D structure for a whole scene all at once. This makes structured light possible at video rate speeds. A disadvantage is that it does not directly measure the distances, but rather infers them from deformations of light patterns. Therefore, the noise of the system is heavily influenced by the algorithm used to determine deformations rather than by physical principles. This makes the noise difficult to characterize.

Past implementations of this technique utilized visible light to project the pattern onto the scene as seen in [5]. One disadvantage to using visible light is that it requires a quickly varying light pattern which would be very bothersome to an average human user. Therefore, recent innovations have produced structured-light depth cameras that utilize near infrared lighting patterns. This is most notably the technique used in Microsoft's Kinect. The Kinect is based on technology from the company Prime Sense. The Kinect is notable in that it is the first depth camera to hit the mass market. At this time, its primary uses involve variants of segmentation and 3D tracking algorithms.

2.1.4 Photometric

Photometric is a very different approach to 3D sensing. Instead of estimating depth, it estimates normals, which in turn infer the 3D structure. Thus 3D structure and orientation are inferred with respect to the camera, but the relative scale must be injected as prior knowledge. The basic principle is to take multiple images of the same scene under various known lighting conditions. If the scene and the camera are static, then for each lighting condition the normals can be estimated for each pixel. Given normal estimates under various known lighting conditions, the final normal estimates are very accurate and robust. Also, with HD cameras, very high resolution surface normal maps are possible. This technique is most notably demonstrated in [6] by Paul Debevic's light stages. With these light stages, it is possible to light a scene in a very controlled manner at very high speeds. This, coupled with high speed, high resolution video cameras, has enabled real-time capture of dynamically changing 3D scenes [7]. This technology has achieved major success and has been featured in a number of movies, including the Spiderman movies.

The major disadvantage of this technology is that it requires a very large and expensive "stage" with expert technical knowledge to run. This, combined with massive amounts of off-line processing required to interpret all captured data, leaves photometric 3D sensing only a possibility for the professionals.

2.1.5 Stereoscopic

Stereoscopic depth sensing, a.k.a. stereo vision, is an estimation of depth from small differences between two or more images captured of the same object from various viewpoints. Humans employ stereo vision through the use of their two eyes directed at the same point but separated by a few inches. This small separation causes differences, called disparities, between the images. Objects that are closer will cause a larger disparity, while objects that approach infinitely far away will have no disparity. Given rigid calibration between cameras and perfect matching of points between images, it is possible to calculate the distance to the point from its disparity. The difficulty of stereo vision is the necessary correspondence

matching, which is difficult in areas of low texture and requires large amounts of time for computation for high quality correspondences. There do exist algorithms that operate in real time, but the quality of the depth map is quite poor. The advantage of stereo vision is that it is passive and thus one of the most versatile 3D sensing technologies.

Interestingly, stereo vision is one of the weaker pieces of 3D information humans use. We perceive depth more through depth cues such as converging parallel lines, occlusions, perspective shrinking, and perspective shifts. These cues are what allow us to perceive depth in traditional 2D television shows. The difficulty in utilizing these same cues for signal processing is the vast prior knowledge required for their use. Even though stereoscopic sensing is weak in humans, it has been shown to be very effective in certain circumstances, such as in [8] for human faces.

2.2 ToF Depth Camera

The previous section describes four 3D sensing technologies capable of video rate 3D capture: ToF, Structured Light, Photometric, and Stereoscopic. In this section, we will focus on the state-of-the-art ToF technology and characterize its noise.

2.2.1 State of the Art

The state-of-the-art ToF depth camera technology is the Photonic Mixer Device (PMD). These devices are small and capable of being fabricated similar to CCD pixels. Due to their similarity to CCD pixels, PMDTechnologies, a leader in commercializing depth cameras utilizing PMD technology, call these devices "smart" pixels [9]. Each smart pixel is significantly larger than a CCD pixel preventing the resolution of depth cameras from being on par with color cameras. The current state-of-the-art resolution is 204×204 found in PMDTechnology's CamCube 2.0 depth camera, which is the depth camera used to generate real depth images in this thesis.

The PMD technology smart pixel can be seen in Figure 2.2. It is a five terminal device with two photo-gates in the middle. It works by driving

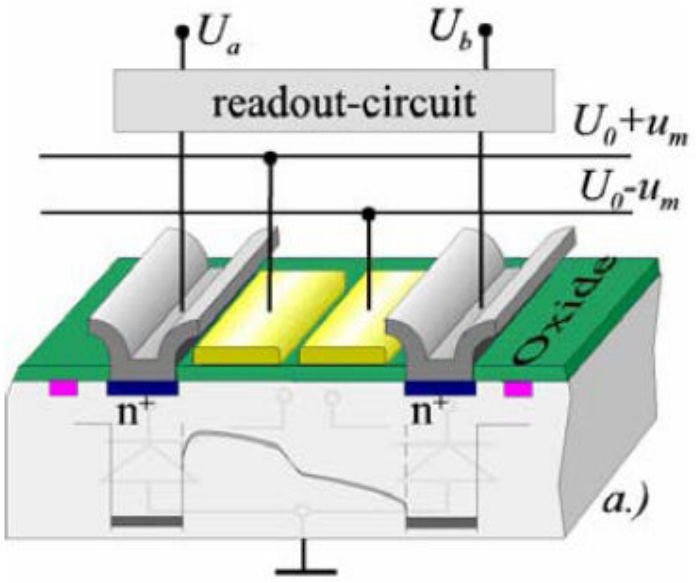


Figure 2.2: PMDTechnology's smart pixel diagram [9].

electrons to the left or right readout channel based on the differences in phase between the driving signal u_m and the received echo signal. This process can be modeled as the cross correlation between the received echo signal and the sent reference signal. If we consider the sent modulation signal, $s(t)$, as a rectangle wave and the received echo signal, $r(t)$, as a sinusoid (which is reasonable given the low-pass characteristics of IR-LEDs), then

$$s(t) = \sum_{n=-\infty}^{\infty} \text{rect}\left(\frac{2t}{T} - 2n\right)$$
$$r(t - T_L) = a_0 \cos(\omega(t - T_L)) + B$$

which makes the cross correlation function

$$\varphi(\tau) = (s \otimes r)(\tau) = \frac{k}{T} \int_{t=-T/2}^{T/2} s(t) r(t + \tau) \mathrm{d}t$$
$$= k \left[\frac{a_0}{\pi} \cos\left(w(\tau - T_L)\right) + \frac{B}{2}\right]$$

where k is a constant corresponding to the number of periods per integration time. In order to make the phase estimation invariant to the amplitude and offset of the received signal, a state-of-the-art four phase

algorithm is employed [10]. It requires that four measurements be taken corresponding to $\omega\tau = \{0°, 90°, 180°, 270°\}$. Using these four delays the phase difference between $s(t)$ and $r(t)$ can be calculated as

$$\phi = \arctan\left(\frac{\varphi_{0°} - \varphi_{180°}}{\varphi_{90°} - \varphi_{270°}}\right)$$

Along with phase, it is also possible to calculate a_0 and B_k, where B_k is the accumulated offset.

$$a_0 = \frac{\sqrt{(\varphi_{0°} - \varphi_{180°})^2 + (\varphi_{90°} - \varphi_{270°})^2}}{2/\pi}$$

$$B_k = \frac{\varphi_{0°} + \varphi_{90°} + \varphi_{180°} + \varphi_{270°}}{2}$$

Figure 2.3 gives a good visual of the relationship between the various calculated values. The amplitude corresponds to the height of the received signal above the offset, and the offset represents the accumulation of light not corresponding to the modulated signal, i.e. background illumination and object diffuse reflectivity. These additional values are provided at minimal additional computations, but grant much more information about the scene. The amplitude of a measurement is related to the strength of the received signal and also to the uncertainty of the measurement. Thus it can be used as a threshold to discard outlying points, and as a weighting function to place more emphasis on more confident terms. The offset information can be thought of as a grayscale picture of the scene which is conveniently acquired from exactly the same viewpoint as the depth information. The grayscale information is often used for denoising and/or calibration.

2.2.2 Uncertainty

There are three major sources of noise for ToF measurements: (1) photon shot noise, (2) photo charge conversion noise, and (3) quantization noise [11]. As the number of incoming photons increases, photon shot noise eventually dominates the other forms of noise. Shot noise is described by the Poisson distribution corresponding to the photon arrival process.

In [12], Lange determines that the resolution of the depth measurement is

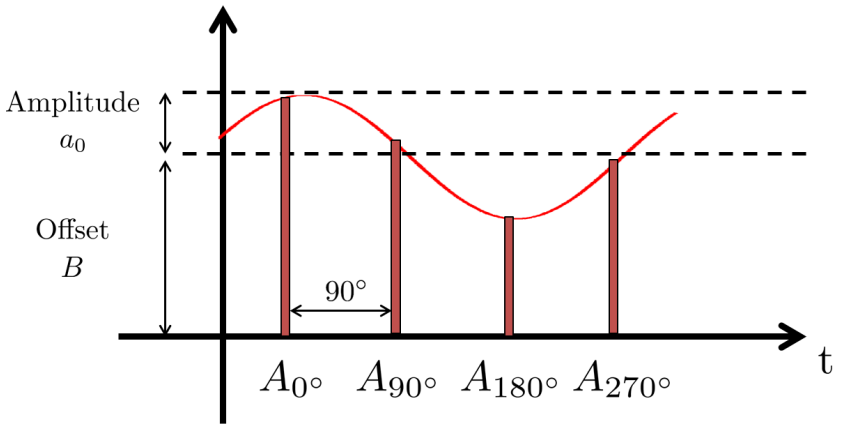


Figure 2.3: The cross correlation function between the received echo signal and the sent reference signal for a single modulation period.

described by

$$\Delta D = \frac{\Delta\varphi}{2\pi}\frac{c}{2f_{mod}} = \frac{c}{4\sqrt{8}f_{mod}}\frac{\sqrt{B}}{a_0}$$

where $c/(2f_mod)$ is the distance before the phase wraps around. Since it is possible to estimate B and a_0, each measurement contains within itself the information of its own reliability [9].

CHAPTER 3

REGISTRATION

Registration is the process of transforming different data sets into a common coordinate frame. This problem is not new to depth cameras. All of the 3D sensing techniques described in Chapter 2 are limited by their viewpoint. Therefore, in order to develop full 3D models, multiple 3D point sets are required from various viewpoints. Each of these sets must then be registered to the others. Therefore, there exist techniques that effectively solve this problem for high quality (a.k.a. low noise and high resolution) point cloud data sets. Our problem is to design a registration technique that can effectively and efficiently register depth camera data which is of significantly lower quality than previous data. In this chapter we review previous techniques in order to understand and motivate our proposed method in Chapter 6.

3.1 Variants of ICP

The dominant method for the rigid registration of two sets of points is the technique known as ICP [13]. Originally, ICP stood for Iterative Closest Point [14], but it has developed and is now a class of techniques more appropriately labeled Iterative Corresponding Point. The basic idea of ICP is to match a subset of points in one set to points from the other set and to determine the best rigid transformation in the least squared sense. This process is repeated until a maximum number of iterations or until some convergence criterion is met. Thus, the major variations of ICP consider different methods to match points and different methods to determine the optimal rigid transformation. All forms of ICP include the following six categories:

1. Point selection

2. Point matching
3. Weighting
4. Rejection
5. Choice of error metric
6. Method of minimization

Rusinkiewicz and Levoy presented a nice overview and comparison of the many variations of ICP in [13]. Considering a base implementation of ICP they systematically vary only one of the previous six categories and explore the effects of the variations on the convergence, specifically with a look at speed of convergence. In particular they find that the point matching and error metric have the greatest contribution to the convergence rate, while point selection, weighting and rejection have more to do with robustness. The method of minimization is not studied, but chosen appropriate to the choice of error metric.

ICP operates on two assumptions. The first is that good initial registrations are available. This is important for the inherent non-linear optimization involved in ICP; it will always converge, but it is possible that it will get stuck in a local minimum and not find the true registration. As seen in non-linear optimization, a good initial guess is often sufficient to avoid spurious local minima. The second is that the two point sets contain sufficient overlap. This assumption is clear from the standpoint of the point selection algorithms. If the point sets do not contain sufficient overlap, then correct point correspondences will never be generated. If either of these assumptions is invalid, ICP will still converge, but most likely to a false registration.

3.1.1 Point Matching

If corresponding pairs of points were known a priori, it would take only one iteration of ICP to determine the optimal rigid body transformation. The problem with registering two 3D point sets is that we must determine both correspondence and registration at the same time. ICP separates these goals by first determining the "optimal" correspondences and then

determining the optimal registration in the least square sense. The optimal correspondences are a heuristic. Given the assumption that there exists a good initial guess for the rigid transformation between the two point sets, then it is reasonable to assume that the point closest in Euclidean distance may be a good choice for correspondence. This assumption becomes more valid as the estimated registration becomes closer to the actual registration. Thus, the beginning iterations may contain many incorrect yet close correspondences, but assuming the registration estimate is converging to the correct registration the final iterations will contain mostly correct correspondences.

One problem with closest point correspondence is the computations required to determine it. Given N points in both sets, the computations to find the closest points require $\mathcal{O}(N^2)$ computations. This can be sped up by utilizing a k-d tree [15] to $\mathcal{O}(N \log N)$, but this is still very slow for a large number of points. Therefore, two other methods are explored in [13]: Projection [16] and Normal Shooting [14].

The projection method [16] for determining correspondences utilizes the fact that typical 3D point sets come from scanning devices that often utilize a camera. A point set that comes from a particular viewpoint can be projected onto the image plane of the camera used for capture and create what is known as a range image, essentially similar to depth images. The idea of projection correspondence matching is to project a new point set onto the range image of another point set. Then the corresponding points are those that are closest on the 2D range image. The advantage of this method is that projection is constant in computation time. Thus, corresponding points are determined magnitudes faster than by the closest point method. The disadvantage is that 3D points are being matched in 2D. This loss of dimension implies that the correspondences may be more prone to error, but they still have the property that as registration error approaches zero the matches should approach optimal. Rusinkiewicz and Levoy [13] showed that the projection method typically required more iterations than the closest point method, but the projection method is so much faster that it still converges magnitudes faster.

The normal shooting method determines correspondences by following the ray starting at a point in the direction of its normal and finding its intersection with the surface described by the other set of points. Normal

shooting requires a surface description of one of the sets of points, typically a mesh, and it requires an efficient method to compute the intersection of rays with that surface, typically a ray tracing method. The results of [13] show that normal shooting is faster than closest point ICP but not as fast as projection ICP.

3.1.2 Error Metric

Given a set of matched pairs from one of the point matching methods, let $\boldsymbol{p}_i \in S_1$ represent the i^{th} point matched to $\boldsymbol{q}_i \in S_2$. Then the simplest ICP error metric is the sum of squared distances between $\boldsymbol{p}_i$ and $\boldsymbol{q}_i$ with respect to rigid transformations of rotation, $\boldsymbol{R}$, and translation, $\boldsymbol{t}$.

$$\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}} = \underset{\boldsymbol{R},\, \boldsymbol{t}}{\operatorname{argmin}} \quad \sum_i \|\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i\|_2^2 \tag{3.1}$$

The minimization of this error metric has been solved in many ways [17, 18, 19]. But it was shown in [20] that these methods are essentially equivalent in terms of numerical accuracy and stability. One nice result of these methods is that they present closed form solutions to Equation (3.1).

An alternative error metric was proposed in [14] that minimizes the point to plane distance between matched points. Equation (3.1) can be considered a point to point error metric and it was shown that this metric provides slow convergence for planes sliding across one another. The issue is that in order for one plane to slide across another plane it incurs a penalty at all of the points matched on the flat part. This can be overcome if the error metric considers not distance from point to point, but from a point to the plane defined by a point. This method uses the surface normals associated with each point and considers only the distance between points in the direction of the normal. It is formulated as follows:

$$\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}} = \underset{\boldsymbol{R},\, \boldsymbol{t}}{\operatorname{argmin}} \quad \sum_i \left[(\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_i)^T \boldsymbol{n}_i\right]^2 \tag{3.2}$$

The disadvantage of this error metric is that it does not admit a closed form answer. One solution is to solve it by some iterative non-linear method. Another option is to linearize $\boldsymbol{R}$ using the small angle assumption,

$\cos\theta \approx 1$ and $\sin\theta \approx \theta$. This approximation is reasonable since for large angles the solution error is likely to be dominated by error due to mis-matched points, not the small angle approximation, and as the registration becomes better the small angle approximation becomes more valid. It was shown in [13] that using the linearization of $\boldsymbol{R}$ method to solve Equation (3.2) provides an increased convergence rate and more robustness over the point to point error metric of Equation (3.1).

3.2 Registration Drift

In this section, we consider the problem of registering three or more 3D point sets. One straightforward extension of ICP is to choose one point set as the basis and to register by ICP all other point sets to the basis. This solution works if a suitable basis can be found which does not violate the assumptions of ICP. In many cases, such a basis cannot be found such as in the reconstruction of an entire 3D model. In this case, for whatever viewpoint is considered the basis, the points on the opposite side of the object will have few points in common with the basis. Thus, registration between these point sets would violate the second assumption of ICP and is likely to fail.

An alternative extension is to order the point sets such that adjacent point sets have sufficient overlap. Then by registering the adjacent point sets in order, a common coordinate frame can be obtained by propagating registrations. For example, from the registration of the first and second sets, the second set can be transformed into the coordinate frame of the first set. From the registration of the second and third sets, the third set can be transformed into the coordinate frame of the second set, but then it can also be transformed into the coordinate frame of the first set. Likewise, the last point set can be propagated through all previous registrations to the first coordinate frame. It is possible to propagate the registrations and only apply the propagated registration once to bring any frame to the desired coordinate frame. This method is often used in practice.

Registration drift is the result of small errors in each pair registration that accumulate through registration propagation. For example, consider the task of obtaining a full 3D model. If the ordered point sets are

generated from viewpoints that make a complete circle around the object, then registration drift would be when the surface does not connect upon a full rotation around the object. Registration drift is a significant problem when there are large numbers of point sets.

One way to combat registration drift is to determine all of the registrations simultaneously. Neugebauer [16] demonstrates a method for simultaneous registration by first determining good initial estimates of each registration and then considering the final registration to be only determined by a local linearized correction factor. Simultaneous registration is performed through a global least squares solution for all correction factors.

Another approach is to diffuse the pairwise registration errors evenly across all registration pairs. Shih et al. [21] achieve this by first performing pairwise registrations and then using these as constraints for a multiview registration, while [22] represents the multiview registration problem as a graph and converts it to a quadratic programming problem of Lie algebra parameters.

The final approach is through the building of intermediate surfaces representations. Huang et al. [23] utilize a partition of unity surface built from piecewise quadratic functions defined on octree cells. A prototype surface is constructed for the given level of the octree. Each point set registration is refined according to prototype surface. Then the level of the octree is increased by one and the surface is reconstructed and new registrations are generated. This is repeated for the desired number of octree levels. Claes et al. [24] begin by converting each point set to a variational implicit surface (VIS) model and proceed with all levels of registration from crude to fine utilizing this model. In particular, they solve the multiview registration problem in terms of these VIS models.

3.3 Depth Camera Registration

The previous sections detail the approaches previously taken to solve the registration problem for high quality sets of 3D points. In particular we are interested in the registration of 3D point sets produced by a depth camera. This implies that our data will have considerably higher noise and lower

resolution. It will also be important to consider the registration drift problem since depth cameras produce a very large number of point sets, but it would make real time applications intractable if we required a global registration with each new set of data. Therefore an ideal registration technique suited for depth cameras is one that:

- Incorporates smoothing and denoising
- Determines an accurate registration without having to correct all past registrations

CHAPTER 4

INTEGRATION

In Section 1.2 we proposed to combat the limitations, low resolution and high noise, of a single capture from a depth camera by integrating data over multiple time instances. The desire to integrate is a perspective change from the viewer-centered representation available through each frame of the depth camera to an object-centered representation [25]. In the viewer-centered representation, the 3D data acquired at each time instant must be considered essentially distinct objects. While in the object-centered representation, data acquired at each time instant is recognized as a distinct view of the same surface. Thus, with proper alignment it may be possible to combine data from many samples in order to recover the surface. This problem of surface reconstruction from a set of measurements of the surface is a well studied problem in computer graphics. Surface reconstruction methods range from construction of a mesh [26], to implicit surfaces [27, 28], to point set surfaces [29, 30]. In some cases there may be more points than necessary to adequately define a surface; thus, [31, 32] consider the problem of consolidating a large point set to a smaller more representative point set with less noise and more uniform sampling over the surface.

In this chapter, we consider the various ways in which a surface can be represented and expound on two influential methods of surface reconstruction.

4.1 Surface Representation

There are two ways that a surface can be represented: explicitly or implicitly.

4.1.1 Explicit Representation

The explicit form of a surface is a graph of a set of functions of two variables. An adaptation of the definition from [25] is:

Definition 4.1. *Let $g_i : U_i \subseteq \mathbb{R}^2 \mapsto \mathbb{R}$ be a set of functions used to describe the surface and $T_i : \mathbb{R}^3 \mapsto \mathbb{R}^3$ be a set of 3D rigid transformations; i.e., it applies a rotation and translation to every point. Then*

$$Surface = \bigcup_i T_i([U_i, g_i(U_i)])$$

A simple way to understand this definition is that an explicit representation is a set of 3D patches made from localized 2D functions. A straightforward example is a mesh representation of a 3D object. Meshes consist of finite support, constant, 2D functions positioned between three points in 3D. Meshes are significant due to their popularity in computer graphics, 3D modeling, and 3D movie animation. More elaborate surfaces are possible by considering more complicated 2D functions such as B-splines and Bézier surfaces or Non-Uniform Rational B-Splines (NURBS).

Explicit 3D patches are usually defined by a set of control points. For example, meshes are built from the interpolation between vertices, which are control points on the surface. An explicit surface built from higher order functions such as quadratics or B-splines may be defined by control points not on the surface.

The advantage of an explicit representation is that the surface is clearly defined with known properties such as differentiability. In the case of meshes, the property that the surface is defined by the interpolation between sets of surface points allows for efficient storage and rendering. The disadvantage of explicit surfaces is that only the surface is defined. Information about points not on the surface must be computed with respect to the explicit surface. Another disadvantage particular to incremental integration is that given an explicit surface derived from one set of points it is difficult to update the surface given an additional set of points. The best solution is often to discard the previous surface and construct a new explicit surface from the combined set of points.

4.1.2 Implicit Representation

The implicit form of a surface corresponds to a function of the form

$$f : \mathbb{R}^3 \mapsto \mathbb{R} \quad | \quad f(x, y, z) = \text{constant}$$

Fundamentally, the explicit form is a special case of the implicit form since all explicit forms can be transformed into an implicit form but not vice versa. The implicit function most often considered in surface reconstruction is the signed distance function. If $f(\boldsymbol{x})$ is a signed distance function, then $|f(\boldsymbol{x})|$ is the distance from $\boldsymbol{x}$ to the closest point on the surface and $\text{sign}(f(\boldsymbol{x}))$ denotes whether the point is inside or outside the surface. For signed distance functions, the surface is defined as the set $\{x \in \mathbb{R}^3 : f(\boldsymbol{x}) = 0\}$.

The advantage of an implicit surface is that it is defined for all points in $\mathbb{R}^3$. Thus, it contains more information. The disadvantage is that the storage of all this information can be expensive. Often, given an implicit definition of a surface, an explicit mesh representation is derived from the implicit surface using the marching cubes algorithm [33].

In terms of incremental integration, since the implicit form is based on continuous functions defined at all points, it is possible to combine the implicit information at each point to derive a new implicit surface. Thus, implicit surfaces are better suited to incremental updating.

4.2 Reconstruction Algorithms

Since implicit surfaces are better suited to incremental integration, we now expound on two surface reconstruction techniques that utilize signed distance implicit functions and that form the basis of our proposed integration algorithm.

4.2.1 Volumetric Reconstruction

"A volumetric method for building complex models from range images" [26] was presented at SIGGRAPH in 1996, and it remains one of the dominant

methods to reconstruct a 3D model from a set of high quality depth images. It is based upon the following set of desirable properties:

- Representation of range uncertainty
- Utilization of all range data
- Incremental and order independent updating
- Time and space efficiency
- Robustness
- No restrictions on topology
- Ability to fill holes in the reconstruction

These desirable properties also apply to the integration of depth camera data except that we define the order to be sequential in time. The combination of the ability to utilize all data while being incremental and time and space efficient is what makes this method particularly practical.

As stated in the title, the algorithm is based on a volumetric sampling method. In this case, they define a continuous scalar function which is a weighted signed distance function. This function is sampled in a regular volume pattern. Each new frame of depth data is first converted into a mesh by simply connecting neighboring points in the depth image. The weighted signed distance is then calculated for each point of the volumetric grid using this mesh. The signed distances and weights are then combined through the following formulas:

$$D(\boldsymbol{x}) = \frac{\sum_i w_i(\boldsymbol{x}) d_i(\boldsymbol{x})}{\sum_i w_i(\boldsymbol{x})}$$

$$W(\boldsymbol{x}) = \sum_i w_i(\boldsymbol{x})$$

where i indicates the index of the frame, w_i are the volumetric weights calculated for each frame, and d_i are the volumetric signed distances calculated for each frame. The nice part about this formulation is that it is extremely easy to add new data, since assuming that $D(\boldsymbol{x})$ and $W(\boldsymbol{x})$ are

saved separately, then new distances and weights can be calculated as

$$\hat{D}(\boldsymbol{x}) = \frac{W(\boldsymbol{x})D(\boldsymbol{x}) + w_{i+1}(\boldsymbol{x})d_{i+1}(\boldsymbol{x})}{W(\boldsymbol{x}) + w_{i+1}(\boldsymbol{x})}$$

$$\hat{W}(\boldsymbol{s}) = W(\boldsymbol{x}) + w_{i+1}(\boldsymbol{x})$$

The final mesh is determined by the marching cubes on the volumetric sampled signed distance function. The main advantage of the overall algorithm is that it contains all of the desired properties mentioned above. The disadvantage is that it is designed for high quality depth data. Given a noisy low res depth frame, the resulting signed distance function would be very noisy and could result in erroneous and non-robust surfaces. The main problem is that the signed distance function for each frame is calculated in a non-rigorous way. The weighting functions are included in order to handle uncertainty in the measurements, but these are heuristically defined.

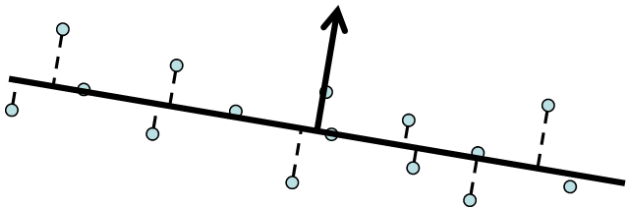
4.2.2 Implicit Moving Least Squares

Implicit Moving Least Squares (IMLS) was originally proposed as a method for interpolating and approximating polygon soup [27]. It is based on the interpolation technique Moving Least Squares (MLS), which is a method that fits planes to local neighborhoods of points in a least squares fashion. Unlike traditional MLS which is often considered a projection operator, IMLS constructs localized planes, or functions, at each input point, as seen in Figure 4.1. The sum of these functions approximates the signed distance function. It has been shown in [34] that IMLS is a provably good estimate of the surface given proper sampling conditions. IMLS is defined by the function $f : \mathbb{R}^n \to \mathbb{R}$ such that

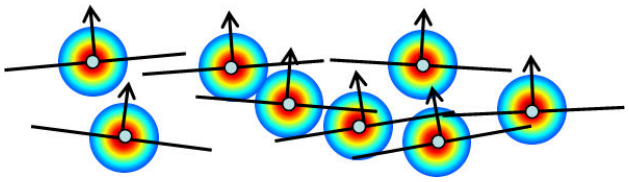
$$f(\boldsymbol{x}) = \frac{\sum_{i \in I} \boldsymbol{n}_i^T(\boldsymbol{x} - \boldsymbol{s}_i)\phi_i(\boldsymbol{x})}{\sum_{i \in I} \phi_i(\boldsymbol{x})} \tag{4.1}$$

$$\phi_i(\boldsymbol{x}) = \exp\left(\frac{-||\boldsymbol{x} - \boldsymbol{s}_i||^2}{2\sigma_i^2}\right)$$

In this equation, I is a set of indexes to points, $\boldsymbol{n}_i$ are the normals defined for each point, $\boldsymbol{s}_i$ are the position of each point, and σ_i is the local smoothing parameter which can either be constant or different for each i.



(a) Visualization of Moving Least Squares. Plane minimizing the local projection distance.



(b) Visualization of Implicit Moving Least Squares. Localized planes fit to each point.

Figure 4.1: Visualization of MLS and IMLS.

This function is extremely similar to the signed distance function defined in [26], but instead of i signifying the distance calculated from entire meshes, now i corresponds to each of the input points. Thus, each input point is considered as a local observation of a plane. The weighting term is used to produce a globally smooth function, but the algorithm is very sensitive to σ_i. Too much smoothing will lose significant geometric features, while too little will let noise produce erroneous results.

Similar to the signed distance function of [26], the IMLS definition can be incrementally updated. Given the value of the signed distance function $f(\boldsymbol{x})$ and $w(\boldsymbol{x}) = \sum_i \phi_i(\boldsymbol{x})$, then for new points J

$$\hat{f}(\boldsymbol{x}) = \frac{w(\boldsymbol{x})f(\boldsymbol{x}) + \sum_{j\in J} \boldsymbol{n}_j^T(\boldsymbol{x} - \boldsymbol{s}_j)\phi_j(\boldsymbol{x})}{w(\boldsymbol{x}) + \sum_{j\in J} \phi_j(\boldsymbol{x})}$$

This incremental update property allows for efficient updating and storage of $f(\boldsymbol{x})$ in a volumetric grid.

One advantage of IMLS over the signed distance function of [26] is that it can be derived from Local Kernel Regression [28]. Let $f(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ be a function that we wish to approximate from noisy observations

$y_i = f(\boldsymbol{x}_i) + \epsilon$. The Taylor expansion of $f(\boldsymbol{x}_i)$ around $\boldsymbol{x}$ is given by

$$f(\boldsymbol{x}_i) \approx f(\boldsymbol{x}) + (\boldsymbol{x}_i - \boldsymbol{x})^T \nabla f(\boldsymbol{x}) + \frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{x})^T H_f (\boldsymbol{x}_i - \boldsymbol{x}) + \cdots$$

where H_f is the Hessian of f and ∇f is the gradient. If we consider only a second order approximation, then the goal is to find $f(x)$ and $\nabla f(x)$ that minimize the local least squares fit.

$$\underset{f(\boldsymbol{x}), \nabla f(\boldsymbol{x})}{\operatorname{argmin}} \sum_i \left[y_i - \left(f(\boldsymbol{x}) + (\boldsymbol{x}_i - \boldsymbol{x})^T \nabla f(\boldsymbol{x}) \right) \right]^2 \phi_i(\boldsymbol{x})$$

where $\phi_i(\boldsymbol{x})$ is a localizing weighting function. Next we make two simplifying assumptions. First, assume that the input points are close to the surface such that $y_i \approx 0$. Second, set $\nabla f(\boldsymbol{x}) = \boldsymbol{n}_i$, where $\boldsymbol{n}_i$ are the normals associated to each input point i; then this second order minimization becomes a first order of the form

$$\underset{f(\boldsymbol{x})}{\operatorname{argmin}} \sum_i \left[f(\boldsymbol{x}) + (\boldsymbol{x}_i - \boldsymbol{x})^T \boldsymbol{n}_i \right]^2 \phi_i(\boldsymbol{x})$$

Taking the derivative and setting equal to zero gives the solution found in Equation (4.1).

Using this derivation from local kernel regression, [28] was able to implement a more robust version of IMLS which they termed Robust Implicit Moving Least Squares (RIMLS). RIMLS has the advantage that it solves a robust minimization instead of least squares and it utilizes a bilateral filtering with respect to the normals. This enables RIMLS to have sharper features than IMLS since IMLS uses an isotropic smoothing function. The disadvantage of RIMLS is that it requires an iterative solution to solve the robust minimization; thus RIMLS is no longer able to be incrementally updated.

Another advantage of IMLS is that, because the signed distance function is analytically defined, it is possible to derive the gradient of $f(\boldsymbol{x})$.

$$\nabla f(\boldsymbol{x}) = \frac{\sum_i \boldsymbol{n}_i^T (\boldsymbol{x} - \boldsymbol{s}_i) \nabla \phi_i(\boldsymbol{x}) + \sum_i \phi_i(\boldsymbol{x}) \boldsymbol{n}_i - \sum_i f(\boldsymbol{x}) \nabla \phi_i(\boldsymbol{x})}{\sum_i \phi_i(\boldsymbol{x})}$$

Unlike $f(\boldsymbol{x})$, $\nabla f(\boldsymbol{x})$ is not incrementally additive. This is because it is a function of $f(\boldsymbol{x})$ which changes given new data. A small approximation can

solve this problem. If we assume that $f^{(i)}(\boldsymbol{x}) \approx f^{(i+1)}(\boldsymbol{x})$, then we can combine gradients as follows:

$$\nabla \hat{f}(\boldsymbol{x}) = \frac{\nabla f^{(i)}(\boldsymbol{x}) w^{(i)}(\boldsymbol{x}) + \nabla f^{(i+1)}(\boldsymbol{x}) w^{(i+1)}(\boldsymbol{x})}{w^{(i)}(\boldsymbol{x}) + w^{(i+1)}(\boldsymbol{x})}$$

If the assumption $f^{(i)}(\boldsymbol{x}) \approx f^{(i+1)}(\boldsymbol{x})$ is violated, it is probably because one set does not have very many points close to $\boldsymbol{x}$; thus more weight is applied to the estimate with points closer.

The IMLS signed distance function is a theoretically sound and practical signed distance function which is capable of being incrementally updated when sampled in a volumetric grid. Thus, IMLS will form the surface representation basis used for integration and registration with our proposed method.

CHAPTER 5

SIGNED DISTANCE REGISTRATION: THEORY

In this chpater we analyze an ideal signed distance function in order to motivate a method for registering a set of points to a signed distance representation of a surface. In particular, we consider how to find the closest point to the surface given the signed distance function and its gradient.

5.1 Properties of a Signed Distance Function

In order to register a new set of points to the signed distance model, we utilize properties of ideal signed distance functions.

Definition 5.1 (Projection)**.** *A Projection $P_S(\boldsymbol{x})$ is the point on the surface S closest to $\boldsymbol{x}$, i.e.,*

$$\|\boldsymbol{x} - \boldsymbol{P}_S(\boldsymbol{x})\| \leq \|\boldsymbol{x} - \boldsymbol{s}\| \quad \forall \boldsymbol{s} \in S$$

Theorem 5.2 (Existence of Projection)**.** *Let S be a compact, orientable surface (i.e. 2-manifold); then there always exists at least one $\hat{\boldsymbol{s}} \in S$ such that for $\boldsymbol{x} \in \mathbb{R}^n$*

$$\|\boldsymbol{x} - \hat{\boldsymbol{s}}\|_2 \leq \|\boldsymbol{x} - \boldsymbol{s}\|_2 \quad \forall \boldsymbol{s} \in S$$

Define

$$\boldsymbol{P}(\boldsymbol{x}) = \hat{\boldsymbol{s}}$$

Proof. Let

$$\delta = \inf_{\boldsymbol{s} \in S} \|\boldsymbol{x} - \boldsymbol{s}\|_2$$

By the definition of inf there exists a sequence $\{\boldsymbol{s}_n\}_{n=1}^{\infty} \subseteq S$ such that

$$\lim_{n \to \infty} \|\boldsymbol{x} - \boldsymbol{s}_n||_2 = \delta$$

Because S is compact there exists a convergent subsequence $\{\boldsymbol{s}_{n_k}\}_{k=1}^{\infty}$ such that

$$\lim_{k\to\infty} \boldsymbol{s}_{n_k} = \hat{\boldsymbol{s}} \in S$$

Thus $\|\hat{\boldsymbol{s}} - \boldsymbol{x}\|_2 = \delta$ which implies that

$$\|\boldsymbol{x} - \hat{\boldsymbol{s}}\|_2 \leq \|\boldsymbol{x} - \boldsymbol{s}\|_2 \quad \forall \boldsymbol{s} \in S$$

■

Assume for the rest of our analysis that S is a compact, orientable surface.

Definition 5.3 (Uniqueness of Projection)**.** *A projection $P(\boldsymbol{x})$ is unique if*

$$\|\boldsymbol{x} - \boldsymbol{P}_S(\boldsymbol{x})\| \leq \|\boldsymbol{x} - \boldsymbol{s}\| \quad \forall \boldsymbol{s} \in S$$

with equality iff $s = P(\boldsymbol{x})$.

Definition 5.4 (Ideal Signed Distance Function)**.** *Let $f : \mathbb{R}^3 \mapsto \mathbb{R}$ be the signed distance function associated with surface $S \subseteq \mathbb{R}^3$. Then*

$$f(\boldsymbol{x}) = \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 \Psi(\boldsymbol{x}) \tag{5.1}$$

where $\Psi(\boldsymbol{x})$ is the sign function determining whether $\boldsymbol{x}$ is inside or outside the surface.

For the rest of this analysis we will consider only a subset of the domain $\{\boldsymbol{x} \in \mathbb{R}^3 : \Psi(\boldsymbol{x}) \geq 0\}$. Thus, we will work with

$$f(\boldsymbol{x}) = \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 \tag{5.2}$$

Results apply for $\{\boldsymbol{x} \in \mathbb{R}^3 : \Psi(\boldsymbol{x}) \leq 0\}$ with appropriate sign changes.

Property 5.5. *The ideal signed distance function (sdf) $f : \mathbb{R}^3 \mapsto \mathbb{R}$ is well defined and continuous*

Proof. The ideal sdf is well defined due to Theorem 5.2 since there always exists a projection and the 2-norm is well defined.

The function f is continuous $\Leftrightarrow \forall\, \epsilon > 0 \quad \exists\, \delta > 0$ such that for $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3$ and $\|\boldsymbol{x} - \boldsymbol{y}\|_2 < \delta$ implies that $|f(\boldsymbol{x}) - f(\boldsymbol{y})| < \epsilon$

$$\begin{aligned}\|\boldsymbol{y} - \boldsymbol{P}(\boldsymbol{y})\|_2 &\leq \|\boldsymbol{y} - \boldsymbol{P}(\boldsymbol{x})\|_2 \\ &\leq \|\boldsymbol{y} - \boldsymbol{x}\|_2 + \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 \\ &\Rightarrow \|\boldsymbol{y} - \boldsymbol{P}(\boldsymbol{y})\|_2 - \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 \leq \|\boldsymbol{y} - \boldsymbol{x}\|_2\end{aligned}$$

By symmetry a similar inequality can be derived starting at $\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2$, which implies that

$$|f(\boldsymbol{x}) - f(\boldsymbol{y})| = \big|\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 - \|\boldsymbol{y} - \boldsymbol{P}(\boldsymbol{y})\|_2\big| \leq \|\boldsymbol{x} - \boldsymbol{y}\|_2 \leq \delta = \epsilon$$

■

Theorem 5.6. *Let $\mathbb{X} \subseteq \mathbb{R}^3$ be the set such that $\boldsymbol{P}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{X}$ is unique. Then for $\boldsymbol{x} \in \mathbb{X}$ and $\boldsymbol{y} = (1 - \lambda)\boldsymbol{x} + \lambda \boldsymbol{P}(\boldsymbol{x})\ \forall\, \lambda \in [0, 1]$*

$$\boldsymbol{P}(\boldsymbol{x}) = \boldsymbol{P}(\boldsymbol{y})$$

and $\boldsymbol{y} \subseteq \mathbb{X}$.

Proof. By definition

$$\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 < \|\boldsymbol{x} - \boldsymbol{s}\|_2 \quad \forall\, \boldsymbol{s} \in S, \boldsymbol{s} \neq \boldsymbol{P}(\boldsymbol{x})$$

Thus,

$$\begin{aligned}\|\boldsymbol{y} - \boldsymbol{P}(\boldsymbol{x})\|_2 &= \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 - \|\boldsymbol{x} - \boldsymbol{y}\| \\ &< \|\boldsymbol{x} - \boldsymbol{s}\|_2 - \|\boldsymbol{x} - \boldsymbol{y}\|_2 \\ &\leq \|\boldsymbol{y} - \boldsymbol{s}\|_2 \quad \forall\, \boldsymbol{s} \in S, \boldsymbol{s} \neq \boldsymbol{P}(\boldsymbol{x}) \\ &\Rightarrow \boldsymbol{P}(\boldsymbol{y}) = \boldsymbol{P}(\boldsymbol{x})\end{aligned}$$

and $\boldsymbol{P}(\boldsymbol{y})$ is unique. ■

Theorem 5.7 (Derivative of Signed Distance Function)**.** *Let $\mathbb{X} \subseteq \mathbb{R}^3$ be the set such that $\boldsymbol{P}(\boldsymbol{x})$ for $\boldsymbol{x} \in \mathbb{X}$ is unique. The gradient of the signed distance function ∇f on $\mathbb{X}$ is equal to*

$$\nabla f(\boldsymbol{x}) = \frac{\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})}{\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2} \tag{5.3}$$

Proof. On $\mathbb{X}$, $\boldsymbol{P} : \mathbb{X} \mapsto S$ is a valid function. Thus we can write

$$\begin{aligned}
\nabla f(\boldsymbol{x}) &= \nabla \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2 \\
&= \frac{1}{2} \frac{\nabla \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2^2}{\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2} \\
&= \frac{[\boldsymbol{I} - \boldsymbol{P}'(\boldsymbol{x})](\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))}{\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2} \\
&= \frac{(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})) - \boldsymbol{P}'(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))}{\|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|_2}
\end{aligned}$$

where $\boldsymbol{P}'(\boldsymbol{x})$ is the Jacobian matrix of the projection.

Claim 5.8. *If $\boldsymbol{P}'(\boldsymbol{x})$ exists, then $\boldsymbol{P}'(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})) = 0$ on $\mathbb{X}$*

By definition of the total derivative

$$\lim_{\|\boldsymbol{h}\|_2 \to 0} \frac{\|\boldsymbol{P}(\boldsymbol{x} + \boldsymbol{h}) - \boldsymbol{P}(\boldsymbol{x}) - \boldsymbol{P}'(\boldsymbol{x})(\boldsymbol{h})\|_2}{\|\boldsymbol{h}\|_2} = 0$$

Let $\boldsymbol{h} = -\delta(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))$. Then

$$\lim_{\delta \to 0} \frac{\|\boldsymbol{P}(\boldsymbol{x} - \delta(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))) - \boldsymbol{P}(\boldsymbol{x}) + \boldsymbol{P}'(\boldsymbol{x})(\delta(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})))\|_2}{|\delta| \|(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))\|_2} = 0$$

By Theorem 5.6, $0 \leq \delta \leq \|\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})\|$ implies that
$\boldsymbol{P}(\boldsymbol{x} - \delta(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))) - \boldsymbol{P}(\boldsymbol{x}) = 0$. Thus

$$\lim_{\delta \to 0} \frac{|\delta| \|\boldsymbol{P}'(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))\|_2}{|\delta| \|(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x}))\|_2} = 0$$

which implies that

$$\boldsymbol{P}'(\boldsymbol{x})(\boldsymbol{x} - \boldsymbol{P}(\boldsymbol{x})) = 0$$

■

For completeness we note that $\boldsymbol{P}'(\boldsymbol{x})$ is not a continuous function and that there exist discontinuities where $\boldsymbol{P}'(\boldsymbol{x})$ is not technically defined. We neglect these cases since the limit of $\nabla f(\boldsymbol{x})$ as it approaches these points exists even if $\boldsymbol{P}'(\boldsymbol{x})$ is not defined.

Corollary 5.9. *For $\boldsymbol{x} \in \mathbb{X}$*

$$\boldsymbol{P}(\boldsymbol{x}) = \boldsymbol{x} - f(\boldsymbol{x}) \nabla f(\boldsymbol{x}) \tag{5.4}$$

Proof. By simple manipulation of Equation (5.3) ■

CHAPTER 6

SIGNED DISTANCE REGISTRATION: ALGORITHM

In this chapter we propose a novel registration algorithm based on the analysis of signed distance functions from the previous chapter, which we call Signed Distance Registration (SDR). Instead of the ideal signed distance function described in Equation (5.1), which is not available from a sampling of surface points, we utilize the IMLS definition described in Equation (4.1). The algorithm consists of registering a new frame of data to a previous IMLS representation of the surface. Once registered, the new frame is incrementally integrated into the IMLS representation. It is then ready for the registration of the next frame.

6.1 Flow Diagram

The following is the overall flow diagram for our algorithm. For notation, let the superscript $^{(i)}$ denote the information and functions associated with the i^{th} frame from the depth camera. Therefore,

- $I^{(i)}$ denotes the set of indices pertaining to data gathered in the i^{th} frame by the camera. For example, let $\boldsymbol{n}_j \ \forall j \in I^{(i)}$ denote the normals of i^{th} frame.
- $T^{(i)}$ denotes the transformation of $I^{(i)}$ to the coordinate frame of $f^{(i-1)}$; thus, $T^{(i)}(I^{(i)})$ includes a transformation of all points and normals indexed by $I^{(i)}$.
- $f^{(i)}$ is the definition of IMLS utilizing the set of transformed sets $\left\{T^{(i-1)}(I^{(i-1)}), T^{(i-2)}(I^{(i-2)}), ..., T^{(2)}(I^{(2)}), I^{(1)}\right\}$.

The algorithm is illustrated in Figure 6.1. As shown, the first data set $I^{(1)}$ is used to generate the first IMLS model $f^{(1)}$, which is then used with

the next set of data $I^{(2)}$ to estimate the transformation from $I^{(2)}$ to $f^{(1)}$. Then this transformation is used with $I^{(2)}$ to generate the next IMLS model. Thus, given any IMLS model $f^{(i)}$ and the next set of points $I^{(i+1)}$, we determine the registration transformation and the next IMLS representation.

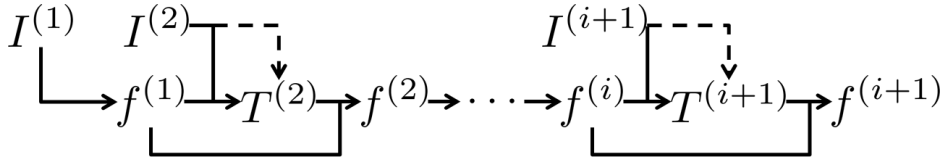


Figure 6.1: Flow diagram for signed distance registration algorithm.

6.2 Input

The input necessary for our proposed algorithm is a sequence of 3D point sets with associated surface normals. In practice, the depth camera only provides a sequence of 3D point sets; therefore, it is necessary to estimate surface normals. In the absence of noise, one could consider a rough approximation of the surface as the quadrilateral mesh formed by connecting 3D points adjacent in the depth image. In this case, surface normal could be estimated by typical methods employed by meshes. But, when one starts to consider depth images with significant noise, mesh approximations quickly fail. Therefore, we employ a more generic method, PlanePCA described in [35], designed for estimating surface normals for general point clouds. The PlanePCA method was shown to be one of the simpler and higher quality, in terms of speed and accuracy, surface normal estimation schemes. PlanePCA is computed for a point $\boldsymbol{p} \in \mathbb{R}^3$ by considering points $\boldsymbol{q}_i$ in the neighborhood around $\boldsymbol{p}$. Let $\boldsymbol{Q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_n, \boldsymbol{p}]^T$ be the matrix of the neighbors of $\boldsymbol{p}$ including $\boldsymbol{p}$ itself. Then PlanePCA solves

$$\underset{\boldsymbol{n}}{\operatorname{argmin}} \|(\boldsymbol{Q} - \bar{\boldsymbol{Q}})\boldsymbol{n}\|_2^2 \tag{6.1}$$

where $\bar{\boldsymbol{Q}}$ is the matrix where all of the rows are equal to the centroid of the points filling $\boldsymbol{Q}$. Equation (6.1) can be solved by selecting the right singular

vector corresponding to the smallest singular value of $(\boldsymbol{Q} - \bar{\boldsymbol{Q}})$. PlanePCA can be understood as fitting the plane that passes through the centroid and has its normal in the direction of minimum covariance. An often computationally heavy task of determining the neighborhood of a given point is simplified by only considering the neighborhood defined on the 2D depth image. In practice, we consider a 5×5 block with $\boldsymbol{p}$ positioned in the middle. In order to not include points across depth boundaries, we also utilize a threshold within the 5×5 block. This threshold is set relative to the size of the scene and is typically 5 to 10 times the average spacing between points in 3D.

One difficulty encountered when fitting a plane to a set of points is that the estimated normal has two equally valid polarities. Normals for adjacent points sampled from a plane can be opposite of one another. Typically this problem is solved by propagating the polarity starting at some arbitrary point. We overcome this difficulty by comparing the direction of the normal with the direction to the center of the camera. If the angle between the normal and the direction to the camera is greater than 90°, then it would have been impossible to capture this point from the depth camera; therefore, we flip the polarity.

6.3 Building a Model

The coordinate frame of the first set of points constitutes the base coordinate frame. All other frames will be registered to the base; therefore, no registration is required with the first frame. The first step of the algorithm is to build a surface representation from the first set of points. As stated in Equation (4.1), the IMLS signed distance function $f(\boldsymbol{x})$ is computed as the weighted sum of signed distances in the direction of the normals for each point. One way to construct the signed distance function is to accumulate points that contribute to this sum. In a similar way, one could consider accumulating points to represent a model that could be utilized by closest point ICP. This is often the simple approach taken to combat registration drift. While this approach allows one to utilize all previous data to combat drift, it means that the computations required to match points will increase over time. Therefore, we consider a way to

implement our algorithm in a space and time efficient manner.

6.3.1 Time and Space Efficient

One of the key challenges of integrating depth camera data over time is managing the large influx of data. In Section 1.2, we state that one of our goals is the use of all available data. If we assume a point accumulation model as a method to use all data, then letting M be the number of relevant 3D points collected each frame, the amount of data that must be considered after N frames is $\mathcal{O}(NM)$. If we consider the closest point ICP variant, its computational costs are linear in the number of considered points. Thus, the longer the algorithm is running, the more computation required and the slower the algorithm will run. Along with computation time, there is also the issue of storing all the previous data. Given $M = 5000$ points/frame, a frame rate of 20 frames/sec and 48 bytes/point, then after only one minute of recording we would have $6,000,000$ points requiring 274 MB of storage space. Modern storage limits make this a manageable amount, but the real difficulty is that all this data would need to be accessed each frame. Therefore, memory bandwidth comes into play and may cause further slowdowns. In order to address the issues of space and time efficiency we propose to sample and store the IMLS model in a volumetric grid of samples, similar to [26].

As mentioned in Section 4.2.2, it is possible to sample the IMLS function and its derivative and incrementally update them given new information. A drawback to this approach is that instead of evaluating the IMLS function at M points each frame, it must be evaluated at G^3 points, where G is the size of one side of a cube defining the volumetric sampling area. But the advantage is that the IMLS signed distance function can then be evaluated in constant time through a tri-linear interpolation of the grid. Thus, by incrementally updating the samples of a grid it is possible to bound the amount of computations required at each frame. This is demonstrated in Figure 6.2 which is a plot of the amount of time necessary to compute the registration indexed by the frame. As can be seen, both the basic SDR and closest point methods grow roughly linearly with time, with SDR growing much faster. But the gridded SDR approach has roughly constant time

performance. Given this bound, optimized parallel algorithms can be developed that have the potential to make this registration perform in real time. An additional bonus is that the gridded IMLS function can be easily converted into a mesh through marching cubes. Thus, after each frame it is possible to view the representative surface.

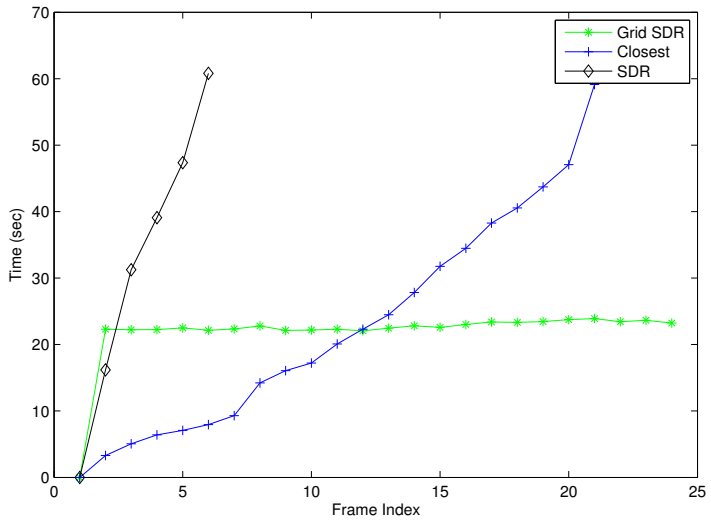
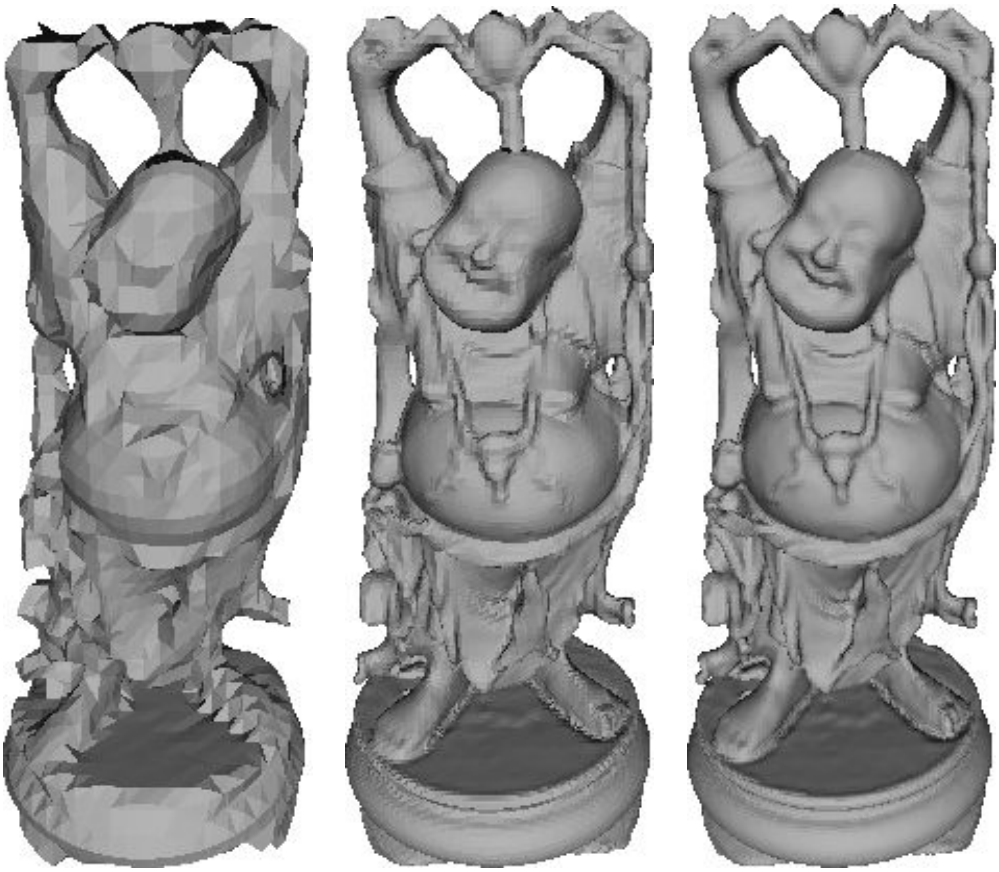


Figure 6.2: Comparison of implementation timing.

The addition of the grid is nice in that it constrains the necessary time and storage, but it also introduces another parameter into the algorithm, namely the grid spacings. Figure 6.3 visualizes IMLS surfaces for three different grid spacings. As the number of samples per unit length increases, more details of the model can be seen. Figure 6.4 compares the registration error for various grid sizes with the error obtained without using a grid and evaluating the IMLS exactly from the set of all previous points. It can be seen from this graph that there reaches a point where the grid is dense enough. This is most likely due to the fact that small details contribute little to the overall registration and that most of the registration is dominated by the macro structure. Further research will need to be done to identify the conditions under which sampling the signed distance function still gives accurate conditions for convergence.



(a) $50 \times 100 \times 50$ (b) $150 \times 300 \times 150$ (c) $250 \times 500 \times 250$

Figure 6.3: Visualization of surface using three different grid spacings. Each grid spacing is in terms of the number of elements in x, y, z.

6.4 ICP Variant

6.4.1 Point Matching

From Corollary 5.4, we are now able to propose a new method of point matching for ICP. Given a signed distance function $f : \mathbb{R}^3 \mapsto \mathbb{R}$ and a set of points P, we consider $\hat{P} = \{\boldsymbol{p} \in P : \boldsymbol{p} \in \mathbb{X}\}$ and define a set of matching points Q to be the projection of $\hat{P}$ onto the surface defined by f, i.e. $\boldsymbol{p}_j \in \hat{P}$

$$\boldsymbol{q}_j = \boldsymbol{P}(\boldsymbol{p}_j) = \boldsymbol{p}_j - f(\boldsymbol{p}_j)\nabla f(\boldsymbol{p}_j) \tag{6.2}$$

This is similar to the closest point criterion originally proposed for ICP except that we are matching points to a continuous surface instead of

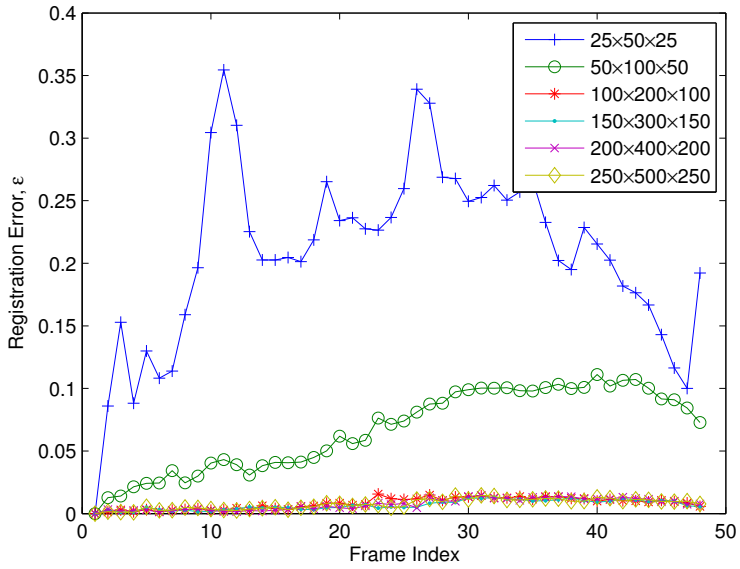


Figure 6.4: Comparison of registration error due to grid spacing.

another set of points. Thus, we would expect that for an ideal signed distance function this matching would be at least as good as closest point matching for the same surface densely sampled.

6.4.2 Error Metric

Given this new definition of point matching we can write the ICP objective function. We will consider the point to plane error metric described in Section 3.1.2.

$$\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}} = \underset{\boldsymbol{R},\boldsymbol{t}}{\operatorname{argmin}} \sum_j \left[\left(\boldsymbol{R}\boldsymbol{p}_j + \boldsymbol{t} - \boldsymbol{q}_j \right)^T \nabla f\left(\boldsymbol{q}_j \right) \right]^2 \tag{6.3}$$

Typically, point to plane error metric is solved by linearizing $\boldsymbol{R}$. This can be done using the Rodrigues angle formula.

$$\boldsymbol{R} = (1 - \cos\theta)\boldsymbol{z}\boldsymbol{z}^T + \cos\theta\boldsymbol{I} + \sin\theta[\boldsymbol{z}]_\times$$

Here $\boldsymbol{z}$ is a unit vector denoting the axis of rotation. By the small angle approximation,

$$\boldsymbol{R} \approx \boldsymbol{I} + \theta[\boldsymbol{z}]_\times$$

Thus, if we let

$$\boldsymbol{r} = \theta \boldsymbol{z} \quad \text{and}$$
$$\nabla f(\boldsymbol{q}_j) = \boldsymbol{n}_j$$

then

$$\begin{aligned}
&= \left(\boldsymbol{R}\boldsymbol{p}_j + \boldsymbol{t} - \boldsymbol{q}_j\right)^T \boldsymbol{n}_j \\
&= \left[\boldsymbol{p}_j + \boldsymbol{r} \times \boldsymbol{p}_j + \boldsymbol{t} - \boldsymbol{q}_j\right]^T \boldsymbol{n}_j \\
&= (\boldsymbol{p}_j - \boldsymbol{q}_j)^T \boldsymbol{n}_j + \boldsymbol{t}^T \boldsymbol{n}_j + (\boldsymbol{p}_j \times \boldsymbol{n}_j)^T \boldsymbol{r}
\end{aligned}$$

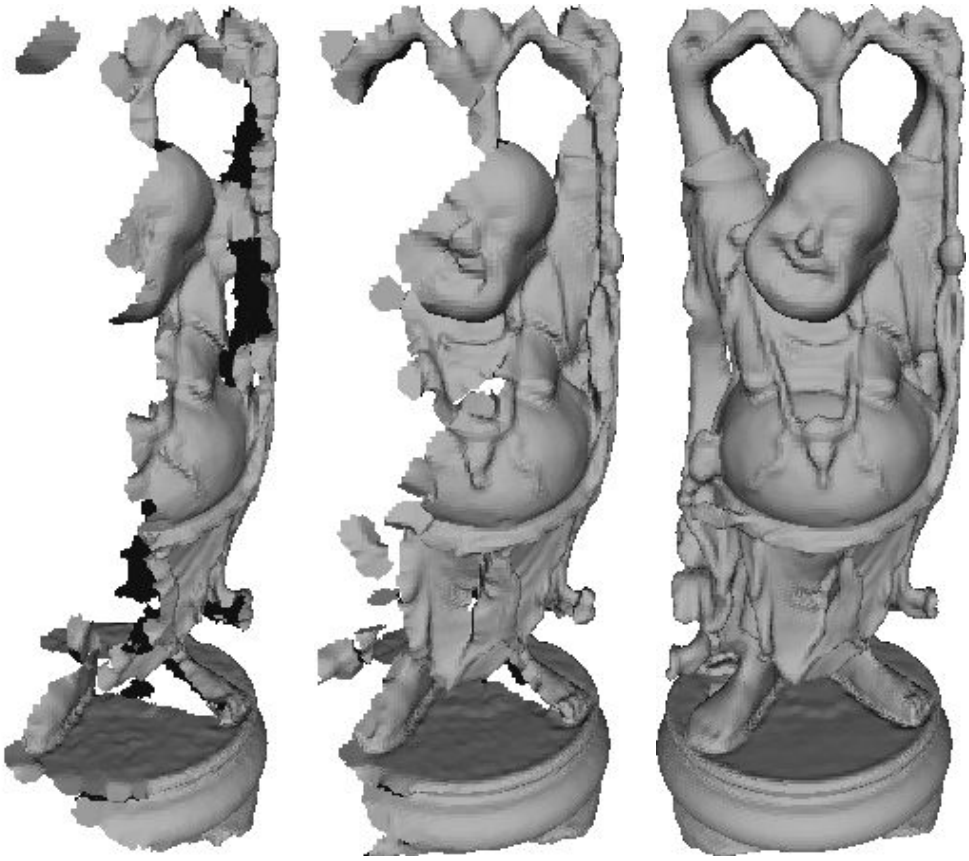
Thus the final objective function is

$$\underset{\boldsymbol{r},\boldsymbol{t}}{\operatorname{argmin}} \sum_j \left[(\boldsymbol{p}_j - \boldsymbol{q}_j)^T \boldsymbol{n}_j + \boldsymbol{t}^T \boldsymbol{n}_j + (\boldsymbol{p}_j \times \boldsymbol{n}_j)^T \boldsymbol{r}\right]^2 \tag{6.4}$$

This can be formulated as a matrix times the unknown parameters $\boldsymbol{r}$ and $\boldsymbol{t}$ and solved using least squares or total least squares.

6.5 Output

The output of our algorithm is a grid representation of the signed distance function. This uniform grid representation is the necessary format to run the marching cubes algorithm that takes an implicit function and converts it to a mesh. The mesh visualization of the output of our algorithm for three different stages of reconstruction can be seen in Figure 6.5.



(a) 1 Frame (b) 5 Frame (c) 20 Frame

Figure 6.5: Visualization of the surface after different numbers of integrated frames. Each visualization is a mesh created from the isosurface function in MATLAB (MATLAB's marching cube implementation) applied to the zero set of the IMLS function sampled in a $150 \times 300 \times 150$ voxel grid.

CHAPTER 7

NUMERICAL RESULTS

In this section we provide results from simulations of our algorithm utilizing synthetic depth images created from high quality 3D mesh models. These simulations allow for a ground truth comparison between previous state of the art registration methods. We provide results utilizing noiseless points and normals, as well as normals calculated from points. We also test our algorithm with depth images that have been perturbed by Gaussian noise with different variances.

7.1 Synthetic Depth Images

In order to have a ground truth registration for comparison of various registration methods, we computed synthetic depth image sequences from known high quality 3D mesh models. OpenGL and other graphic rendering programs often utilize a depth image of 3D meshes in order to speed up renderings. We avoid these depth images since they are often approximations utilized solely for rendering speed purposes. Therefore, we construct our synthetic depth images by projecting each triangle of the mesh onto a user defined image plane. Then for each pixel that the triangle covers we compute a depth from weighted averages of the depths to all of the vertices. The closest depth and corresponding surface normal are saved for each pixel. The synthetic depth maps that we construct are of similar resolution as real depth cameras, 204×204. The depth image sequences are taken with the camera stationary and the 3D model rotating and translating in space. The real advantage of synthetic depth images is that these transformations are known and provide a ground truth for comparison. Another advantage is that the synthetic depth images are essentially noise free. Thus, we can also test the response of the registration

algorithms under known noise variance.

7.2 Registration Error Metric

Given a ground truth registration, we would like to compare the error of our algorithm against previous state-of-the-art registration techniques. In order to have a comparison we need an error metric. This error metric should be invariant to the particular object being registered, but should consider the scale of the object since registration includes a translation vector which is scale dependent. Thus, we propose the following error metric:

$$\epsilon(\boldsymbol{R}, \boldsymbol{t}) = \frac{1}{\lambda} \left(\lambda \| \boldsymbol{R} - \boldsymbol{R}^* \|_2 + \| \boldsymbol{t} - \boldsymbol{t}^* \|_2 \right) \tag{7.1}$$

where λ is the scale of the object, $\boldsymbol{R}^*$ and $\boldsymbol{t}^*$ are the ground truth registrations, and the norms are the matrix and vector norms respectively. For practical purposes we can consider $\lambda \approx \frac{1}{2} \max\{width, length, height\}$ of the object we are considering. The motivation of our metric is that the matrix norm

$$\| \boldsymbol{A} \|_2 = \max_{\|\boldsymbol{x}\|_2 = 1} \| \boldsymbol{A}\boldsymbol{x} \|_2$$

implies $\| \boldsymbol{R} - \boldsymbol{R}^* \|_2 \leq 2$ since rotation matrices are unitary. Thus, λ scales the matrix norm such that it is proportional to the translation error.

7.3 Comparisons

In this section we compare three variants of ICP registration: SDR, Closest, and Projection. Our comparisons are based on 48 frames of synthetic data created as described above.

7.3.1 Noise-Free

Our first experiment is with noise-free points and normals. The results can be seen in Figure 7.1. The errors presented in Figure 7.1 are negligible and all methods provide indistinguishable registrations. Figure 7.2 shows a view of the point cloud from SDR from the front and back as well as a view of

the original mesh that produced the depth images. The holes in the point cloud are from areas where the depth images never covered. Figure 6.5 demonstrated a front view of the mesh produced from the IMLS function evaluated with 20 frames of data. Through all these figures it can be seen qualitatively and quantitatively that all methods provide excellent results.

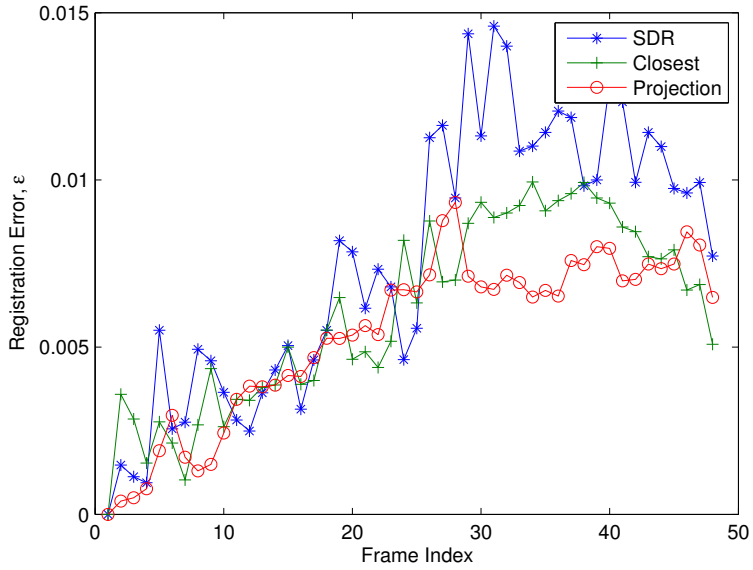


Figure 7.1: Noise-free comparison between ICP variants.

7.3.2 Estimated Normals

Real data would not have ground truth normals along with the data. Thus the first natural extension is to compare the registration methods utilizing noise-free points with estimated normals. For all estimated normals we utilize the normal estimation procedure described in Section 6.2. The results of this experiment can be seen in Figure 7.3. Again, the errors are negligible and all methods perform roughly equivalently. Since the normals are estimated from neighborhoods around each point, Figure 7.4 demonstrates the smoothing effect on the normals even though the points are very accurate. Figure 7.5 looks at the IMLS surface constructed after 1, 5, and 20 frames of data are registered and incorporated into the model.

(a) SDR (b) SDR (c) Original (d) Original

Figure 7.2: Visualization of the set of points registered using SDR with noise-free synthetic data, alongside the original model used to generate synthetic depth images.

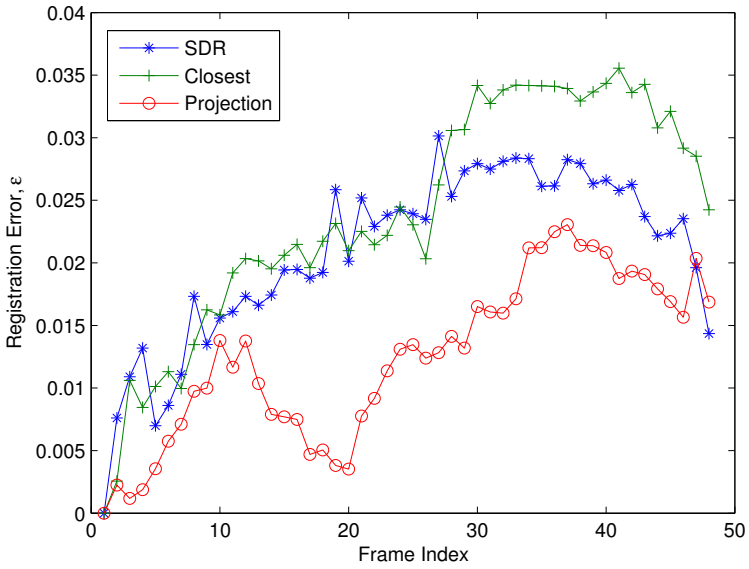
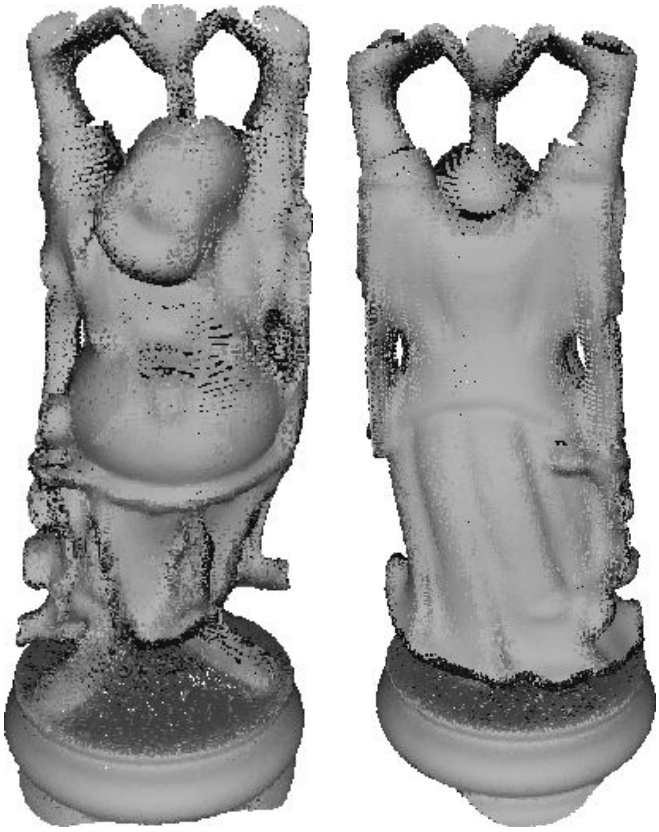


Figure 7.3: Comparison of ICP variants with noise-free points and estimated normals.

7.3.3 Gaussian Noise

Real depth data is also not noise free. Thus in the following set of experiments we demonstrate the performance of our three methods under

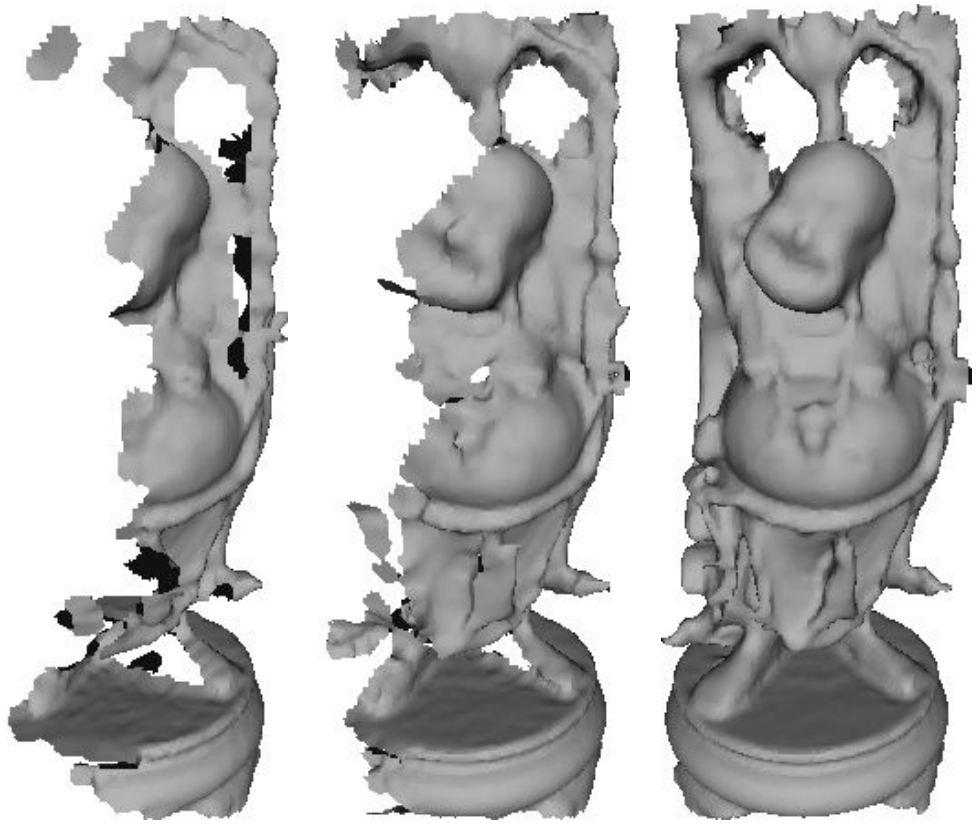


(a) Front (b) Back

Figure 7.4: Visualization of the set of points registered using SDR with noise-free synthetic points and estimated normals.

Gaussian noise applied to the depth image. In this way, 1D Gaussian noise applied at each pixel creates noise in the viewing direction of each pixel, which is an accurate depiction of real noise from depth images. Gaussian noise is reasonable since the shot noise described in Section 2.2.2 will tend to Gaussian noise for sufficiently large numbers of photons, which is generally the case. Real depth images can be modeled as having Gaussian noise with changing variance depending on the amplitude of the received signal. In these experiments we only consider Gaussian noise with a constant variance.

Figure 7.6 compares the registration errors under $\sigma = 0.001$ variance noise. Even for this case of a small amount of noise it is clear that our SDR algorithm begins to provide more accurate registration. In general, it is expected that SDR and closest point ICP would outperform projection ICP since the former two incorporate past information while projection ICP is



(a) 1 Frame (b) 5 Frames (c) 20 Frames

Figure 7.5: Visualization of the mesh constructed from the gridded IMLS function after 1, 5, and 20 frames of data have been incorporated into the model. Points are noise free with estimated normals.

only frame-to-frame registration, but from this figure we can see that projection ICP is slightly better than closest point ICP. This can be explained by the fact that even though closest point ICP is utilizing past data, the noise thickens the surface and will cause closest point ICP to register only to the outside of the thick surface. Thus, after each frame the surface continues to thicken and sway the registration results. Projection ICP does not have this problem since it only considers pair-wise frames.

Figure 7.7 compares the registration errors under $\sigma = 0.002$ variance noise. In this case, SDR proves much more robust than either of the other methods. This can be seen in Figures 7.8, 7.9, and 7.10 which display visible errors for both the closest point and projection ICP methods. An interesting view is the IMLS surface used in the SDR method visualized in Figure 7.11. This demonstrates the smoothing properties of IMLS and

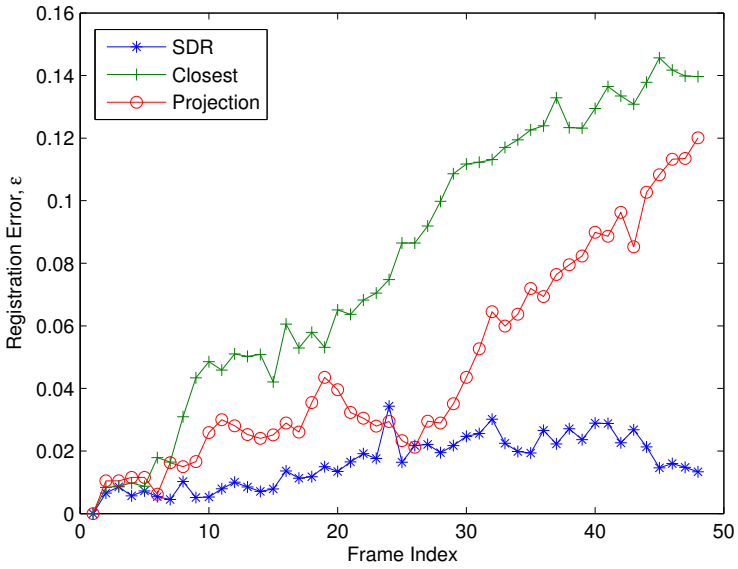


Figure 7.6: Comparison of ICP variants with point noise of variance $\sigma = 0.001$ and estimated normals.

provides insight into why it maintains accurate registration where the other two methods fail.

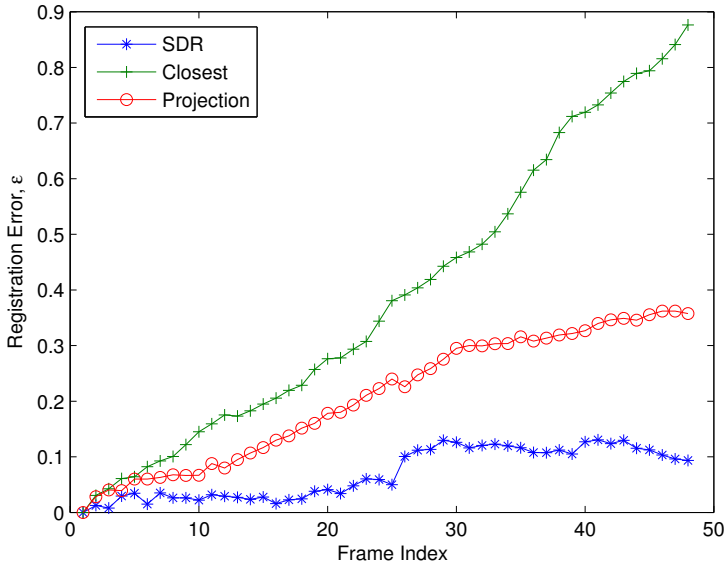
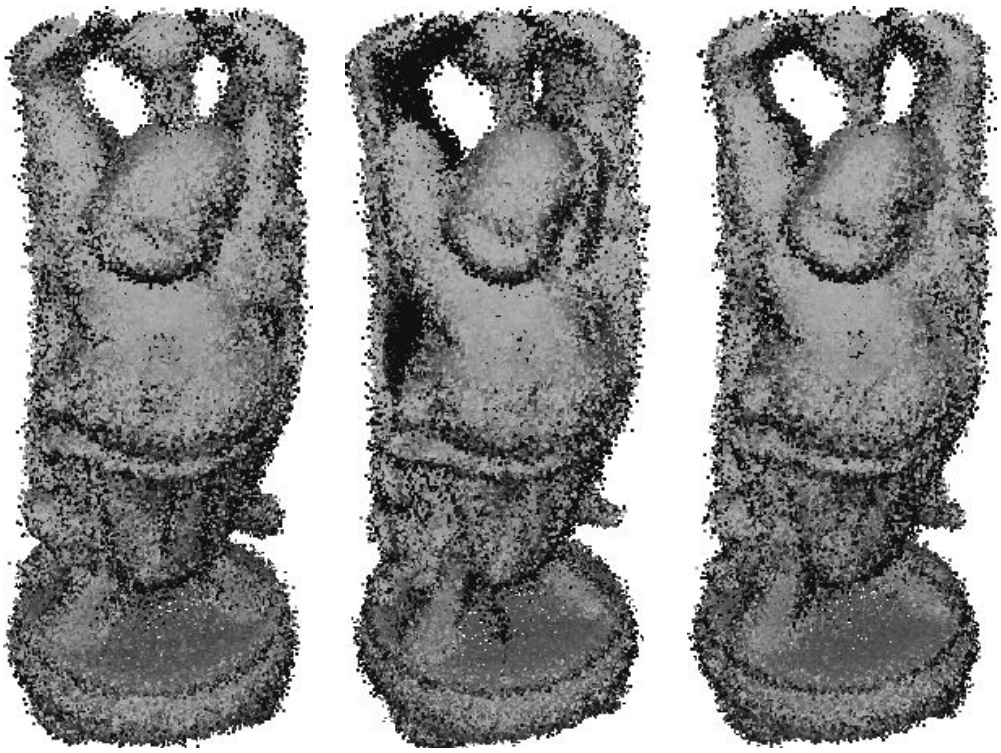


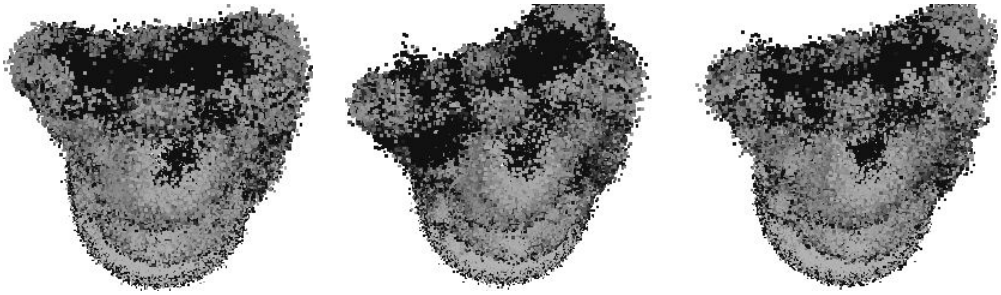
Figure 7.7: Comparison of ICP variants with point noise of variance $\sigma = 0.002$ and estimated normals.

The results for Gaussian noise of variance $\sigma = .04$ can be seen in Figure



(a) SDR (b) Closest (c) Closest

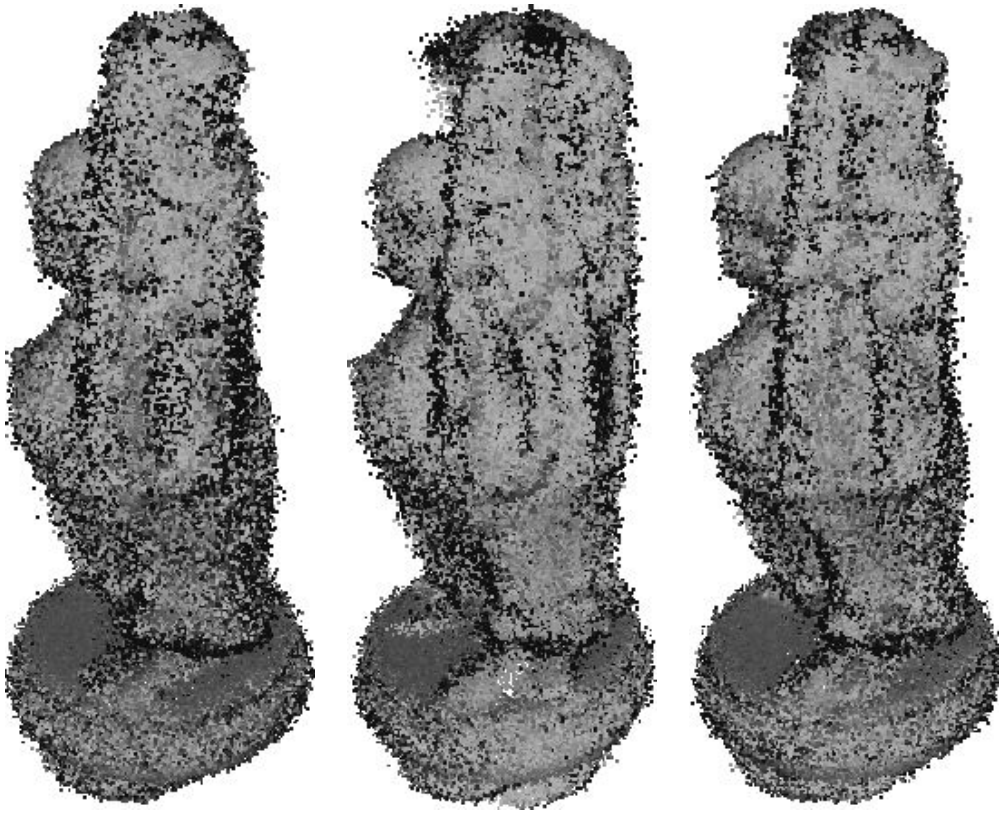
Figure 7.8: Front view of the qualitative comparison of ICP variants with point noise of variance $\sigma = 0.002$ and estimated normals.



(a) SDR (b) Closest (c) Closest

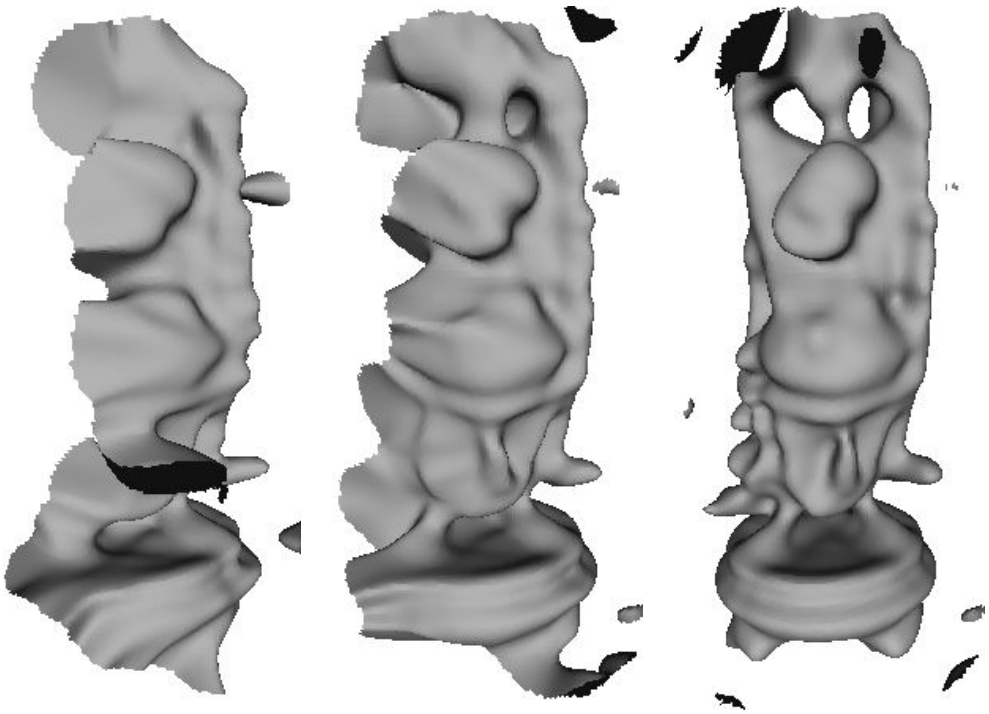
Figure 7.9: Top view of the qualitative comparison of ICP variants with point noise of variance $\sigma = 0.002$ and estimated normals.

7.12. Under this heavy noise, all registration methods show significant error, but SDR shows the least. Errors of this measure are the difference between registration artifacts and total registration failure.



(a) SDR (b) Closest (c) Closest

Figure 7.10: Side view of the qualitative comparison of ICP variants with point noise of variance $\sigma = 0.002$ and estimated normals.



(a) SDR (b) Closest (c) Closest

Figure 7.11: Visualization of the mesh constructed from the gridded IMLS function after 1, 5, and 20 frames of data have been incorporated into the model. Points have noise variance of $\sigma = 0.002$ and the normals are estimated.

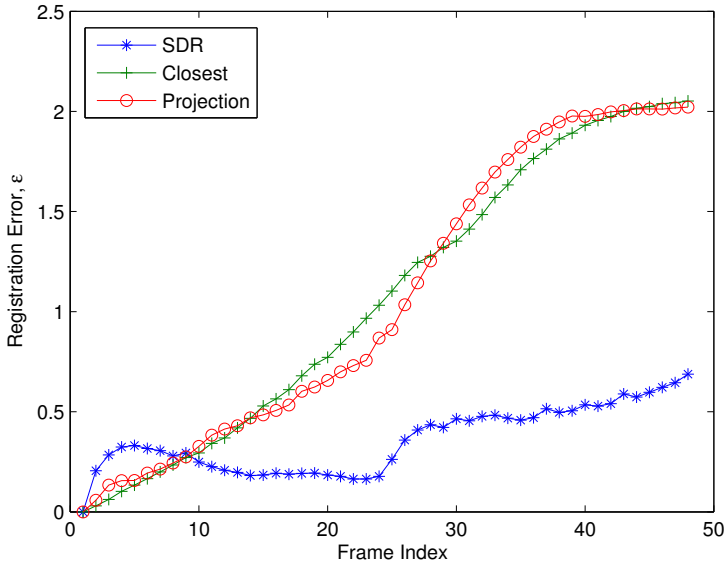


Figure 7.12: Comparison of ICP variants with point noise of variance $\sigma = 0.004$ and estimated normals.

CHAPTER 8

CONCLUSION

In this thesis we propose a combined approach to registration and integration of a streaming depth image sequence in order to overcome the low quality (low resolution and high noise) found in real depth images. Previous methods had either considered these problems separately or not considered a method that could be extended to the necessary streaming nature of depth cameras. The low quality of depth cameras is the current limiting factor to its prolific use in consumer applications. Thus, we propose to combine information over time in order to capitalize on the speed of depth camera technology and overcome its limitations.

We presented current state-of-the-art algorithms that consider the problem of registration and integration separately but are well suited for streaming data and the merging of these algorithms. In particular, current state-of-the-art ICP variants are either fast or capable of utilizing past data but not both. And current state-of-the-art integration methods grow computationally cumbersome after only a few frames of integration. Thus, we present a method that can bound the amount of necessary computations per frame and is well suited for parallel implementation. Thus it is capable of utilizing all previous data and still maintaining a consistent speed over time.

Our analysis in Chapter 5 demonstrated a method for finding the closest point to the surface from the signed distance function and its gradient. Utilizing this analysis, we proposed a new method for point matching in the ICP framework which utilizes the approximate signed distance function defined by IMLS. We also propose a volumetric sampling methodology for storing and updating the IMLS function that allows for bounded computations per time step even after an arbitrary amount of time. As an added bonus, the necessary computations to calculate the IMLS function in a grid at each time step are also the computations needed to run the

marching cubes algorithm for quick production of a mesh each frame. Thus, this registration technique, incorporated into a real-time point rendering system, can share computations.

Finally, we demonstrate through the use of synthetic depth sequences the robustness of our proposed method over both projection and closest point ICP.

REFERENCES

[1] Y. Cui, S. Schuon, D. Chan, S. Thrun, and C. Theobalt, "3d shape scanning with a time-of-flight camera," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2010, pp. 1173–1180.

[2] S. Schuon, C. Theobalt, J. Davis, and S. Thrun, "Lidarboost: Depth superresolution for tof 3d shape scanning," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition CVPR 2009*, 2009, pp. 343–350.

[3] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk, "The digital Michelangelo project: 3d scanning of large statues," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 131–144.

[4] Stanford University Computer Graphics Laboratory, "The Stanford 3d scanning repository," May 2010. [Online]. Available: http://www.graphics.stanford.edu/data/3Dscanrep/

[5] S. Rusinkiewicz, O. Hall-Holt, and M. Levoy, "Real-time 3d model acquisition," in *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '02. New York, NY, USA: ACM, 2002, pp. 438–446.

[6] O. Alexander, M. Rogers, W. Lambeth, J.-Y. Chiang, W.-C. Ma, C.-C. Wang, and P. Debevec, "The digital emily project: Achieving a photorealistic digital actor," *IEEE Computer Graphics and Applications*, vol. 30, pp. 20–31, 2010.

[7] D. Vlasic, P. Peers, I. Baran, P. Debevec, J. Popović, S. Rusinkiewicz, and W. Matusik, "Dynamic shape capture using multi-view photometric stereo," in *ACM SIGGRAPH Asia 2009 papers*, ser. SIGGRAPH Asia '09. New York, NY, USA: ACM, 2009, pp. 174:1–174:11.

[8] T. Beeler, B. Bickel, P. Beardsley, B. Sumner, and M. Gross, "High-quality single-shot capture of facial geometry," *ACM Trans. Graph.*, vol. 29, pp. 40:1–40:9, July 2010. [Online]. Available: http://doi.acm.org/10.1145/1778765.1778777

[9] T. Ringbeck, T. Mller, and B. Hagebeuker, "Multidimensional measurement by using 3-d pmd sensors," *Advances in Radio Science*, vol. 5, pp. 135–146, 2007. [Online]. Available: http://www.adv-radio-sci.net/5/135/2007/

[10] S. Hussmann and T. Edeler, "Pseudo-four-phase-shift algorithm for performance enhancement of 3d-tof vision systems," *IEEE Transactions on Instrumentation and Measurement*, vol. 59, pp. 1175 – 1181, 2010.

[11] S. Hussmann, T. Ringbeck, and B. Hagebeuker, "A performance review of 3d tof vision systems in comparison to stereo vision systems," in *Stereo Vision*, A. Bhatti, Ed. InTech, November 2008. [Online]. Available: http://www.intechopen.com/articles/show/title/a_performance_review_of_3d_tof_vision_systems_in_comparison_to_stereo_vision_systems

[12] R. Lange and P. Seitz, "Solid-state time-of-flight range camera," *IEEE J. Quantum Electron.*, vol. 37, no. 3, pp. 390–397, 2001.

[13] S. Rusinkiewicz and M. Levoy, "Efficient variants of the ICP algorithm," in *Third International Conference on 3D Digital Imaging and Modeling (3DIM)*, June 2001, pp. 145 –152.

[14] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," *Image Vision Comput.*, vol. 10, pp. 145–155, April 1992. [Online]. Available: http://portal.acm.org/citation.cfm?id=138628.138633

[15] A. Nuchter, K. Lingemann, and J. Hertzberg, "Cached k-d tree search for icp algorithms," in *Proceedings of the Sixth International Conference on 3-D Digital Imaging and Modeling*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 419–426.

[16] P. J. Neugebauer, "Geometrical cloning of 3d objects via simultaneous registration of multiple range images," in *Proc. Conf. Int Shape Modeling and Applications*, 1997, pp. 130–139.

[17] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-9, pp. 698–700, 1987.

[18] B. K. P. Horn, "Closed-form solution of absolute orientation using unit quaternions," *J. Opt. Soc. Am. A*, vol. 4, no. 4, pp. 629–642, Apr 1987. [Online]. Available: http://josaa.osa.org/abstract.cfm?URI=josaa-4-4-629

[19] M. W. Walker, L. Shao, and R. A. Volz, "Estimating 3-d location parameters using dual number quaternions," *CVGIP: Image Underst.*, vol. 54, pp. 358–367, October 1991. [Online]. Available: http://portal.acm.org/citation.cfm?id=119076.119080

[20] D. W. Eggert, A. Lorusso, and R. B. Fisher, "Estimating 3-d rigid body transformations: a comparison of four major algorithms," *Mach. Vision Appl.*, vol. 9, pp. 272–290, March 1997. [Online]. Available: http://portal.acm.org/citation.cfm?id=250152.250160

[21] S.-W. Shih, Y.-T. Chuang, and T.-Y. Yu, "An efficient and accurate method for the relaxation of multiview registration error," *IEEE Trans. Image Process.*, vol. 17, no. 6, pp. 968–981, 2008.

[22] K. Pulli, "Multiview registration for large data sets," in *Proc. Second Int 3-D Digital Imaging and Modeling Conf*, 1999, pp. 160–168.

[23] Q.-X. Huang, B. Adams, and M. Wand, "Bayesian surface reconstruction via iterative scan alignment to an optimized prototype," in *Proceedings of the fifth Eurographics symposium on Geometry processing*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 213–223.

[24] P. Claes, D. Vandermeulen, L. Van Gool, and P. Suetens, "Robust and accurate partial surface registration based on variational implicit surfaces for automatic 3d model building," in *Proc. Fifth Int. Conf. 3-D Digital Imaging and Modeling 3DIM 2005*, 2005, pp. 385–392.

[25] R. M. Bolle and B. C. Vemuri, "On three-dimensional surface reconstruction methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 1, pp. 1–13, 1991.

[26] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1996, pp. 303–312.

[27] C. Shen, J. F. O'Brien, and J. R. Shewchuk, "Interpolating and approximating implicit surfaces from polygon soup," in *Proceedings of ACM SIGGRAPH 2004*. ACM Press, Aug. 2004, pp. 896–904.

[28] C. Oztireli, G. Guennebaud, and M. Gross, "Feature preserving point set surfaces based on non-linear kernel regression," *Computer Graphics Forum*, vol. 28, no. 2, p. 493501, 2009. [Online]. Available: http://vcg.isti.cnr.it/Publications/2009/OGG09

[29] N. Amenta and Y. J. Kil, "Defining point-set surfaces," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04. New York, NY, USA: ACM, 2004, pp. 264–270.

[30] G. Guennebaud and M. Gross, "Algebraic point set surfaces," *ACM Trans. Graph.*, vol. 26, pp. 23.1–23.9, July 2007. [Online]. Available: http://doi.acm.org/10.1145/1276377.1276406

[31] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer, "Parameterization-free projection for geometry reconstruction," in *ACM SIGGRAPH 2007 papers*, ser. SIGGRAPH '07, vol. 26, no. 3. New York, NY, USA: ACM, 2007, pp. 22.1–22.5.

[32] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," in *ACM SIGGRAPH Asia 2009 papers*, ser. SIGGRAPH Asia '09. New York, NY, USA: ACM, 2009, pp. 176:1–176:7.

[33] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, ser. SIGGRAPH '87. New York, NY, USA: ACM, 1987, pp. 163–169.

[34] R. Kolluri, "Provably good moving least squares," *ACM Trans. Algorithms*, vol. 4, pp. 18:1–18:25, May 2008. [Online]. Available: http://doi.acm.org/10.1145/1361192.1361195

[35] K. Klasing, D. Althoff, D. Wollherr, and M. Buss, "Comparison of surface normal estimation methods for range sensing applications," in *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ser. ICRA'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 1977–1982.