3D Global Shape Descriptors Applied in Scan Registration

Dissertation with the aim of achieving a doctoral degree
at the Faculty of Mathematics, Informatics and Natural Sciences
Department Informatics
of Universität Hamburg

submitted by Bo Sun
2015 in Hamburg

Day of oral defense: 13.11.2015

The following evaluators recommend the admission of the dissertation:

Prof. Dr. Jianwei Zhang

Prof. Dr. Stefan Wermter

I would like to dedicate this thesis to my loving parents, my dear wife and my lovely daughter, all of whom had a major influence in my life, goals and ambitions.

Acknowledgements

Doing a PhD is an exciting but challenging journey, which would be impossible without the help and support of many people. First of all I am grateful to my supervisor Prof.Dr. Jianwei Zhang for giving me the opportunity to pursue a PhD at University of Hamburg, and providing me such a liberal academic environment.

Writing a dissertation is no easy task, but the same can be said of reviewing it. I would like to thank all the people who helped by reviewing parts of this work, especially Prof. Dr. Stefan Wermter.

All the members of our TAMS group are greatly acknowledged, the time spent with you was wonderful and memorable. My sincere thanks go to Tatjana (Lu) Tetsis for helping me so much and so kindly during the past four years. Being away from my family was tough, however, many friends from Hamburg provided another family away from home. I own special thanks to my Chinese friends in Hamburg and Benjamin Adler together with his family. In this regards, Jianhua Zhang, Junhao Xiao, Guoyuan Li, Gang Cheng, Jian Chen, Haojun Guan, Junhu He, Liwei Zhang and Bingwei He deserve special mention.

Finally, I would like to thank the people closest to me for their love and support and for reminding me that work is not everything in life: my parents, my wife Hui Wang, especially to my little daughter Ziyao Sun who brings me a lot of happy moments.

The China Scholarship Council (CSC) is gratefully acknowledged for funding me during my studied in Hamburg.

Bo Sun

Hamburg, August 2015

Abstract

With the advent of low-cost and high-performance depth sensors, the usage of three-dimensional (3D) point clouds is becoming attractive and increasingly popular. Meanwhile, there is a growing demand to process and understand the 3D data. 3D scan registration is the cornerstone of several advanced 3D data processing techniques, and the odometry-free scan registration of 3D point clouds is a research hot spot recently. For the present, the majority of state-of-the-art registration methods depend on reliable initial estimates or local salient features. The 3D registration techniques based on global shape descriptors have not attracted as much attention as they deserve. Generally speaking, the original 3D data is depicted by a global descriptor, and the transformation between the original data could be solved by aligning the corresponding global descriptors. In this thesis, we propose two novel registration methods based on two global shape descriptors, namely Hough Transform Descriptor and Spherical Entropy Image respectively.

For the Hough Transform Descriptor-based registration method, the original 3D scans are projected into the Hough domain. In this way, 3D rotation of the original data is decoupled from its 3D translation, and then the rotation and translation between the original data could be recovered separately. The rotation is estimated firstly by aligning the corresponding translation invariant Hough Transform Descriptors, and there is only translation between the original data after rotating them according to the estimated rotation matrix. The Phase Only Matched Filter (POMF) is employed to estimate the translation between the rerotated scans. We also propose a novel shape descriptor named Spherical Entropy Image (SEI), and develop a novel registration method based on SEI aided by the Spherical Harmonic analysis techniques. Since SEI is not translation-invariant, it is impossible to estimate the rotation and translation separately as the aforementioned Hough Transform Descriptor-based registration method does. In our SEI-based registration algorithm, we integrate the rotation estimation and translation recovery into an iteration framework, which is one of our major contributions.

Besides, the possibility of using SEI as a local shape descriptor in feature matching task is also discussed.

Elaborate experiments with regard to public available datasets and the dataset captured by our custom-built platform are implemented to validate the efficiency of our proposed registration algorithms. The experiment results illustrate the parameter-insensitivity, runtime stability, high reliability and efficiency of our novel algorithms in the registration of featureless, partially overlapping and largely transformed 3D scan pairs.

Kurzfassung

Die Verwendung dreidimensionaler Punktwolken gewinnt durch die zunehmende Verbreitung kostengünstiger und leistungsfähiger Tiefenkameras an Bedeutung. Daraus entsteht die Anforderung, diese Daten sinnvoll interpretieren und verarbeiten zu können. Dreidimensionale Registrierung einzelner Punktwolken ist ein elementares Verfahren in der Verarbeitung, wobei bislang zur Registrierung a-priori Schätzungen der Scan-Transformationen benötigt wurden. Momentan stehen besonders die odometrie-freien Registrierungsalgorithmen im Fokus der Forschung. Hierbei werden die 3D-Daten durch globale Deskriptoren abgebildet, wonach die Transformation zwischen einzelnen Scans durch Ausrichtung der jeweils übereinstimmenden Deskriptoren erreicht werden kann. Diese Dissertation stellt zwei neue Ansätze zur Registrierung von Punktwolken vor: den Hough-Transform-Deskriptor und das Spherical-Entropy-Image.

Zur Anwendung des Hough-Transform-Deskriptors wird die Punktwolke in Hough-Domain projeziert, so dass 3D Rotation und Translation entkoppelt werden. Anschließend können Rotation und Translation unabhängig von einander rekonstruiert werden: die Rotation wird durch Ausrichtung der translationsinvarianten Hough-Transform-Deskriptoren geschätzt. Nach Anwendung der errechneten Rotationsmatrix verbleibt lediglich eine Translation, die dann durch Anwendung des Phase Only Matched Filter (POMF) rekonstruiert wird. Ferner werden Spherical-Entropy-Image (SEI) Deskriptoren sowie das darauf basierende Verfahren der Spherical Harmonic Analysis vorgestellt. Da SEI nicht translationsinvariant ist, können Rotation und Translation nicht unabhängig von einander ermittelt werden. Deswegen wird die Schätzung beider Parameter mit Hilfe einer iterativen Architektur erreicht. Zusammen mit der Möglichkeit, SEI auch zur lokalen Oberflächenbeschreibung zu verwenden, gehört diese Architektur zu den wichtigen Beiträgen dieser Dissertation.

Die vorgestellten Verfahren werden durch Experimente an mehreren Datensätzen validiert. Dazu gehören sowohl öffentlich zugängliche Beispieldaten sowie Daten von einer eigens konstruierten, mobilen Scan-Plattform. Die Resultate dieser Experimente bescheinigen die Parameter-Insensitivität, Laufzeitverhalten, Zuverlässigkeit und Effizienz der Algorithmen.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Nomenclature

Acronyms / Abbreviations

FFT	Fast Fourier Transform
$FPFH$	Fast Point Feature Histograms
HTD	Hough Transform Descriptor
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
$POMF$	Phase Only Matched Filter
$3DSC$	3D Shape Context
SEI	Spherical Entropy Image
$SHOT$	Signature of Histograms of Orientation
SHT	Spherical Harmonic Transform

CHAPTER 1

Introduction

A journey of a thousand miles starts beneath one's feet.

Laozi

Contents

1.1 Introduction of shape descriptor

THree-dimensional (3D) data analysis is a hot research topic due to the continuous and rapid improvement of 3D sensing technologies and the increasingly popular applications of 3D data. 3D shape descriptors are used to collate the information stored in sensed 3D point clouds so that surfaces can be represented more compactly and compared efficiently. 3D shape descriptor is a significant 3D data analysis technique which is related to many 3D data processing tasks, such as 3D scan registration, robotic mapping, 3D object recognition, surface matching and 3D shape retrieval.

1.1.1 Definition

3D shape descriptors are widely used in 3D computer vision and 3D computer geometry communities for comparing 3D surfaces or detecting the similarity between 3D surface patches. A 3D shape descriptor is the compact representation of the original 3D data and maintains as much information as possible; in this way it could be used to compare the original data effectively and efficiently. A 3D shape descriptor is usually in the form of a vector calculated based on the geometry information of point clouds; in other words, a 3D shape descriptor could be regarded as a function or black box whose input is the point permutation of point clouds (sometimes combing the intensity information of points) and output is a compact vector extracting the information contained in point clouds.

A 3D shape descriptor is a trade-off between compactness and descriptiveness. It tries to describe the original data using the shortest vector without loss of information, which is impossible though. Hence, different kinds of shape descriptors with regard to different applications are designed by balancing compactness and loss of information. Since a large amount of 3D shape descriptors are proposed due to its significance, several taxonomies of 3D shape descriptors are available.

The shape descriptors could be discriminated according to whether they preserve the orientation information of original data or not. Specifically, the **object-oriented shape descriptors** are calculated under an object-oriented reference frame; in this way, they are rotation-invariant and exclude the orientation and position information of objects. Object-oriented shape descriptors are usually employed as local features in 3D object recognition or as global features in 3D object classification. The **viewer-oriented shape descriptors** are calculated under a viewer-oriented reference frame and preserve the orientation information of original 3D point clouds. These kinds of shape descriptors have not attracted much attention yet. The two global shape descriptors proposed in this thesis belong to the viewer-oriented shape descriptors. Since the viewer-oriented shape descriptors maintain orientation information of the original data, they could be used to recover the transformation between the original data.

According to size of the support on which descriptors are calculated based, 3D shape descriptors could be divided into **global shape descriptors** and **local shape descriptors**. The local shape descriptors encode the information of local neighbourhood around the keypoints, while for the global shape descriptors, they describe the entire surface. The local shape descriptors are naturally suited to handle clutter and occlusions. The global shape descriptors are prone to occlusions, but they are more descriptive in terms of objects with poor geometric structures such as household objects. Generally speaking, the majority of feature-based scan registration and object recognition algorithms use the local shape descriptors, and the majority of 3D object classification and shape retrieval techniques employ the global shape descriptors.

In addition, the shape descriptors could be categorized into **signature-based descriptors** and **histogram-based descriptors**, which inherently emphasize descriptiveness and robustness respectively. Signature-based shape descriptors describe 3D surfaces by encoding one or more geometric measurements computed individually at each subset of the 3D surface. Signature-based methods are potentially highly descriptive thanks to use the of spatially well-localized information. On the other hand, the histogram-based shape descriptors describe 3D surfaces by encoding counters of local geometrical entities into histograms according to a specific classification criteria. In broad terms, signatures are potentially highly descriptive thanks to the use of spatially well-localized information, whereas histograms trade off descriptive power for robustness by compressing geometric structure into bins.

1.1.2 Application in data registration

3D shape descriptors are the foundation of several advanced 3D data processing techniques, such as 3D object recognition, 3D shape retrieval and 3D scan registration. In this thesis, we focus on the application of shape descriptors in 3D scan registration, and propose two global shape descriptors which could be used in 3D scan registration efficiently and outperform the state-of-the-art methods.

Data registration is the process of spatially aligning two or more datasets of an object or scene based on the overlap. The alignment process could determine the correspondences between points in different datasets, enable the fusion of information and estimate the motion of sensors. Besides, if identities of objects in one of the datasets are available, identities of objects and their poses in another dataset can be determined by registering the two datasets. The task of data registration is to find an optimal geometric transformation between corresponding datasets. The data might be 2D images, 3D point clouds (textured or not), 3D volumes and so on, which are taken at different times, using various sensors, and from diverse viewpoints. Data registration could be interpreted as a box taking two or more datasets as inputs and the geometric transformation between the input datasets as output.

Specifically, 3D scan registration is the cornerstone of several advanced 3D data processing techniques, e.g., model-based object recognition and mobile robotic mapping. The odometry-free registration of 3D point clouds of large outdoor scenes is a research hot spot recently, especially in the scenarios where stereo vision is not precise enough and Global Positioning System (GPS) is unavailable (e.g., underwater robots and mine robots). Moreover, the registration techniques which are successful in large outdoor scenes could be easily applied to indoor scenes.

For the present, the majority of existing registration methods depend on local shape descriptors. The 3D local shape descriptor-based registration methods inherit the philosophy of 2D feature-based image registration techniques, and make use of explicit feature correspondences in the overlap of datasets. The common procedure of local shape descriptor-based registration methods includes: keypoints extraction, shape description, shape descriptors matching, transformation estimation and refinement. The 3D local shape descriptor-based registration techniques are intensively studied and widely used, and various local shape descriptors have been employed for 3D scan registration. The local shape descriptor-based registration methods could deal with scan pairs with partial overlaps and large offsets. Besides, they could also handle clutter and occlusion effectively. But the local shape descriptor-based registration methods confront challenges about how to eliminate the mismatches, especially in large scale scenes with numerous similar 3D patches. Furthermore, the performances of the local shape descriptor based registration algorithms are closely related to some crucial parameters, such as the size of support, threshold of feature correspondences. It means the parameters used in local shape descriptor-based methods should be adjusted carefully through numerous trials based on professional experience. It is more frustrating that the parameters which perform well in one instance are highly possible to fall down in other instances.

Less attention is paid to registration methods based on global shape descriptors so far. The global shape descriptor-based registration methods could also be called global feature-less registration algorithms since they are not dependent on specific surface patches which are salient and rich in geometrical information. The global feature-less registration algorithm depict the original 3D data by a global shape descriptor, and the transformation between the original data could be solved by matching the corresponding global descriptors. Certainly, the global shape descriptor should maintain the orientation and position information of the original data. Generally speaking, the global shape descriptor-based registration methods are prone to occlusion and the existing techniques have not achieved wide employment. But we believe the global shape descriptors-based registration methods have untapped potential.

1.2 Motivation

The 3D scan registration is the cornerstone to Mobile Robotic Mapping, and acquiring the precise spatial models of physical environments where robots locate themselves is one of the fundamental issues in building truly autonomous mobile robots, since mapping is essential to the subsequent tasks, such as path planning and robot navigation.

In the context of 3D rigid scan registration, the registration methods could be classified into local alignment methods and global alignment methods according to whether an initial estimate is required. In practice, the initial estimate is usually provided by Inertial Measurement Unit (IMU). However, the drift of information provided by IMU is nearly inevitable. The local registration methods could give much precise results *when they succeed*, so they are usually taken as a refinement technique of some global alignment algorithms, which only give a coarse solution. Practical application shows that the local registration methods have the following three distinct drawbacks:

1. requirement of good initial guesses;
2. easy to get trapped in local minima;
3. runtime would vary considerably for different scan pairs, even with same size.

On account of the shortcomings of the local alignment methods, global methods that make use of the global appearance of 3D scan are increasingly popular. The majority of global registration methods describe the global appearance of 3D scan by the combination of many local shape descriptors. Moreover, the local shape descriptors employed by this kind of feature-based registration methods should be calculated around the keypoints which are salient under an object-oriented reference frame. The common procedure of feature-based registration methods includes: keypoints extraction, feature description, feature matching, transformation estimation and refinement. The performances of this kind of registration methods depend heavily on the correctness of feature matching. Although many techniques have been developed to eliminate the false feature correspondences, the mismatches are still the main reason causing false registration results. The scan registration methods based on local shape descriptors could cope with large offset, partial overlap and occlusion efficiently for the cases in which they succeed. But the wide usage found that they have the following shortcomings:

1. they depend on the salient features of the scan pairs, and give unpromising results with regard to objects with poor geometric structures such as mugs and cans;

2. tiny impropriety of crucial parameters could lead to failure of registration, so numerous trials should be implemented to fix the parameters carefully;
3. they are susceptible to noise since they only use part of the information of the original data;
4. fluctuation of processing time is also a problem for this kind of registration methods.

So in a word, the motivation of this thesis contains three aspects: the significance of 3D point clouds registration, deficiency of state of the art registration algorithms, and untapped potential of global shape descriptors applied in scan registration.

1.3 Objectives

3D scan registration aims to calculate the geometric relation between two overlapped 3D point clouds. Assume we have two 3D scans S_1 and S_2, the scan registration task computes the rotation R and translation T between them. The translation T is usually represented by a vector containing three elements, while the rotation R has several different representations, please refer to Appendix B for the details. From the view of mathematics, the scan registration task solves the following equation:

$$S_1 = R * S_2 + T \tag{1.1}$$

given S_1 and S_2. Equation 1.1 is widely used in literatures to explain the scan registration task. But strictly speaking, it is inaccurate. Because usually the S_1 and S_2 are partially overlapped, which means some points in S_2 can not find their correspondences in S_1. Equation 1.1 does not hold for all points in S_1 and S_2.

Let $S_{1o} \subseteq S_1$ represent the points in S_1 which have correspondences in S_2. By the same token, let $S_{2o} \subseteq S_2$ represent the points in S_2 which have correspondences in S_1. Then a more accurate description of scan registration task is:

$$S_{1o} = R * S_{2o} + T \tag{1.2}$$

Certainly, the S_{1o} and S_{2o} are unknown, and that is an important part of scan registration task.

Considering the significance of 3D scan registration and deficiency of existing registration methods, we aim to develop new 3D scan registration strategies based on global shape descriptors to make the registration methods have the following characteristics:

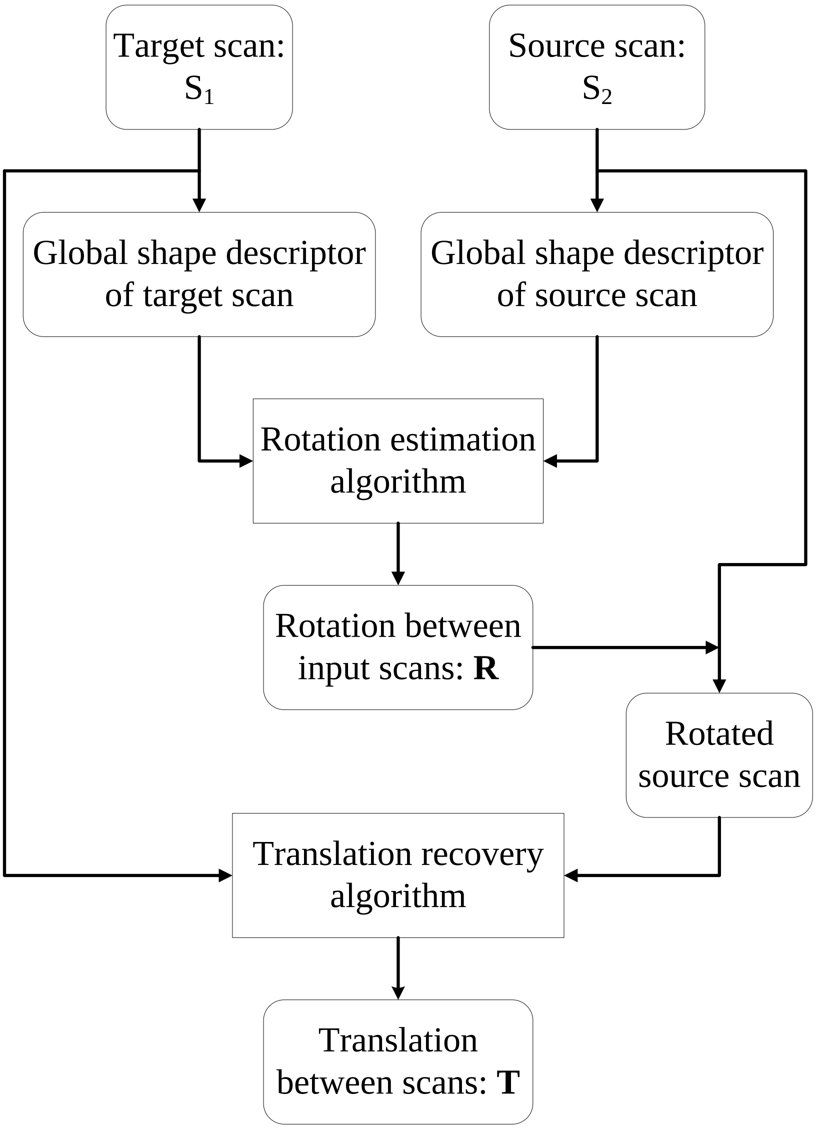

Target scan: S_1
Source scan: S_2
Global shape descriptor of target scan
Global shape descriptor of source scan
Rotation estimation algorithm
Rotation between input scans: **R**
Rotated source scan
Translation recovery algorithm
Translation between scans: **T**

Fig. 1.1 Diagram of the proposed scan registration pipeline.

- **feature-less:** the global shape descriptors-based registration methods should not depend on any specific features;
- **noise-immune:** the novel methods ought to resist the noise effectively by using the overall appearance of scans;
- **partially overlapping:** the registration methods are supposed to effectively deal with partially overlapped scan pairs;

- **parameter insensitivity:** the insensitivity to specific parameters would help the registration methods to be used by laymen;
- **runtime stability:** the stability of runtime is quite significant for practical applications.

We propose two scan registration methods based on two different global shape descriptors respectively. And these two registration methods share the same pipeline, which is depicted in Figure 1.1. The similarity and difference between these two registration methods is discussed in Chapter 5. Generally speaking, the registration pipeline contains three key problems:

1. how to describe the original 3D point clouds using the compact and descriptive shape descriptors;
2. how to determine the rotation between 3D scans based on the global shape descriptors;
3. how to recover the translation between 3D scans.

1.4 Outline

The rest of this thesis is organized as follows:

Chapter 2 broadly surveys the existing different kinds of shape descriptors, reviews the Fourier analysis techniques applied in scan registration, and presents several public resources related to 3D data analysis.

In Chapter 3, we introduce the way how to convert 3D point clouds into Hough domain to achieve the Hough Transform Descriptor, and propose the registration method based on Hough Transform Descriptor.

Chapter 4 presents the details of Spherical Entropy Image including motivation, implementation and discussion. Besides the scan registration algorithm employing Spherical Entropy Image as a global shape descriptor, we also discuss the possibility using Spherical Entropy Images as a local shape descriptor.

Chapter 5 concludes the thesis, compares the two scan registration methods proposed in this thesis, and points out the future work.

1.5 Publications related to this thesis

Some of the work presented in this dissertation has been published in two conference papers.

- Bo Sun, Wewei Kong, Liwei Zhang, Jianwei Zhang (2014). Fourier Analysis techniques applied in data registration: A survey. Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014 International Conference on, pages 1-5, Beijing, China [135]
  Part of Chapter 2
- Bo Sun, Weiwei Kong, Junhao Xiao, and Jianwei Zhang (2014). A Hough Transform based Scan Registration Strategy for Mobile Robotic Mapping. In Proceeding of IEEE International Conference and Robotics and Automation (ICRA), pages 4612–4619, Hongkong, China. [134]
  Part of Chapter 3

CHAPTER 2

State of the Art

> Stones from other hills may serve to polish jade.
>
> *Shijing·Xiaoya·Heming*

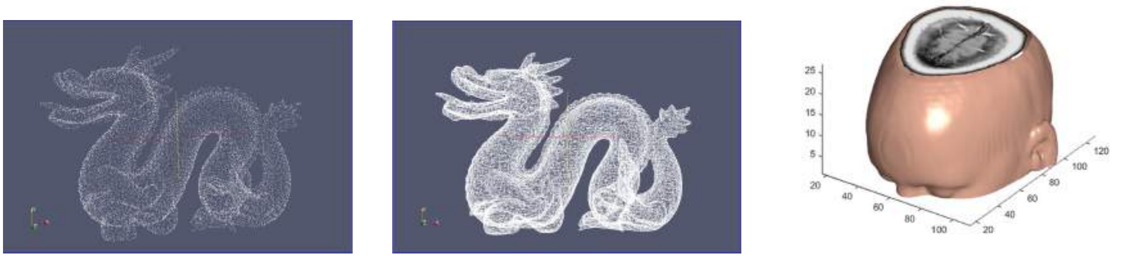
Contents

2.1 Background

IN the wake of new-generation and low-cost depth sensors, the three-dimensional (3D) data is becoming easily available. Moreover, with the increase of computational power, the usage of 3D data is more and more popular. This leads to an increasing amount of 3D data and breeds the advancement of 3D data processing techniques, such as 3D data registration, 3D object recognition and 3D shape retrieval. Admittedly, the 3D data processing techniques are closely related to the two-dimensional (2D) image processing techniques, and a large amount of 3D data analysis algorithms are extended from or inspired by the corresponding 2D image analysis algorithms. However, the difference between shape representations of 3D and 2D data makes some 2D computer vision techniques incapable in 3D data analysis, especially when the photometric properties are not available in 3D data. In 3D data processing, the objects are typically depicted by geometry rather than appearance.

In this section, we discuss the category of 3D data, which is commonly disregardful, and then talk about the capture techniques of 3D data.

2.1.1 3D data category



(a) (b) (c)

Fig. 2.1 Examples of 3D data: (a) and (b) Dragon from "Stanford 3D Scan Scanning Repository" [133] and displayed by *Paraview* [75]. (c) A MRI volumetric image as a example included in MATLAB [147].

When the 3D data is acquired by a range scanner, only the *boundary* of the object is available. In these cases, which are quite common in reality, the object is represented as a *surface*, or called 2D manifold in differential geometry. The internal structure of the object is almost only available in medical applications, where the 3D data is represented as volumetric images. We classify the 3D data into three categories: *Point Cloud*, *Polygon Mesh* and *Volumetric Image*. Point Cloud and Polygon Mesh only represent the surfaces of objects, while Volumetric Image contains internal structure of objects in addition to the boundary information. With adding or removing some information, these three kinds of data could be converted to each other.

Point Cloud

A 3D **Point Cloud** is a set of points in 3D space, and these points are usually defined by x, y and z coordinates and contain one or more channels of data. Photometric information of points, such as intensity and RGB information, is given as a scalar or a vector which is referred as texture. Geometric information of points, such as normal, curvature, range, acquisition viewpoint, could also be given. Please note that the photometric or geometric information is dispensable, and in general only the locations of the points are available in majority of 3D point clouds. PCD (Point Cloud Data) [106] is a typical and recent point cloud file format developed by Point Cloud Library (PCL). An example of Point Cloud is shown in Figure 2.1a.

Polygon Mesh

Besides the 3D points, **Polygon Mesh** also contains connectivity information between the points, such as edges and faces. The edges are defined by two points, or named vertices in mesh; the faces are defined by three or more vertices. The faces usually consist of triangles, quadrilaterals or other simple convex polygons. Among them the triangles are most common, and *Triangle Mesh* is an important type of polygon mesh. PLY (Polygon File Format) [155] is a typical polygon file format developed at Stanford University by Turk *et al.*. An example of Polygon Mesh is shown in Figure 2.1b.

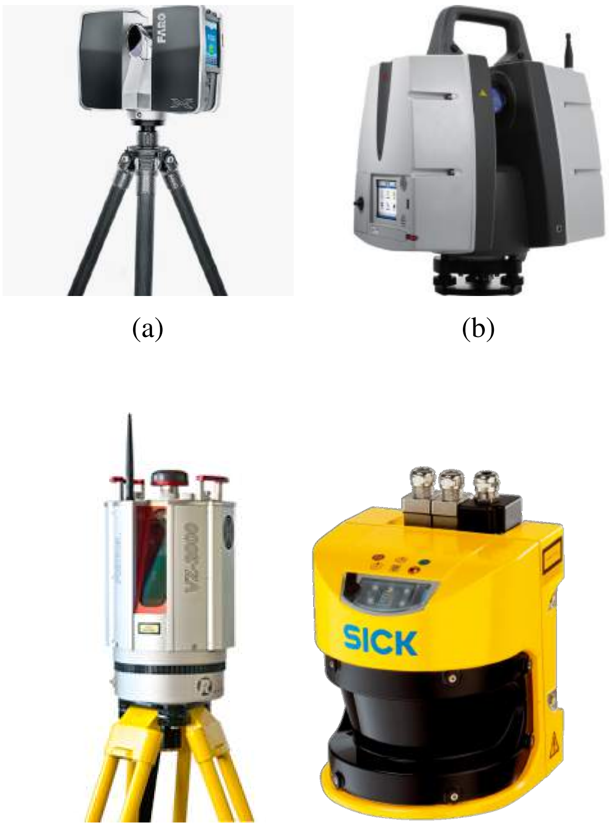
Volumetric Image

Aided by medical imaging techniques, such as X-ray computed tomography (CT), Magnetic Resonance Imaging (MRI) and Positron Emission Tomography (PET), the detection of internal structure of organs could be available. **Volumetric Image** is a common representation for this kind of 3D data. Volumetric Image could be regarded as a 3D matrix, where each voxel describes the properties of objects. And Volumetric Image is the direct extension from 2D image to 3D space. Volumetric Image could also be treated as a stack of 2D image slices. DICOM (Digital Imaging and Communications in Medicine) [39][116] is the most popular file format for volumetric images. DICOM is developed by *DICOM Standards Committee*. An example of Volumetric Image is presented in Figure 2.1c.

In this section, we just mention one of the most typical file format for each data category, please refer to [17][160] and [80] for more information about file formats of 3D data. Naturally, the same data can be represented by different file formats, and by the same token a file format could represent different kinds of data.

2.1.2 3D data acquisition techniques

Since we just focus on the **Point Cloud** data in this thesis, only the techniques to capture 3D point clouds are discussed in this section. We classify the popular 3D scanners into four categories: Laser Scanner, Structured Light Scanner, Photogrammetry and Sonar. Thereinto, Photogrammetry belongs to passive methods, while the other three belong to active methods.



(a) (b)

(c) (d)

Fig. 2.2 Examples of Laser Scanners: (a) FARO Focus 3D X 130 [153]; (b) Leica ScanStation P30 & P40 [83]; (c) RIEGL VZ-2000 [81]; (d) SICK S3000 Professional [128].

Laser Scanner

Laser scanner, or LiDAR is a broad term that includes different kinds of technologies employing laser to estimate the locations of 3D objects. Four laser scanners are depicted in Figure 2.2. Most basically, LiDAR emits laser pulses and collected feedback relating to what happens to the emitted laser pulses. Naturally, the more pulses LiDAR collects, the more points could be sensed. According to the type of feedbacks LiDAR analyses, the LiDAR could be divided to pulse-based LiDAR and phase-based LiDAR. Pulse-based LiDAR is also named as Time-of-Flight scanner. Generally as we known, most of laser scanners are pulse-based for the present.

As suggested by its name, time-of-flight LiDAR is an active scanner that determines the location by recording the time between bouncing off the laser pulse to an object and receiving the feedback of laser pulse. In this way, the accuracy of a time-of-flight LiDAR is heavily dependent on the precision of its timer. The time-of-flight LiDAR scanners could measure the objects pretty far away, but only perceive the objects in its direction view. That means if the time-of-flight scanners want to scan the entire field of view, they have to change the range finder's direction of view either by rotating the range finder itself or using a system of rotating mirrors. So the time-of-flight scanners are relative slow compared with other kinds of LiDAR, and they can collect about $50,000$ points in a second.

The phase-based LiDARs are applied with lasers that continuously emit laser beams, so they are also named as continuous wave laser scanner. This kind of laser scanning systems modulate the power of the laser beams. The phase shift between the emitting laser beam and the returning laser beam is employed to calculate the distance that the laser beam traveled. The phase-based LiDARs are very fast and could gather about one million points per second. Besides, the phase-based laser scanning systems could give highly precise measurements. But phase-based LiDAR could only sense the object within 80 meters.

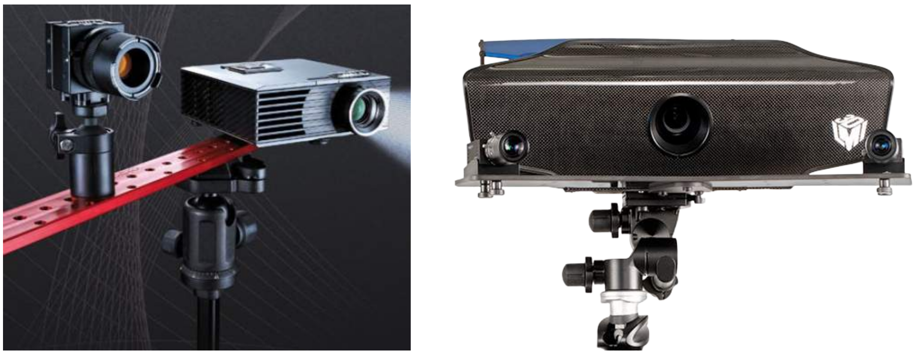

(a) (b)

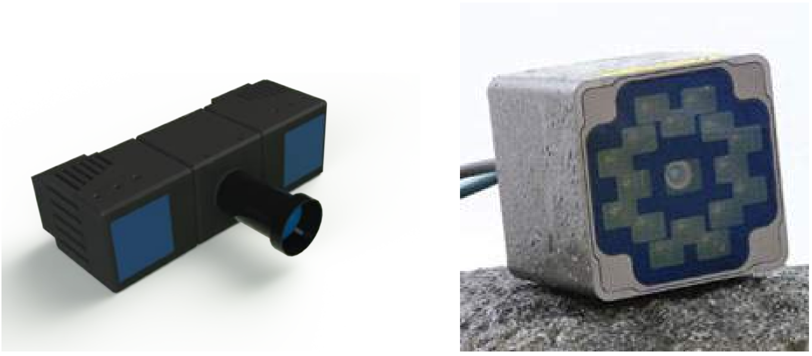
Fig. 2.3 Examples of Structured Light Scanners: (a) David SLS-2 [37]; (b) HDI Advance R3 [87].

Structured Light Scanner

Structured Light Scanners usually contain light sources and cameras. The light source projects a pattern of light onto an object, the camera records the distortion of light pattern and then used for the geometric reconstruction of the surface. The patterns of parallel stripes are most widely used. The structure light scanners could be divided into static scanners, of which the light source and camera are in fixed location and hand-held scanners which could be swept over an object.

The precision of structured light scanners depends on several factors: optical quality of projector, wavelength of light, camera resolution, display resolution, calibration algorithm and so on. Overall, the precision and range of structured light scanners is far below that of laser scanning systems. But they are much cheaper than the professional laser scanning systems. Two structured light 3D scanners are presented in Figure 2.3.

It is worth to note that the Time-of-Flight cameras could also be regarded as a special kind of structure light scanners. Two examples of Time-of-Flight cameras are depicted in Figure 2.4. Besides, *Microsoft Kinect* [161] is probably the most famous hand-held Time-of-Flight camera.



(a) (b)

Fig. 2.4 Examples of Time-of-Flight cameras: (a) PMD CamCube 3.0 [109]; (b) FOTONIC E70 [49].

Photogrammetry

3D data could also be captured based on a block of overlapped images, and this technique is called Photogrammetry or Stereoscopy. Photogrammetry is a passive capturing technique.

(a) (b)

Fig. 2.5 Examples of stereoscopic vision systems: (a) ZED 2K Stereo Camera [131]; (b) Bumblebee XB3 1394b [110].

2D image-based 3D data acquisition technique has made great progress in the last few years. The close-range photogrammetry has gained the level where common digital cameras could be employed to estimate the 3D position of the close-range points. Unlike the early photogrammetry technique which requires the manually identification of same targets in the different images, the recent photogrammetry software is able to automatically identify similar features in different photographs. Although becoming relatively ubiquitous, the quality of photogrammetry systems and their produced 3D data varies widely. The inexpensive photogrammetry systems are consumer-oriented, but have little chance to be used for scientific or industrial purpose.

One of the major drawbacks of the image-based 3D data acquisition techniques is the requirement to illumination. Without an illumination source, the photogrammetry can not work. By the same token, the objects in dark areas could not be well located. Two stereo cameras are shown in Figure 2.5.

Sonar

Sonar, which is an acronym for "SOund Navigation And Ranging", uses sound waves to determine positions of objects. Sonar is traditionally adopted in 2D technology, and the recent advancements allow it to be used to create 3D point clouds of underwater environments.

Sonar systems could be categories into *passive sonar* which just listens the sound made by external sound sources, and *active sonar* which emits sound pulses and listens the echoes. And active sonar systems are widely used in 3D data capture. Normally, an active sonar system contains a sound transmitter and a receiver. Active sonar systems create a sound pulse called a "ping", and then collect the reflections of the pulse, commonly named "echo". The distance between the objects and active sonar system is measured based on the travelled time of the sound pulse aided by the speed of sound. Two examples of sonar scanners are presented in Figure 2.6.

The sonar systems used for 3D data capture contains three sub-categories: single beam sonar, multi-beam sonar and robust multi-beam sonar. Single beam sonar is the old and traditional type of sonar, which acquires 3D position in a narrow path with a "ping". It could be imagined as gathering one point at a time. Multi-beam sonar is the most widely used, and it creates very dense point clouds of underwater objects. Multi-beam sonar emits sound waves in a fan shape and extracts directional information from the returning sound waves based on the spatial filtering techniques. In this way, multi-beam sonar systems could produce a swath of depth readings from a single ping. Robust multi-beam sonar could use a large number of sonar beams at a high rate to create real-time 3D point clouds of underwater environments. Unlike the traditional multi-beam sonar which needs a post-process, the robust multi-beam sonar could create real-time 3D point clouds just like video. At the present speaking, the robust multi-beam sonar is not widely used because it is incredibly expensive and overkills for most of present applications.

(a) (b)

Fig. 2.6 Examples of Sonar Imaging systems: (a) Eclipse 3D Multibeam Imaging Sonar [154]; (b) Echoscope Real Time 3D Sonar [33].

2.2 3D shape descriptors

2.2.1 The taxonomy

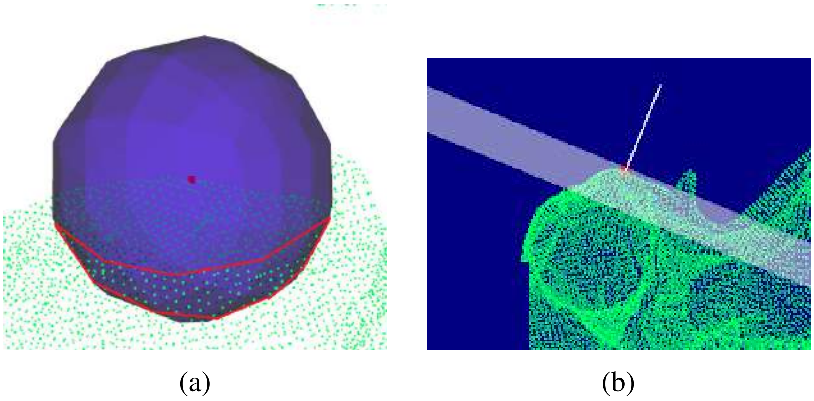
3D shape descriptor is a compact representation of the original 3D data and simultaneously maintains as much information as possible. It is a trade-off between compactness and information loss. Since 3D shape descriptor is an important basic factor for many computer vision and computer geometry applications, such as scan registration [124] [156], object recognition [54] [24], surface matching [108], shape retrieval [143], object classification [13], there are quite a large amount of shape descriptors in literature. Considering the diverse shape descriptors, it is necessary to category the shape descriptors in order to know the state-of-the-art well.

Taxonomies in specific area with regard to particular applications are available. For example, Tombari *et al.* divide 3D descriptors into two main categories: namely *Signatures* and *Histograms*. [150] [123] And Guo *et al.* classify existing 3D descriptors into three broad categories: signature-based, histogram-based and transform-based methods. [54] But the above-mentioned two taxonomies are presented under the context of 3D object recognition applications. Jonhan and Remco divide the feature descriptors into four kinds: global features, global feature distributions, spatial maps and local features [143]. Papadakis *et al.* category the shape descriptors into methods based on 2D representations, methods based on 3D representations and hybrid methods [103]. Actually, the view-based shape descriptors are increasingly popular and achieve better overall performance compared to the 3D representation-based shape descriptors in shape retrieval community. And these two taxonomies are also discussed in 3D shape retrieval community.

In our taxonomy, we category the existing 3D shape descriptors into **Local Shape Descriptors** and **Global Shape Descriptors** according to the support size. It is worthy to note that 3D shape descriptors based on local features mentioned in [143] belong to global shape descriptor in our taxonomy, since they are gathered up to describe the whole 3D scan. The popular view-based shape descriptors in shape retrieval application also belong to global shape descriptor in our taxonomy. And we discriminate the global shape descriptors according to whether they preserve the orientation and position of original data or not. Generally speaking, the shape descriptors calculated under object-oriented reference frame exclude the orientation and position information, and they are usually used in 3D shape retrieval and object classification. The shape descriptors calculated under viewer-oriented reference frame preserve the orientation and position information of the original 3D point clouds. This kind of shape descriptors are not a speck of public interest at present and deserve more attention. The global shape descriptors preserving orientation and position could be employed for scan registration and surface matching, which is one of the motivations of this thesis.

The global descriptors and local descriptors are reviewed separately, and each of them is reviewed chronologically. We try to describe the shape descriptors in brief, plain and understandable language to make them accessible even for the non-professionals. Unlike the majority of scientific literatures to analyse the theories behind the algorithms, we describe the algorithms in the view of implementation. In our view, this way is effective and easy to understand. For some methods, we point out their labels in order to echo available taxonomies, for example, signature-based or histogram-based for local shape descriptors, and 3D representation-based or view-based for global descriptors. And limited by the space, it is impossible to propose all shape descriptors, actually we present the popular and widely used ones. Further, although we divide the shape descriptors based on their support size, the global shape descriptors and local shape descriptors are connected closely. For example, the widely used local descriptors Point Feature Histograms and Fast Point Feature Histograms are directly extended from the global descriptors Surflet-Pair-Relation Histograms.

2.2.2 Local shape descriptors



(a) (b)

Fig. 2.7 Illustration of typical shape descriptors (a) Point Signature (b) Spin Image.

Point Signature shape descriptor is proposed by Chua and Jarvis [31] in 1997. For each keypoint p, compute the shared points C between a sphere centred at p with radius r and the surface, for example the points within red lines in Figure 2.7a, and calculate the best-fit plane P for all the shared points C, this could be achieved by RANSAC (RANdom SAmple Consensus) or SVD (Singular Value Decomposition) technique in our opinion. The normal of plane P is recorded as n_1. Then, translate the fitted plane P following the direction of n_1 to point p to form a new plane $P^{'}$ and then project the points in C onto $P^{'}$ to get a new points set $C^{'}$. Find the point q in $C^{'}$ which has largest distance to its correspondence in C, and n_2 is defined by the unit vector from p to q. Please note that q is on plane $P^{'}$ and $P^{'}$ parallels P, so n_2 is perpendicular to n_1. The *Point Signature* of keypoint p is expressed by the point set C with each point expressed by two parameters: the signed distance between the point and its corresponding point in$C^{'}$, and the clockwise rotation angle fromn_2 around n_1. For the implementation of *Point Signature*, we do not agree with the opinion regarding the intersection computing of a sphere with the surface is difficult [124][54], the intersection could be achieved by finding the points on surface with distance r to keypoint. It is not a difficult problem both in theory and in implementation.

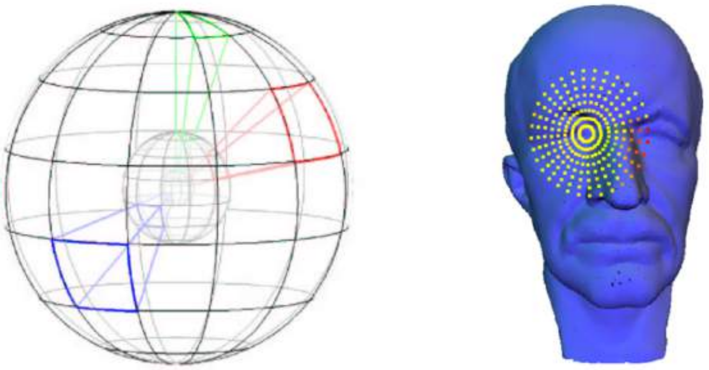
Johnson and Hebert present *Spin Image* in 1998 [66][65]. For each keypoints p, compute its normal and the tangent plane, just like Figure 2.7b shows. And then project every point in its support q onto the tangent plane to get the signed distance between the projection $q^{'}$ and keypoints p. By the way, the support of a reference point in local shape descriptor context is the sets of points whose distance to the reference point is less than a fixed threshold. The distance is named α, and the sign of α is determined by the direction of normal of p. The distance between q and its projection $q^{'}$ is recorded as β. In this way, every point in support is expressed by two scalars α and β. A 2D histogram is built by accumulating the number of points that fall into the bins indexed by (α, β). Finally, the 2D histogram is processed by bilinear interpolation to form the *Spin Image*. *Spin Image* is employed in many applications and regarded as a benchmark to evaluate the local shape descriptors. But its descriptive power is limited by omitting the cylindrical angular coordinate. However, please note that omitting cylindrical angular coordinates is a somehow helpless choice, because of the ambiguity of coordinate system. Many invariants of *Spin Image* also prove its popularity, such as a spherical spin image [117], a spin image signature [5] and a scale invariant spin image [36].

Point's fingerprint proposed by Sun and Abidi consists of a set of 2D contours [137] [138]. For each keypoint p, construct a local reference frame based on its normal and tangent plane. Then generate 3D geodesic circles around p by finding the points which have the same geodesic distances to p. The 3D geodesic circles could be calculated with fast marching methods [125]. Project the 3D geodesic circles onto the tangent plane of p to achieve 2D contours. The resultant 2D contour is similar to human fingerprint, so it is called *Point's fingerprint*. By the way, geodesic distance is the distance measured along the shortest route between two points on the surface. *Point's fingerprint* is also a 2D image-based representation just like Spin Image, but more discriminative than Spin Image.

In 2002, Yamany and Farag proposed a new surface representation called*Surface Signature* for the purpose of scan registration and surface matching [165]. Regardless of complicate theory, it is quite easy to compute *Surface Signature* from the view of implementation. For each keypoint p, estimate the surface normal on p, which could be achieved by PCA (Principal Component Analysis) by the way. For each point q in the support of p, compute its two parameters: the first is its distance to p; the second is the angle between vector $\overrightarrow{qp}$ and normal of p. *Surface Signature* is a 2D histogram indexed by these two parameters.

An irregular local surface representation is proposed by Chen and Bhanu in 2004 [27][28]. Instead of computing local shape descriptor based on all the surrounding points, this kind of representation selects points whose normals have large difference with the normal of reference point in neighbourhood. A 2D histogram indexed by shape index and the dot product of surface normals is built. Similarly with *Spin Image*, the 2D histogram is processed by bilinear interpolation. The abnormality is that the 2D histogram is not the final shape descriptor. The final local shape representation comprises of this 2D histogram, the centroid of *Local Surface Patch*, and the surface type which is determined by the Gaussian and mean curvatures.

All the aforementioned five shape descriptors try to describe the support of keypoint using a 2D histogram. It is a trend at the beginning of 21st century. Generally speaking, these shape descriptors usually build the local reference frame on keypoint using normal and tangent plane, and they try to project the surrounding points onto tangent plane and extract features based on the projections. But things changed when *3D Shape Context* appeared in 2004.



(a) (b)

Fig. 2.8 Illustration of typical shape descriptors (a) 3D Shape Context, reprinted from [51] (b) Normal Based Signature (NBS), reprinted from [85].

3D *Shape Context* is first presented by Frome *et al.* [51] in 2004. It is a straightforward extension of successful 2D shape context presented in [11]. For each keypoint p, estimate its surface normal and determine the spherical support centred at p and its north pole parallels the surface normal. Then divide the spherical support region equally along the azimuth and elevation dimensions, but logarithmically along the radial dimension, just like Figure 2.8a shows. Each bin stores the weighted number of points located in each sub-volume. The calculation of 3D *Shape Context* does not involve any computational complex procedure. But the problem is that the original 3D *Shape Context* has a degree of freedom in the azimuth direction, and calculation of the 3D *Shape Context* along all azimuth directions requires large memory. Frome *et al.* also present the "Harmonic Shape Context (HSC)" to solve the ambiguity along azimuth by applying the Spherical Harmonic Transform (SHT) to the original 3D Shape Context. However, HSC performs inferiorly than 3D*Shape Context.* The exciting news is that a Unique Shape Context (USC) is proposed by Tombari *et al.* in 2010 [149], which builds a repeatable and dis-ambiguous local reference frame associating the keypoint. As reported in [149], USC decreases the memory requirement dramatically and performs better feature matching accuracy than the original 3D *Shape Context*.

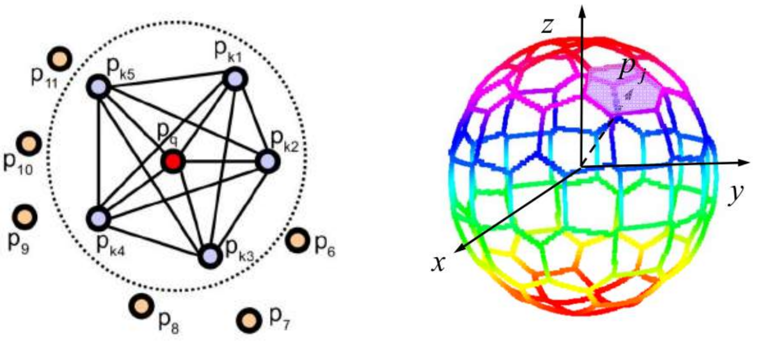
Li and Guskov propose a local surface signature named "Normal Based Signature (NBS)" in 2005 [85]. Normal and tangent planes are estimated for each keypoint p and sample a disc around p on tangent plane into $N \times M$ grids just like Figure 2.8b depicts. Then project the normal of each point q in support onto the direction connecting the keypoint p and grid nodes, and a $N \times M$ matrix stores the projections. Since there is a freedom along the radial direction of disc, a discrete cosine transform and a discrete Fourier transform is applied to the $N \times M$ matrix successively to address this ambiguity. Finally, the upper left corner of the magnitude of discrete Fourier coefficients is used to form the NBS local shape descriptor. The NBS performs well in surface matching application as reported in [85]. NBS continues the trend of using projections on the tangent plane to describe the local surface.

Snapshot is a computational efficient local surface descriptor proposed in 2007 by Malassiotis and Strintzis [97]. The Local Reference Frame (LRF) is built for each keypoint p based on eigenvalue decomposition on the covariance matrix of the coordinates of points within its support. A pin-hole camera is located on the z axis of LRF with a distance d away from p. The x and y axes of the camera coordinate system share the same direction with the x and y axes of LRF, while the z axis is the inverse direction of z axis of LRF. The points on the surface are projected onto the image plane by the virtual pin-hole camera. The common pin-hole camera could be depicted by a function which inputs 3D points and outputs 2D positions on camera image plane of the 3D points. So the projection is an easy procedure known the internal geometrical structure of the virtual pin-hole camera. The *snapshot* stores the least distance between the 3D points and camera image plane. *snapshot* is computed at least one order of magnitude faster comparing with *Spin Image* and *Surface Signature* as reported in [97]. Although without using tangent plane, *snapshot* still inherits the idea of projecting 3D points onto a 2D plane from *Point Signature*, *Spin Image*, *Point's fingerprint* and *Surface Signature*.

Thrift is a local structure representation inheriting the idea from the proven successful 2D image technique SIFT (Scale-Invariant Feature Transform) [89]. Inspired by SIFT using image gradients to describe image patches, *Thrift* employs the surface normals of points as the basis of shape descriptor [45] [46]. The surface normals are approximated by fitting a plane using the points in support based on the least-squares techniques. An advantage of adopting normals to describe the surface is that the normal information is invariant to point density according to the analysis in [45]. Essentially, *Thrift* is a 1D histogram accumulating the cosine function of angle between two normals of the same point in different scale support. From the view of implementation, *Thrift* firstly determines the support of each keypoint p; secondly for every point q in the support compute two normals with regard to two different scale windows around q using the least-squares method; finally builds a 1D histogram based on the cosine function of the angle between the two normals.

Rather than a specific local shape descriptor, the variable-dimensional local shape descriptor (VD-LSD) is more like a generalization representation which could be instantiated into several existing local shape descriptors [139], for example *Spin Image* and *Point Signature*. *VD-LSD* is presented by Taati *et al.* in 2007. *VD-LSD* first builds the covariance matrix of the coordinates of points in support, and solves the covariant matrix by Eigenvalue Decomposition to achieve Eigenvalue scalars and Eigenvalue vectors which are used to construct LRF. Nine different kinds of properties of the point q in support could be calculated based on LRF and the eigenvalue scalars, and a subset of them are used to form the local shape descriptors, in other words, the maximum dimension of *VD-LSD* is nine. The properties contain position properties, direction properties and dispersion properties. An automatic selection of the properties employed by final *VD-LSD* based on the captured geometry and characteristics of the sensing devices is provided later [140]. With large dimensions and poor compactness, *VD-LSD* requires much more memory and runtime. But the time efficiency of the overall scan registration tasks could be improved since *VD-LSD* is highly descriptive, which reduces the feature matching runtime as reported in [139].

Exponential Map is employed by Novatnack and Nishino in 2008 to constitute both scale-dependent and scale-invariant local shape descriptors [98]. For each point q in support region of keypoint, it calculates its geodesic polar coordinate $\{gd(q), \alpha(q)\}$, where $gd(q)$ is the geodesic distance between p and q, and $\alpha(q)$ is the polar angle of tangent of the geodesic between p and q. The final descriptor is a 2D domain indexed by the geodesic polar coordinates storing surface normals of the points in support region. A *geodesic distance-weighted shape vector image* which uniquely maps a 3D surface onto a canonical rectangular domain and encodes the surface shape characteristics to form a two-channel shape vector image is proposed by Hua *et al.* in 2008 [59]. For each channel, a descriptor is generated using the similar technique as SIFT [89]. The final descriptor concatenates the two shape vector images followed by a geodesic distance-weighted inhomogeneous linear diffusion. The aforementioned two local shape descriptors are based on the obscure differential geometry theory, so they have not attracted much attention.



(a) (b)

Fig. 2.9 Illustration of typical shape descriptors (a) Point Feature Histograms (PFH), reprinted from [51] (b) Intrinsic Shape Signature (ISS), reprinted from [172].

The famous histogram-based local shape descriptor *Point Feature Histograms* (PFH) is proposed in 2008 by Rusu *et al.* [122]. For each keypoint p, it estimates the normal of every point in its support region. And for each pair of points in support, it defines a Darboux frame placed at the keypoint and calculates four features inheriting from Surflet-Pair-Relation Histograms [159] based on the directions of Darboux frame, points' normals and distance between these two points. A histogram with 16 bins is built by accumulating these features. PFH involves huge computational load since four features are calculated for every point pair in keypoint's support region, as Figure 2.9a shows. Hence Rusu *et al.* present a Simplified Point Feature Histogram (SPFH) based on only the features of point pairs involving keypoints, and employ neighbouring SPFH values to create the final shape descriptor called *Fast Point Feature Histograms* (FPFH) [119].

Similarly with 3D shape context, *Intrinsic Shape Signatures* (ISS) also places a sphere centred on the kepoint and computes the shape descriptor based on the spatial information of points falling in the spherical bins [172]. But rather than dividing the spherical surface evenly along the azimuth and elevation as 3D shape context, ISS uses a discrete spherical grid to divide the spherical angular space relatively uniformly and homogeneously, as Figure 2.9b depicts. And the radial distances are equally divided. The density weights of the points falling into each bin are accumulated to construct the final ISS descriptor similar to 3D Shape Context. Generally speaking, ISS has much similarity to 3D shape context.

In 2009, two local shape descriptors involving differential geometry, just like Exponential Map and geodesic distance-weighted shape vector image proposed in 2008, are presented: *MeshHOG* (Mesh Histogram of oriented Gradients) [169] [168] and *HKS* (Heat Kernel Signature) [136]. But these two descriptors are much more popular.

MeshHOG requires the gradients of points. So first of all it defines a scalar function f on the points and calculates the gradients $\bigtriangledown f$. The term f could be normal, texture, curvature, density and so on. As aforementioned local shape descriptors,*MeshHOG* also builds a LRF for each keypoint and projects the gradients onto the planes of LRF. For each plane, firstly it partitions the plane into four slices according to the spatial information of gradients, and then divides the spatial slices into eight parts according to the orientation information of gradients. Thus the final *MeshHOG* descriptor is a vector containing 96 $(3 \times 4 \times 8)$ elements.

HKS is proposed by Sun *et al.* considering the advantages of heat kernel, such as stability to noise and invariance to scales of support. And Sun *et al.* restricts the heat kernel in the temporal domain to solve the computation complexity problem, which is a main contribution of [136]. The HKS descriptor could be interpreted as a multi-scale Gaussian curvature. It is stable to perturbations of the shape and invariant to isometric deformations, so it could be used for non-rigid shapes. Further, HKS has several variants [20] [114] [77] [78] [171], which prove its effectiveness and popularity.

As *Thrift*, *2.5D SIFT* is another extension of the famous 2D SIFT [89]. *2.5D SIFT* proposed by Lo and Siebert in 2009 juxtaposes nine histograms, each constitutes two histograms: the histogram of range surface topology types and the histogram of range gradient orientations [88]. From the view of implementation, *2.5D SIFT* firstly acquires the properties of keypoints, including locations, scales, orientations, Gaussian curvatures and so on, which requires large amount of memory and computation; secondly it places nine overlapped elliptical Gaussian weighted subregions on the keypoints limited by the scale of keypoint's support; thirdly for each subregions, two histograms are built and normalized respectively: an eight-element histogram accumulates the gradient orientations and a nine-element histogram stores the surface type based on the shape index values; finally all the histograms are constituted to form the final *2.5D SIFT*. So the final *2.5D SIFT* is a vector containing 153 $(9 \times (8+9))$ elements.

In 2010, another variant of SIFT called *Shape Index SIFT*, is proposed by Bayramoglu and Alatan [10] based on the theorem that combination of SIFT and shape index allows matching surfaces with variable scales and orientations. Its implementation shares much similarity with other 3D SIFT-based algorithms: divide the local patch around keypoint into 16 subregions; for each subregion, generate an 8-element histogram with accumulating the gradient orientations of the shape index; concatenate 16 histograms to form the final descriptor. *Shape Index SIFT* requires less pre-process comparing with 2.5 SIFT and its performance is better than 2.5D SIFT [88] and the shape index-based method in [27][28] as reported [10]. Another SIFT-like 3D local shape descriptor named *LD-SIFT* is proposed in 2012 [36]. one of its contributions is that it uses the PCA of the coordinates of points in support region to obtain the local coordinate system, and then aligns the local coordinate system according to the local dominant angle which is determined by the leading local PCA eigenvector. The following procedure is much similar with other SIFT-like descriptors, and it computes shape descriptors based on the derivatives of the depth image.

3D *SURF* (Speeded Up Robust Feature) is an extension of the 2D SURF [9] and is a Haar-wavelet transform-based local shape descriptor [76]. Different with other local descriptors, 3D SURF contains the relative position of keypoint to the shape's center, since it is initially designed to be gathered up to constitute a global shape descriptor. After converting the 3D point clouds into volumetric images, 3D SURF also defines a LRF of the keypoint based on the Haar-wavelet transform of the volumetric image around keypoint. Achieving the LRF, it defines a $N \times N \times N$ grids around the keypoint and computes a description of the keypoint's support according to the 2D SURF. A description vector contains 6 elements is calculated for each grid cell. The final 3D SURF consists of three parts: a 3D vector containing the relative position of the keypoint to center of shape; a scalar indicating the scale of the descriptor; a $N \times N \times N \times 6$ vector describing the support of keypoint.

Another interesting local shape descriptor proposed in 2010 is *NARF* (Normal Aligned Radial Feature) [130]. From the view of implementation, the achievement of *NARF* contains four steps: firstly estimate the normal of keypoint; secondly calculate a normal aligned range image patch on the keypoint; thirdly overlay a star pattern on the range image patch, just like Figure 2.10a shows, and variation of the pixels under each beam corresponds to an element in a vector; finally shift the vector according the dominant orientation to make the final descriptor to be rotation-invariant.

The recent famous local shape descriptor *SHOT* (Signature of Histograms of Orientations) is first proposed in 2010 by Tombari *et al.* [150]. A main contribution of this paper is the analysis of significance of a unique, repeatable and robust 3D LRF to local shape descriptors' performance. Overall, the calculation of *SHOT* includes four steps: firstly, estimate the

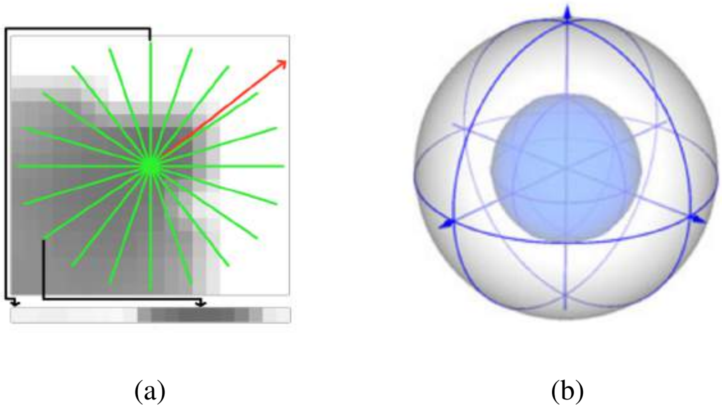


Fig. 2.10 Illustration of typical shape descriptors (a) Normal Aligned Radial Feature (NARF), reprinted from [130] (b) Signatures of Histograms of Orientations (SHOT), reprinted from [150].

normal of every point and build a repeatable and unique LRF for the keypoint; secondly, place an isotropic spherical grid centred on keypoint and divide the keypint's support along the azimuth, elevation and radial direction, as Figure 2.10b sketches; thirdly, for each sub-volume build a local histogram accumulating the point number according to the inner product of point normals; finally the local histograms are gathered up to form the final *SHOT*. Different from 3D Shape Context, the partition of keypoint's spherical support is much coarser for *SHOT*. The authors recommend 32 spherical bins, which are resulted from 8 divisions along azimuth direction, 2 along elevation direction and 2 along radial direction. And the recommended dimension of local histogram is 11, thus the final *SHOT* contains 352 $(8 \times 2 \times 2 \times 11)$ elements. A more detailed description of *SHOT* and its extension to RGB-D data is available recently in [123].

In 2012, a histogram-based local shape descriptor named *HONV* (Histogram of Oriented Normal Vectors) is designed [142]. HONV divides the support of keypoint into non-overlapped partitions; for each partition it builds a 2D local histogram accumulating the number of points according to the azimuth and elevation angles of points normal. The descriptor is formed by concatenating the local histograms. Experiments prove that *HONV* performs better than HOG descriptor in object detection and classification tasks with regard to a RGB-D database.

2.2.3 Global shape descriptors

The famous *Extended Gaussian Image* (EGI) is proposed in 1984 by Horn [56]. Map the surface normal vectors of an object onto a unit sphere by placing their tails at the center of sphere and their heads at the corresponding point on the sphere; in this way, the final mapping is called Gaussian Image of the object. Further, if divide the spherical surface into cells and each cell accumulates the weighted number of the surface normals located in it, a gray-scale image is achieved which is named *Extended Gaussian Image*. The weights of the surface normal could be the area of the surface where the normal is calculated, thus sum of EGI equals the surface area of the object. Rotation of EGI of an object corresponds to the rotation of object, but EGI is invariant to the translation of the object. Based on this property of EGI, it is used for 3D point clouds registration combing the Spherical Harmonic Transform (SHT) in 2006 [96]. This is an early try to use global shape descriptor in scan registration tasks, but two limitations of EGI restricts its performance: firstly, EGI only uniquely defines convex objects, and different non-convex objects could produce the same EGI; secondly, EGI cannot handle the spherical objects where the EGI is constant and less informative. Unfortunately, the experiments in [96] do not involves the scan pairs with large translations; in our opinion, the translation recovery step of this scan registration method is ineffective and requires refinement.

Shape Histogram is a quite primitive and modesty global shape descriptor presented by Ankerst *et al.* in 1999 [4]. It moves the 3D scan to the center of scan and partitions the space into disjoint and complete cells, while each cell accumulates the number of points located in it. Three methods are suggested to divide the space: a shell model, which decomposes the 3D space into concentric shells around the center; a sector model, which partitions the 3D space into sectors that emerge from the center; the combined model, which is a simple combination of the shell model and the sector model which results in a high dimensionality. *Shape Histogram* is a very flexible shape descriptor, and for all the three models, the number of dimensions could be adapted easily to the specific applications. *Shape Histogram* is adopted in 3D object recognition task in [4].

Saupe and Vranić employed the spherical harmonics or moments of *Spherical Extent Function* to describe the 3D models. Achievement of *Spherical Extent Function* contains three steps: firstly normalize the model by a modified PCA; secondly cast several rays from the center of object; finally the object extents (max distance to center) on all rays constitute a function on the sphere, named *Spherical Extent Function*. In order to improve the robustness of *Spherical Extent Function*, the absolute values of spherical harmonic coefficients or moments of *Spherical Extent Function* are stored as the final vector. Furthermore, the experiment results prove that the representation with spherical harmonics performs better than the representation based on moments. The spherical harmonics provides a natural method to generate embedded multi-resolution 3D shape vectors, and could be applied to other functions defined on sphere. However, this spherical harmonics-based representation requires a normalization preprocess of models, while another spherical harmonics-based method solves this problem and provides the rotation-invariant representation directly [70], which will be introduced later.

Shape Distribution is essentially a histogram generated based on the shape function by measuring the geometric properties of 3D objects. Theoretically, any shape functions could be employed to construct *Shape Distribution*, but five shape functions are suggested by [101] considering their simplicity and invariance: **A3** measures the angle between three random points; **D1** measures the distance between the centroid of model and one random point on the surface; **D2** measures the distance between two random points; **D3** measures the square root of the area of triangle determined by three random points; **D4** measures the cube root of volume determined by four random points. The shape function employed by *Shape Distribution* could be selected flexibly according to the particular models and applications. *Shape Distribution* is invariant to rotation and translation, and robust to small perturbations, which means it does not require pose normalization in shape retrieval and object recognition tasks.

A *Reflective Symmetry Descriptor* consisting of measurements of reflective symmetry for the 3D volumetric model with regard to the planes passing through center of the 3D model is proposed in 2002 [68] [69]. Essentially, the descriptor is a 2D spherical function whose values provide a measure of the symmetry of global shape. From the view of implementation, *Reflective Symmetry Descriptor* contains three steps: firstly convert the 3D point clouds into the corresponding volume; secondly, for every plane through the center, compute the symmetry measurement of the volume; thirdly, all the symmetry distances are stored in a 2D array which is indexed by the orientation of planes. The peaks of the descriptor indicate the directions in which the 3D model has strong symmetry. In order to improve its efficiency, the authors combined the 3D Fast Fourier Transform to reduce the runtime and to provide a multi-resolution approximation in less time.

Another statistical global shape descriptor named *Surflet-Pair-Relation Histograms*, similar to Shape Distribution, is proposed in 2003 by Wahl *et al.* [159]. This shape descriptor computes four geometric features based on the oriented points which are named *surflets*. The four geometric features are invariant to translation and rotation, just like the shape functions used by Shape Distribution [101]. In this way, the final shape descriptor is also invariant to rotation and translation. The calculation of four geometric features is same as the PFH and FPFH, or more to the point, the VFH and FPFH inherit the geometric features of *Surflet-Pair-Relation Histograms*. The final descriptor is built by quantizing each geometric feature into N bins, so the final descriptor contains N^4 bins. From the view of implementation, the four geometric features of each surflet pair (p_1, p_2) are calculated as following: firstly discriminate the source point whose normal has smaller angle between the vector $\overrightarrow{p_1 - p_2}$; secondly assuming p_1 is the source point, define a Darboux frame originated on p_1: $u = n_1, v = (p_2 - p_1) \times u, w = u \times v$; thirdly compute the four geometric features based on points' normals and distances: $\alpha = arctan(w \cdot n_2, u \cdot n_2)$, $\beta = v \cdot n_2$, $\gamma = u \cdot \frac{p_2 - p_1}{\|p_2 - p_1\|}$, $\delta = \|p_2 - p_1\|$.

A rotation-invariant representation of 3D shape descriptors is defined on spherical surface, which is called *Spherical Harmonic Descriptor*, in 2003 by Kazhda *et al.* [70]. It could be used with regard to several spherical shape descriptors, such as EGI, Spherical Extent Function, Shape Histogram and volumes. This rotation-invariant representation provides an alternative for the rotation normalization, which is usually achieved by PCA technique but unstable, and could offer multi-resolution representations of the shape descriptors. The key of this rotation-invariant representation is to decompose the spherical functions into different frequencies and store the energies contained in different frequencies. The employed decomposition method is Spherical Harmonic transform, which could be regarded as an extension of Fourier Transform applied to spherical functions. The limitation of this rotation representation is that it loses the information along the radial direction, which means it could not detect the spherical rotation of the spherical representations with different radius.

Novotni and Klein propose a shape descriptor named *3D Zernike* by naturally generalize the spherical harmonics-based shape descriptors. *3D Zernike* achieves the rotation invariance by computing the norms of the decomposition coefficients, quite similar to the algorithm presented in [70]. However, please note that these two similar techniques are published almost simultaneously. From the view of implementation, the building of *3D Zernike* descriptors contains four steps: firstly normalize the translation and scale of the original model; secondly calculate the geometrical moments; thirdly compute 3D Zernike moments based on the geometrical moments; finally define the *3D Zernike* descriptor as the norms of 3D Zernike moments. Since *3D Zernike* descriptor also captures object coherence in the radial direction besides in the direction along the sphere, it could differ independent spherical rotation of different spherical functions while the rotation-invariant spherical harmonics representation could not handle this.

As far as we know, the first view-based shape descriptor of 3D data named *Light Field Descriptor* (LFD) is proposed by Chen *et al.* in 2003 [26]. Placing twenty cameras on the vertices of a regular dodecahedron could achieve 20 silhouette images of the 3D object, but only 10 silhouette images rendered from vertices of dodecahedron over a hemisphere are used by *Light Field Descriptor*. *Light Field Descriptor* is defined based on the 2D features of these 10 silhouette images. The 2D features employed by *Light Field Descriptor* is the combination of region-based Zernike Moment Descriptor and the contour-based Fourier Descriptor, and actually 35 coefficients for Zernike Moment Descriptor and 10 for Fourier Descriptor are adopted. The composite global shape descriptors become a trend immediately with emerging of this first view-based shape descriptor. It is because the view-based technique gives the 3D shape descriptors a chance to use the mature 2D shape descriptors. And benefiting from the burgeoning of the computing power, large quantity of hybrid shape descriptors are used in 3D shape retrieval task recently.

A composite 3D global shape descriptor named *DESIRE* is proposed by Vranić in 2005 [158]. *DESIRE* combines the depth-buffer based features, the silhouette-based features and the ray-extent features. Experiments prove that *DESIRE* outperforms *Light Field Descriptor* in 3D shape retrieval task. From the view of implementation, the achievement of [158] contains five steps: firstly normalize the 3D model using the Continuous Principal Analysis technique; secondly extract depth-buffer based feature vector which consists of magnitudes of 186 Fourier coefficients of six depth-buffer images; thirdly calculate silhouette-based feature vector which is comprised of magnitudes of 150 Fourier coefficients of three silhouette images; fourthly estimate the ray-extent feature vector by storing the magnitude of 136 spherical harmonic coefficients of the spherical extent function; finally concatenate the three basis feature vectors to form the composite *DESIRE* which has 472 $(186+150+136)$ elements.

The descriptor based on *Spherical Trace Transform* is proposed by Zarpals *et al.* in 2007 [170]. The proposed *Spherical Trace Transform* contains tracing the volume generated by the 3D model with two kinds of geometric shapes: a set of 2D planes which are tangential to concentric spheres; a bundle of radius segments. The intersection of 2D plane with 3D volume is 2D image, and the intersection of radius segment with 3D volume is 1D line. And the shape descriptor is generated by applying different kinds of functionals, such as Fourier transform, radial integration transform, Polar wavelet transform, Polar-Fourier transform, to the set of 2D images and 1D lines. This descriptor could also be regarded as a composite descriptor.

Another view-based 3D global shape descriptor named *Multi-Fourier Spectra Descriptor* (MFSD) is proposed in 2009 [144]. The MFSD is composed of four independent Fourier spectra from four different kinds of data types: volume representation and three different kinds of rendered 2D images from 3D model. 3D Fast Fourier Transform is applied to the volume representation and the magnitude of Fourier coefficients are stored. 2D Fast

Fourier Transform is applied respectively to three kinds of rendering 2D images: depth-buffer image, silhouette image and contour image. By the same token, the magnitude of Fourier coefficients are stored. Specifically, 512 dimensions for the volume representation, 1536 dimensions for the depth-buffer image, 768 for the silhouette image and 768 for the contour image are stored. The same authors present another spectra-based shape descriptor named *Dense Voxel Spectrum Descriptor* (DVD) in 2012 [84]. Rather than rendering the 3D model into different kinds of images, DVD describes each piece of volume representation of 3D models by Fourier spectra. From the view of implementation, DVD converts the 3D model into a volume of size $64 \times 64 \times 64$ by assigning distances to the center to the values of voxels. And then slides the volume by a $32 \times 32 \times 32$ block with step 16, for each block compute the 3D Fast Fourier Transform and stores the magnitudes of the Fourier coefficients of 8 low frequencies. The final shape descriptor DVD consists of concatenated magnitudes of Fourier spectra. The total dimension of DVD is supposed to be $((64-32)/16+1)^3 * 8^3 = 13,824$, but the PCA technique is applied to reduce the dimension to $((64-32)/16+1)^3 * 20 = 540$.

Compact Multi-View Descriptor (CMVD) proposed in 2010 also belongs to the category of the view-based global shape descriptor [35]. 2D image are captured by the cameras placed at the 18 vertices of the 32-hedron. Similarly with MFSD, CMVD uses difference kinds of rendering images: silhouette images and depth images. For each image, three rotation-invariant 2D feature vectors are generated by applying three different kinds of transforms: 2D Polar-Fourier transform, 2D Zernike Moments and 2D Krawtchouk Moments. In order to improve the compactness of the final descriptor, the low frequencies of these three 2D transforms are used to form the CMVD while the high frequencies are discarded. Actually, the final dimensions of the descriptor for each view is 212 $(78+78+56)$.

The 3D global shape descriptor named *PANORAMA* uses a set of panoramic views of 3D model to describe the model [103]. From the view of implementation, the calculation of *PANORAMA* contains four steps: firstly normalize the pose of the 3D model based on PCA techniques; secondly place three cylinder centred at the centroid of object and paralleled to the three principal axes, and obtain three panoramic views by projecting the 3D model onto the lateral surfaces of cylinders; thirdly apply 2D Discrete Fourier Transform and 2D Discrete Wavelet Transform to the panoramic views respectively; finally rather than simply concatenating 2D features, the features generated from each panoramic view are weighted according to the direction of cylinders to form the final 3D shape descriptor.

Just like the shape descriptors proposed in [100] [52], the view-based 3D shape descriptor *DB-VLAT* (Depth-Buffered Vector of Locally Aggregated Tensors) [145] involves the Bag-of-Visual Words (BoVW) technique. *DB-VLAT* first renders 18 depth-buffer images with the size of 256×256 from the 3D object. It extracts the local features from all 2D depth-buffer images using Dense SIFT algorithm, and applies clustering techniques to build the vocabulary of Visual Words. Then for each local feature, find its nearest feature in vocabulary of Visual Words and construct a histogram by counting how many times the visual words appear in an image, thus this histogram is regarded as the feature vector of the image. As other view-based shape descriptors, the pose normalization of 3D model is necessary for *DB-VLAT*.

2.3 Fourier Transform applied in scan registration

2.3.1 Basic principles of Fourier Transform

Although scan registration algorithms based on Fourier analysis techniques are not widely and successfully used so far, we review this kind of techniques here, since we recover the translations between two 3D scans based on Fourier analysis techniques.

Fourier analysis techniques are extremely significant in signal processing and pattern recognition, since they decompose the function into a linear combination of sinusoidal basis functions. Each of these basis functions is a complex exponential of a different frequency. In other words, the Fourier analysis techniques map a function into a set of coefficients of basis functions, and the coefficient of the basis function with frequency f gives how much power the function contains at the frequency f. That is, the Fourier analysis techniques give us another way to represent a waveform. Admittedly, there are infinite ways to decompose the signals. But the goal of decomposition is to get something easier to deal with than the original signals, and the reason why sinusoids are adopted is that they are the eigenfunctions of the Laplacian operator, hence they maintain fidelity to most real systems. The basis functions of normal Fourier analysis techniques are induced by the Laplacian operator in Cartesian coordinate system. By the same token, the Laplacian operator also has effective forms in other coordinate systems, e.g. polar and spherical coordinate system. The Polar/Spherical Fourier analysis techniques are connected with Cartesian Fourier analysis techniques by the ***Laplacian operator***.

There are four types of Fourier analysis techniques: Fourier Series (FS), Fourier Transform (FT), Discrete-time Fourier Transform (DTFT) and Discrete Fourier Transform (DFT). The FS breaks down a periodic continuous function into the sum of infinite sinusoidal functions. The FT extend the idea of FS to continuous aperiodic functions. The DTFT is the spectral representation for aperiodic discrete signals, and normally the discrete inputs are acquired by digitally sampling the continuous function. It is interesting that the DTFT frequency representation is always a periodic function. So though the result of DTFT is an infinite summation, sometimes it is convenient to regard the DTFT as a transform to a 'finite' frequency representation (the length of one period). The DFT converts a finite list of equally spaced samples of a function into the list of coefficients of a finite combination of complex sinusoids, ordered by their frequencies. Since digital computer can only work with discrete and finite signals, the only type of Fourier analysis technique could be used in computer software is DFT. Please note that the Fourier Transform is usually used as the generic term of the Fourier analysis techniques in the literatures.

The sequence of N complex numbers $x_0, x_1, \ldots, x_{N-1}$ is transformed into an N-periodic sequence of complex numbers $X_0, X_1, \ldots, X_{N-1}$ according to the DFT formula:

$$X_k = \sum_{n=0}^{N-1} x_n \cdot e^{-i2\pi kn/N} \tag{2.1}$$

Equation 2.1 could be interpreted as the cross correlation of the input sequence, x_n, and a complex exponential at frequency k/N. Thus it acts like a *matched filter*, and X_k is the Fourier coefficient with that frequency, which represents how much power contained in the original sequence at that frequency. Based on the coefficients, the original complex data x_n could be expressed as:

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k \cdot e^{i2\pi kn/N} \tag{2.2}$$

The kernel of Fourier analysis technique is the representation of data in another linear space. The Equation 2.2 could be regarded as the representation of the original sequence in frequency domain.

2.3.2 Data registration based on Fourier analysis

According to the Fourier Rotation Theorem, the estimation of rotation matrix between two 3D scans could be converted to the determination of the rotation information between the magnitudes of the corresponding 3D Fourier spectra.

The early try of registration of translated and rotated images using Fourier analysis techniques is presented in [38]. The idea of algorithm in [38] is quite naive. Since there is only phase drift between the Fourier spectra of datasets if they are pure translational duplicate version to each other, the authors define a function that is the quotient of the Fourier spectra and try all tentative values of rotation angle. The true rotation angle approaches the quotient to the exponential form. In other words, the fully automatic directed search strategy is used to determinate the rotation angle. Obviously this method is computational expensive. Further, the authors of [91] point out that the method rests on the observation that if the rotation angle is rather small (not exceeding ±2 degree), the peak of the quotient may be still spotted out although considerably lower than that associated with a pure translation. This represents a serious problem and makes such technique suitable only for applications where rotations are small and certainly not for general purpose utilizations. Besides, because the rotation angle is applied to magnitude of Fourier spectra not the original data, and the phase drift relates to both translation and rotation, the phase drift caused by rotation could not be compensated, which also reduces its robustness.

More official algorithms of data registration based on Fourier spectrum are proposed in [29][115]. The essential ideas in these two paper are quite similar. They adopt the well known invariant function named *Fourier-Mellin transform*, which is translation-invariant and represents rotation and scale as translations along the corresponding axes in homologous parameter spaces. The Fourier-Mellin Invariant (FMI) descriptor of an image, named Fourier log-magnitude spectra in [115], is obtained by resampling the spectral magnitude of this image to polar coordinates and then resampling along radial coordinate with a logarithmic function. And the FMI descriptor could also be achieved by re-sampling the spectral magnitude directly onto a rectangular polar-logarithmic coordinate in one step, like what is done in [29][115]. In the polar-logarithmic representation of the spectral magnitude, both the rotation and scale are transformed to translation. Actually the FMI descriptor is not firstly proposed in [29], but the previous methods match the FMI descriptors using cross-correlation, or variants of cross-correlation. Since the FMI descriptor is based on the magnitude of Fourier transform, the cross-correlation of the FMI descriptors generally yields a very broad maximum, which leads to the FMI descriptor-based registration methods are unreliable and may give rise to wrong estimates of correlation peaks. The Symmetric Phase-Only Matched Filtering (SPOMF) is introduced to match the FMI descriptors in [29], and this is the main difference from the algorithm in [115]. In this way, the advantages of the SPOMF, which are sharpness of the correlation peaks and robustness in the presence of noise, and the decoupling the rotation, scale and translation acquired by the FMI descriptor are combined together. The problem here is that the limited scale range can be estimated since large scales would alter the frequency beyond recognition. It should be noted that the maximum scale recovered by [29] is 2.0 and the maximum scale recovered by [115] is

1.8. Besides, since the re-sampling of Cartesian frequency values on a polar grid is very sensitive to interpolation, the accuracy of registration algorithms is severely degraded by the approximation errors inherited in the computation of the polar and log-polar Fourier spectra.

A novel frequency domain technique which works in Cartesian coordinates and bypasses the need to transform data from the Cartesian to the polar domain is presented in [91][92][90]. It is an important advance because it is well known that the Cartesian-to-polar coordinate transformation is a numerically sensitive operation, especially when it is dependent on the interpolation of the Fourier spectrum. The fundamental idea of this novel method rests on the properties of the Fourier transform magnitudes of the images: for two roto-translated images, the difference between the Fourier transform magnitude of one image and the mirrored replica of the Fourier transform magnitude of the other have a pair of orthogonal zero-crossing lines. These two lines are rotated with respect to the frequency axes by an angle that is half the rotational angle. In [91], the difference function of normalized Fourier transform magnitudes is defined, and the zero-crossing lines of the difference function are used to determine the rotation angle. Therefore, the estimate of the rotation angle is transformed to the detection of two zero-crossing lines. Subsequently, the phase correlation technique is applied to solve the translation parameters.

The research direction of determining rotation angle based on the difference function of normalized Fourier transform magnitudes is extended to the case of 3D rigid motion in [93][94]. This kind of 3D data registration method contains three procedures: rotation axis determination, estimate of the rotation angle around the axis and translation calculation. The rotation axis is determined in [94] by searching a minimum of radial projection of the polar resampled differences of the magnitude of Fourier spectra. After obtaining the rotation axis, the rotation angle estimation is a planar rotation problem, which is solved by a combination of the a 1D Fourier transform and a 2D polar Fourier transform defined as a cylindrical Fourier transform. This algorithm involves the interpolation operation in the resampling of the difference function in rotation axis estimating step, which increases the uncertainty of the registration result. Another main drawback is that this method requires the common region between two input data to be known, which means this method could not handle the general partial overlapped datasets and that constraints its application.

The methods presented in [72][73] inherit the three-step framework, but applied the 3D Pseudo-polar Fast Fourier Transform (FFT) to avoid the interpolation operation. The Pseudo-polar FFT is firstly proposed in [6]. It could be used to compute the Discrete Fourier Transform (DFT) on pseudo-polar grids without interpolation of Cartesian Fourier spectra. The Pseudo-polar FFT is based on the Fractional Fourier Transform (FrFT) [7] [2], which is a generalization of the conventional Fourier transform. The FrFT depends on a parameters α

and can be interpreted as a rotation by an angle α in the time-frequency plane with respect to the conventional Fourier transform. An FrFT with $\alpha = \pi/2$ corresponds to the conventional Fourier transform. Essentially, the α-order FrFT shares the same eigenfunctions as the conventional Fourier transform, but its eigenvalues are the αth power of the eigenvalues of the conventional Fourier transform. Concretely, the FrFT samples the spectrum of a vector with length N at the frequencies: $\omega_k = \alpha l/N, l = -N/2, \ldots, N/2$. In other words, the FrFT could compute the Fourier spectra with arbitrary frequency resolution. Furthermore, the FrFT of a vector of length N can be computed in $O(NlogN)$ operations for any α.

The most recent registration method based on Cartesian Fourier spectra is presented in [22][21], named Spectral Registration with Multilayer Resampling (SRMR). The SRMR re-samples the spectral magnitude of 3D FFT calculated on discrete Cartesian grids of the 3D data to decouple the 3D rotation and 3D translation, just like the previous techniques. Further, the SRMR also tries to transform the rotation parameters to the translation estimation problem and adopt the phase correlation techniques to figure it out, and this main idea is also inherited from the previous techniques. The most remarkable feature of SRMR is that it uses the spectral structure at a complete stack of layers instead of only one spherical layer. And it does not rely on finding minima indicating the main rotation axes, which makes it extremely robust for the partial overlapped datasets. But the strong point is bought at the cost of working only in a limited range of roll and pitch offsets between input datasets. So the SRMR is only applicable in robotic mapping scenarios, where there is little roll and pitch changes. Furthermore, please note that the SRMR also suffers from the sensitive interpolation of spectral magnitudes.

2.4 Public resources

There are large amount of public available resources related to 3D data, and limited by space, we only list some representatives, which are well organized and widely used.

In Table 2.1, we list 11 free available datasets, which contain data of reconstructed 3D models and 2.5D range images. We list 11 free available software in Table 2.2, which include some shape descriptors, visualization of 3D data and so on. The 12th item of Table 2.2 lists several software widely used in computer graphics community.

Table 2.1 Representatives of FREE public available 3D datasets.

No	Name and Reference	Affiliation	URL
1	Stanford 3D Scanning Repository [34]	Stanford Computer Graphics Laboratory	http://graphics.stanford.edu/data/3Dscanrep/
2	Princeton Shape Benchmark [127]	Princeton Shape Retrieval and Analysis Group	http://shape.cs.princeton.edu/benchmark/
3	Robotic 3D Scan Repository	Jacobs University	http://kos.informatik.uni-osnabrueck.de/3Dscans/
4	Queen's Range Image and 3-D Model Database	Queen's University	http://rcvlab.ece.queensu.ca/~qridb/QR3D/index.html
5	RGB-D Object Dataset [79]	Washington University	http://rgbd-dataset.cs.washington.edu/index.html
6	Large Geometric Models Archive	Georgia Institute of Technology	http://www.cc.gatech.edu/projects/large_models/
7	Laser Registration Datasets [112]	Autonomous Systems Lab in ETH	http://www.asl.ethz.ch/research/index
8	Toyohashi Shape Benchmark [146]	Toyohashi University of Technology	http://www.kde.cs.tut.ac.jp/benchmark/tsb/
9	Canadian Planetary Emulation Terrain 3D Mapping Dataset [152]	ASRL at University of Toronto	http://asrl.utias.utoronto.ca/datasets/3dmap/#References
10	USF Range Image Database	University of South Florida	http://marathon.csee.usf.edu/range/DataBase.html
11	Ford Campus Vision and Lidar Dataset [102]	PeRL at University of Michigan	http://robots.engin.umich.edu/SoftwareData/Ford

Table 2.2 Representatives of FREE public available software related to 3D point clouds.

No	Name and Reference	Developers	URL
1	PCL[121]	Open Perception, Inc.	http://pointclouds.org/
2	SLAM6D [14]	Andreas Nüchter *et al.*	https://svn.openslam.org/data/svn/slam6d
3	CloudCompare	Girardeau-Montaut Daniel	http://www.danielgm.net/cc/
4	ParaView	Kitware, Inc.	http://www.paraview.org/
5	MeshLab	National Research Council (Italy)	http://meshlab.sourceforge.net/
6	Rotation Invariant Shape Descriptors [70]	Michael Kazhdan *et al.*	http://www.cs.jhu.edu/~misha/Code/ShapeSPH/ShapeDescriptor/
7	Shape Symmetry Descriptors [68]	Michael Kazhdan *et al.*	http://www.cs.jhu.edu/~misha/Code/ShapeSPH/ShapeSymmetry/
8	3D Shape Descriptors	Chris Tralie	http://www.ctralie.com/Teaching/ShapeMatching/#egi
9	LightField Descriptors [26]	Ding-Yun Chen *et al.*	http://3d.csie.ntu.edu.tw/~dynamic/3DRetrieval/index.html
10	3DGSS: 3D Geometric Scale Space [98]	Ko Nishino *et al.*	https://www.cs.drexel.edu/~kon/3DGSS/
11	3D Zernike Descriptors [99]	Marcin Novotni and Reinhard Klein	http://cg.cs.uni-bonn.de/project-pages/3dsearch/downloads.html
12	Computer Graphics Research Software (collection)	Ke-Sen Huang	http://www.dgp.toronto.edu/~rms/links.html

CHAPTER 3

Hough Transform Descriptor

By three methods we may learn wisdom: First, by reflection, which is noblest; second, by imitation, which is easiest; and third by experience, which is the bitterest.

Confucius

Contents

3.1 Background

HOUGH Transform is a voting procedure in a parameter space. The object candidates are obtained as local maxima in the so-called accumulator space. Hough Transform is an extensively accepted method for detecting parametrized objects in image analysis, computer vision and digital image processing [62][132]. Furthermore Hough Transform is also used to detect planes in 3D data [16].

In this chapter, Hough Transform is used in 3D scan registration. Data registration is the process of spatially aligning two or more dataset of an object or scene. The task of data registration is to find an optimal geometric transformation between corresponding datasets. As to the 3D rigid case, the optimal geometric transformation has 6 Degrees of Freedom (DOF): 3 for rotation and 3 for translation. So the 3D data registration could be interpreted as a black box which takes two or more datasets as input and outputs six parameters describing the geometric transformation between the datasets. The 3D scan registration is the cornerstone of Mobile Robotic Mapping, and constructing maps is one of the fundamental issues in building truly autonomous mobile robots. That is because maps are essential to the subsequent tasks, such as path planning and robot navigation.

In the context of 3D rigid registration, scan alignment methods could be classified into *local* alignment methods and *global* alignment methods. If an initial estimate of transformation between two input scans is available, the registration problem could be solved using *local* methods through an iterative process. The most famous local methods are Iterative Closest Point (ICP) [30][12][111][118] and Three-Dimensional Normal Distributions Transform (3D-NDT) [141][95][60]. The local methods could cope with the scan pairs with good initial estimates well. Moreover, they are usually adopted as the refinement of global alignment methods, hence local methods are regarded as the cornerstone of the 3D point clouds registration. In addition to the requirements of initial guesses, there are two other major drawbacks of local methods: firstly, they are easy to get trapped in local minima; secondly, their runtime vary dramatically to different scan pairs.

Considering the shortcomings of local methods, *global* methods take the overall appearances of scans into account and do not depend on initial estimates. The majority of existing global registration methods are dependent on specific features. The feature-based strategies make use of explicit feature correspondences in the environment and could deal with scan pairs with partial overlap and large offsets. The common procedure of feature-based registration methods includes: key-point extraction, feature description, feature matching, transformation estimation and refinement. Various features have been adopted for 3D scan registration, such as Spin Image [66], Shape Context (SC) descriptor [51], Fast Point Feature Histograms (FPFH) [119], Depth-Interpolated Local Image Features (DIFT) [3], Signature of Histograms of Orientation (SHOT) [150] and Normal Aligned Radial Feature (NARF) [130]. The feature-based registration methods confront challenges about how to eliminate the mismatches, especially in large scale scenes with numerous similar 3D patches. That is why the feature-based registration algorithms often fail in outdoor mobile robotic mapping applications. Furthermore, the performance of the feature-based algorithms are closely related to some key parameters. In other words, the parameters used in feature-based methods should be selected carefully through numerous trials or effective parameter learning based on a large amount of data. What is more frustrating is that the parameters perform well in one scene are highly possible to fall down in other scenes.

The plane-based registration algorithms [105][163][164] are accepted extensively in robotic mapping scenarios, since it could partially overcome the mismatch issue. The planar segments could be extracted through region growing algorithms or the Hough Transform of scans. However, the plane-based registration algorithms induce another limitation. They require three planar segments with different directions hence only work in plane-rich scenes. This limitation constraints the plane-based methods to work merely in the urban area and fall down in plane-less scenarios.

Recently, most research attention involved with scan registration is paid to developing *global feature-less* registration algorithms. Generally, the original 3D data is depicted by a global descriptor, and the transformation between the original data could be solved by matching the corresponding global descriptors. In [25], the Rough Transform is used to describe the original 3D data and decouple the 3D rotation information from 3D translation. While its drawback is that it produces several candidate solutions which cost huge amount of computation, and requires an extra validation step to pick out the true transformation. Besides that, the translation determination in [25] is dependent on several peaks of Rough Transform and prone to fall down. The Spectral Registration with Multilayer Resampling (SRMR) algorithm, represented in [21], resamples the spectral magnitude of 3D Fast Fourier Transform (FFT) calculated on discrete Cartesian grids of the 3D range data. In this way, the 3D rotation is detached from 3D translation, and then the registration problem could be worked out based on the cross correlation techniques. But this strategy validates only within a restricted range of roll and pitch offsets between scans, which is common for robotic mapping scenarios but not for many other applications.

This chapter presents a *global feature-less* scan registration strategy based on the Hough Transform Descriptor (HTD) of 3D point cloud. Actually, our algorithm is partially inspired by the plane-based registration methods which use the Hough Transform for planes extraction and Hough Transform-based feature-less registration method HSM3D [25]. The scan registration technique presented in this thesis projects the scan data into Hough domain as well as the HSM3D; in this way, the 3D rotation is decoupled from 3D translation. With regard to rotation recovering, the rotation between two scans could be solved by Generalized Convolution Theorem or the novel method proposed by this thesis. After the rotation information is recovered, the translation is quite easy to determine by Phase Only Matched Filter (POMF).

The technique presented in this paper has the following characteristics:

- ***partially overlapped:*** the convolution theorem based on Fourier Transform is adopted to effectively deal with partially overlapped scan pairs.
- ***less dependent on parameters:*** compared to feature-based registration methods, this strategy is less dependent on the proper parameters.
- ***noise-immune:*** this technique could resist the noise effectively, since it uses the overall appearance of scans rather than specific features.

3.2 Overview of Hough Transform

Hough Transform is a feature extraction method which could find imperfect instances of objects within noisy data by a voting procedure. This voting procedure is carried out in a predefined parameter space. The predefined parameter space corresponds to the parametrized object which will be detected. The motivation behind the Hough Transform is that each input measurement indicates its contribution to a globally consistent solution. This consistent solution indicates the position and orientation of the detected objects. If a particular object is present in the input data, the mapping of all of its points into the parameter space should cluster around the peaks which correspond to instances of that object. In other words, Hough Transform maps distributed and disjoint input measurements into a localized accumulation point in the parameter space. Hough Transform is an excellent feature detection method since it could detect the occluded objects on the evidence of their visible parts. Meanwhile, it ignores the local details of the objects, such as adjacency and disconnection. That means Hough Transform does not use the local, neighbouring-pixel information to accumulate evidence for a parametrized object. Therefore, it is fairly stable to random sampling of the input data. This is the advantage of Hough Transform compared with other vision algorithms which are computationally expensive.

The classical Hough Transform is concerned with the identification of lines in 2D image, but later the Hough Transform is extended to identify positions of arbitrary shapes, most commonly circles or ellipses. Hough Transform is originally invented for machine analysis of bubble chamber photographs in 1959 [58] and later patented in 1962 [23] by the U.S. Atomic Energy Commission. In this patent, the straight lines are parametrized by slope and intercept: $y = ax + b$, where the parameter m is the slope of the line and b is the intercept. It is awkward that the slope-intercept parametrization may lead to an unbounded transform space because the slope could be infinite. The universally used rho-theta parametrization is initially proposed in [42]: $r = x\cos\theta + y\sin\theta$, where the parameter r represents the algebraic distance between the origin and line and θ is related to the orientation of line. Although the early research about Hough Transform describe the techniques to find straight lines, a similar transform can be used for finding any other objects which can be represented by a set of parameters. In particular, [61] reformulates the circle finding problem to two stages. The first stage involves a two-parameters Hough Transform to find the circle center. The direction of each point is employed. The second stage identifies the radius of the circle by a simple histogram, which could be regarded as a one dimensional Hough Transform. A comparative study of circle detection algorithms based on Hough Transform is presented in [166] [167]. Computational load of the Hough Transform-based detection techniques increases rapidly with the number of parameters which define the object. Lines have two parameters, circles have three, and ellipses have five. Hough Transform applied to detect ellipse is probably at its upper limit of practicality. [82] introduces the double transform as an interesting method of reducing the parameter space for ellipse detection.

The classical Hough Transform aims to find out the location and orientation of objects which could be defined by an analytic function. And its capacity is limited by the number of parameters in terms of the object which will be detected. The early effort to enable Hough Transform to be used to detect an non-analytic object is proposed in [8] and named Generalized Hough Transform (GHT). GHT is the modification of the Hough Transform using the principle of template matching. GHT maps orientation of an edge point to a reference point of the object. Every point votes its corresponding reference point. The maxima of the Hough space indicates possible reference point of the object. The details of GHT is proposed in Section 27 where GHT is adopted for correspondence grouping to recognize 3D object.

Hough Transform for the detection of 3D objects is also feasible, and the extension of Hough Transform to detect 3D plane is straightforward. Similar to the slope-intercept formulation of 2D line, a plane could be represented by equation: $z = ax + by + c$. This extension also encounters embarrassment when the planar direction becomes vertical. [157]

uses this formulation to detect planes in the 3D point clouds acquired by laser scanner since all planes in that domain are nearly horizontal. The plane parametrization by its normal vector and one point on this plane is more general. In this way, each point of the input data votes for a sinusoidal surface in the 3D Hough space, and the intersection of these sinusoidal surfaces indicates presence of a plane [126]. The survey of 3D Hough Transform for plane detection in point clouds could be found in [16]. Hough Transform is also applied to find cylindrical objects in point clouds [113], which contains two steps: first finds the orientation of the cylinder and then estimates its position and radius.

3.3 Hough Transform Descriptor (HTD) of point clouds

Hough Transform is an extensively accepted method for detecting parametrized objects, and is widely used in plane-based registration methods. But the plane extraction is inevitable error-prone and the final goal is scan registration rather than plane detection, we are inspired to use Hough Transform to align the scans directly by regarding the Hough Transform of the scan as a global descriptor.

The 3D Hough Transform maps the point cloud into Hough Space which is defined by (θ, ϕ, ρ), such that each point in the Hough Space corresponds to one plane in $\mathbb{R}^3$. In addition, θ stands for the angle between the normal vector of the plane and xy plane, ϕ is the angle between the projection of the normal vector on the xy-plane and x axis, and ρ represents the distance of the plane to origin. Although the normal Hough Transform method takes pixel/voxel images as the input, it is not a requisite and a set of unorganized points in $\mathbb{R}^3$ could also be used as the input of Hough Transform.

Following the trend of HSM3D [25], we use Hough Transform to describe the original 3D point clouds. Although Andrea Censi and Stefano Carpin also try to describe the 3D point clouds using Hough Transform [25], the Hough Transform Descriptor based on oriented points proposed in this thesis is totally different.

3.3.1 Hough Transform Descriptor (HTD)

We use the Hesse normal form of planes [15], which uses the normal vector and a point on the plane to define the plane:

$$\vec{n} \cdot \vec{p} = \rho \tag{3.1}$$

$$\vec{n} = [\cos\theta\cos\phi, \cos\theta\sin\phi, \sin\theta] \tag{3.2}$$

Algorithm 1: Compute Hough Transform Descriptor (HTD) of 3D point clouds

```
Input  : Point cloud S contains $n$ points, size of Hough Transform Descriptor (HTD): $\theta_{num}$,
         $\phi_{num}$, $\rho_{num}$
Output : HTD of S: 3D array whose size is $\theta_{num} \times \phi_{num} \times \rho_{num}$
         TID of S: 2D array whose size is $\theta_{num} \times \phi_{num}$

▷ initialize HTD and TID
HTD$(\theta_{num}, \phi_{num}, \rho_{num}) \leftarrow 0$ ;
TID$(\theta_{num}, \phi_{num}) \leftarrow 0$ ;
▷ discretize $\theta, \phi, \rho$
$\theta_{space} = \frac{\pi}{\theta_{num}}$ ;
$\phi_{space} = \frac{2*\pi}{\phi_{num}}$ ;
$\rho_{space} = (distance_{max} - distance_{min})/\rho_{num}$;
$\theta(\theta_{num}) \leftarrow 0 : \theta_{space} : \pi$ ;
$\phi(\phi_{num}) \leftarrow 0 : \phi_{space} : 2 * \pi$ ;
▷ calculate HTD
for Point $P_i$ in point cloud S: $i \leftarrow 1$ to $n$ do
    for $\theta_m : m \leftarrow 0$ to $\theta_{num}$ do
        for $\phi_k : k \leftarrow 0$ to $\phi_{num}$ do
            $\rho = cos(\theta_m) * cos(\phi_k) * P_i^x + cos(\theta_m) * sin(\phi_k) * P_i^y + sin(\theta_m) * P_i^z$;
            $\rho_i = round((\rho - distance_{min})/\rho_{space})$;
            HTD$(m, k, \rho_i) + +$;
        end
    end
end
▷ calculate TID
for $\theta_m : m \leftarrow 0$ to $\theta_{num}$ do
    for $\phi_k : k \leftarrow 0$ to $\phi_{num}$ do
        $\vec{\rho}$ = HTD$(m, k, :)$;
        TID$(m, n) = sum(\vec{\rho}^2)$;
    end
end
```

Given a point p in Cartesian coordinates, we have to find all planes the point lies on, i.e. (θ, ϕ, ρ) that satisfy equation 3.1. One instance of (θ, ϕ, ρ) corresponds to one point in the Hough Space. Actually all (θ, ϕ, ρ) satisfy equation 3.1 form a 3D sinusoid plane in Hough Space.

The Hough Transform descriptor (HTD) of the 3D point clouds could be defined as follows:

$$HTD(\theta, \phi, \rho) = \sum_{i=1}^{N} \delta(< p_i, \vec{n} > -\rho) \tag{3.3}$$

where p_i is the i_{th} point and N is the number of points. In practical implementation, the Hough Space is divided into discrete cells, and the accumulators related to cells increase with respect to the scores computed by equation 3.3. For each point p_i, all the cells getting in touch with its Hough Transform should increase.

The HTD has two significant properties which qualify it to be a global descriptor for scan registration. Firstly, the rigid transformation between two scans corresponds to the transformation of their HTDs, as equation 3.4 shows, where$R \in SO(3)$, $t \in \mathbb{R}^3$ and $\mathbf{HTD}\mid_X$ means the HTD of scan X. Note that the translation between HTDs is dependent on direction, which means $\hat{t} \neq t$. This property converts the scan registration problem to determining the transformation between their corresponding HTDs.

$$\mathbf{HTD}\mid_{(R\bullet S+t)} = R \bullet \mathbf{HTD}\mid_S + \hat{t} \tag{3.4}$$

Secondly, the (θ, ϕ) parameters of HTD are related to the rotation of the scan, while the ρ parameter is correlated with the translation. In other words, the HTD of the rotated duplication of a scan maintains the ρ invariable, while the HTD of the translated duplication leaves (θ, ϕ) alone. This property could be used to decouple the 6DOF transformation into 3DOF rotation and 3DOF translation.

Translation-invariant Descriptor (TID)

Integrating the $\|\bullet\|_2$ of HTD with regard to ρ, the HTD could be mapped onto a spherical surface $\mathbb{S}^2$ and then obtain the translation-invariant version of HTD. Denote the translation-invariant Hough Transform Descriptor as TID, then

$$\mathbf{TID}(\bar{\theta}, \bar{\phi}) = \sum_{\rho} \|\mathbf{HTD}(\bar{\theta}, \bar{\phi}, \rho)\|_2; \;\; (\bar{\theta}, \bar{\phi}) \in \mathbb{S}^2 \tag{3.5}$$

$$\mathbf{TID}\mid_{(R\bullet S+t)} = R \bullet \mathbf{TID}\mid_S \tag{3.6}$$

In this way, the rotational alignment could be achieved in the TID domain regardless of the translation. It should attract attention that the variation of ρ between HTDs of the translated duplicates is related to the (θ, ϕ) parameters, as equation 3.4 depicts, which means it is impossible to recover the 3DOF translation first without the values of (θ, ϕ). In our HTD-based scan registration strategy, the rotation is first determined in the TID domain regardless of ρ, and then the 3D translation is solved based on the determined rotation matrix.

The algorithm about achieving HTD of 3D point clouds is outlined in Algorithm 1.

Algorithm 2: Compute Hough Transform Descriptor (HTD)

```
Input  : Point cloud S contains $n$ points,
         size of Hough Transform Descriptor (HTD): $\theta_{num}$, $\phi_{num}$
Output : HTD of S: 2D array whose size is $\theta_{num} \times \phi_{num}$

▷ initialize HTD
HTD$(\theta_{num}, \phi_{num}) \leftarrow 0$ ;
▷ discretize $\theta, \phi, \rho$
$\theta_{space} = \frac{\pi}{\theta_{num}}$ ;
$\theta(\theta_{num}) \leftarrow 0 : \theta_{space} : \pi$ ;
$\phi(\phi_{num}) \leftarrow 0 : \phi_{space} : 2 * \pi$ ;
▷ calculate HTD
for Point $P_i$ in point cloud S: $i \leftarrow 1$ to $n$ do
    estimate the normal vector of $P_i$: $\theta_i, \phi_i$ ;
    $m = round(\theta_i / \theta_{space})$;
    $k = round(\phi_i / \phi_{space})$;
    $\rho_i = (P_i^x)^2 + (P_i^y)^2 + (P_i^z)^2$;
    TID$(m,k)$ = TID$(m,k) + \rho_i$ ;
end
```

HTD of oriented points

Please note that in equation 3.3, $\vec{n}$ has not relation with p_i. That means that for each point p_i, it should dot product every $\vec{n}$ determined by each (θ, ϕ) in Hough Space to obtain the potential ρ. And if the result ρ together with (θ, ϕ) corresponds to a cell in Hough Space, then the cell should be increased by one.

However, if normal vectors of points can be calculated in advance, they can be used to speed up the estimation of HTD and increase the reliability. That is because the position of a point in object space, together with its normal vector, completely defines a plane. In this way, the point in scan could be directly mapped to a single cell in Hough Space. Moreover, rather than summing the HTD along ρ, it is possible to map the points onto a two-dimensional (θ, ϕ) Hough Space to achieve translation-invariant descriptor directly.

The algorithm about achieving HTD of 3D oriented points is outlined in Algorithm 2. As to the normal estimation of points, we adopt Point Cloud Library [107] to calculate the surface normals. The Point Cloud Library (PCL) is a standalone, large scale, open project for 2D/3D image and point cloud processing. The solution for estimating normals in PCL is converted to an analysis of the eigenvectors and eigenvalues of a covariance matrix created from the nearest neighbours of the query point. But there is no mathematical way to solve for the sign of the normal, the orientation via Principal Component Analysis (PCA) [67] is ambiguous, and not consistently oriented over the entire point cloud dataset. The solution

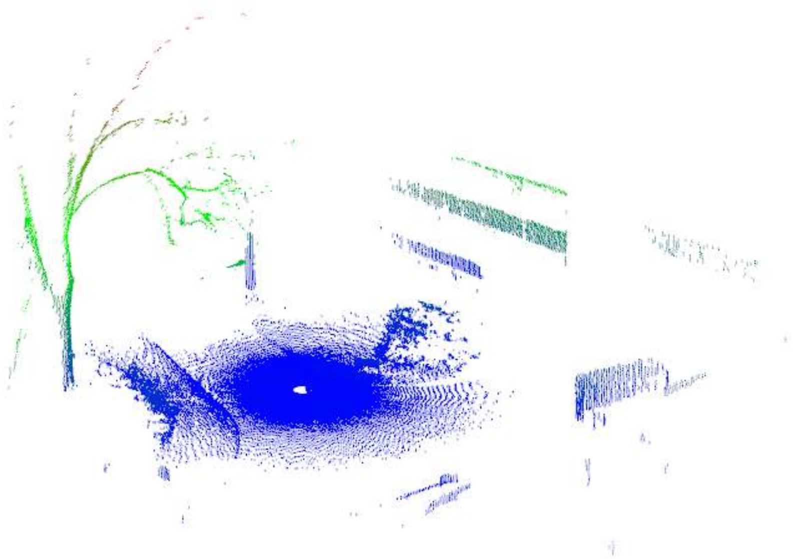


Fig. 3.1 An example scan showing the inhomogeneity of point clouds

to this problem is trivial if the viewpoint V_p is known. To orient all normals $\vec{n}_i$ consistently towards the viewpoint, they need to satisfy $\vec{n}_i \cdot (v_p - p_i) > 0$. The normal estimation in PCL adopt this principle to determine the sign of normal. But when the dataset has multiple acquisition viewpoints, this sign determination method does not work. Actually, the sign disambiguity method in [19] could handle this situation, but PCL has not implemented this algorithm so far.

3.3.2 Discussion

There are several implementation details of HTDs worthy to discuss, including how to design the accumulator of the Hough Space, how to improve the efficiency and so on.

Accumulator design of Hough Transform

The first accumulator design problem is the discretization of the parameters (θ, ϕ, ρ). The precision of the final solution is closely related to the resolution of accumulators. A trade-off has to be compromised between the coarser discretization which is more noise-immune and efficient but presents low accuracy, and the finer discretization offers more precision but occupies more resource. However, as the proposed HTD is going to be used in global registration techniques designed for initial crude alignment, the coarser grids are preferred.

The second accumulator design problem is how the accumulator cells correspond to patches of the unit sphere. The classic manner samples the unit sphere uniformly in the spherical longitudinal and latitude coordinates, thus the accumulator cells correspond to the patches with equivalent polar and azimuthal angles. This manner is straight and intelligible, but the area of patches are disparate. The patches closer to the equator have larger areas, and the patches closer to the poles are smaller; thus accumulator cells corresponding to patches closer to the equator are correlated to more normal vectors. Another accumulator is presented in [25], and their solution projects the unit sphere onto the smallest cube that contains the sphere. Each face of the cube is discretized regularly. This design is a trade-off between efficiency and manipulability, but the area inequivalent problem is still unsettled. The accumulator designed in [16] leads to each accumulator cell corresponds to the equal patch area. This design samples the unit sphere uniformly in the longitudinal coordinate, but the latitude space of the patch is determined by its longitudinal coordinate to make sure the patches have equal area. Apparently, the azimuthal space between adjacent cells is irregular, and this disorder makes the projection from unit sphere onto cylinder obscure to express.

In our proposed HTD, the classic manner is adopted, further the values of the accumulator cells are normalized by the area of their corresponding patches on the unit sphere.

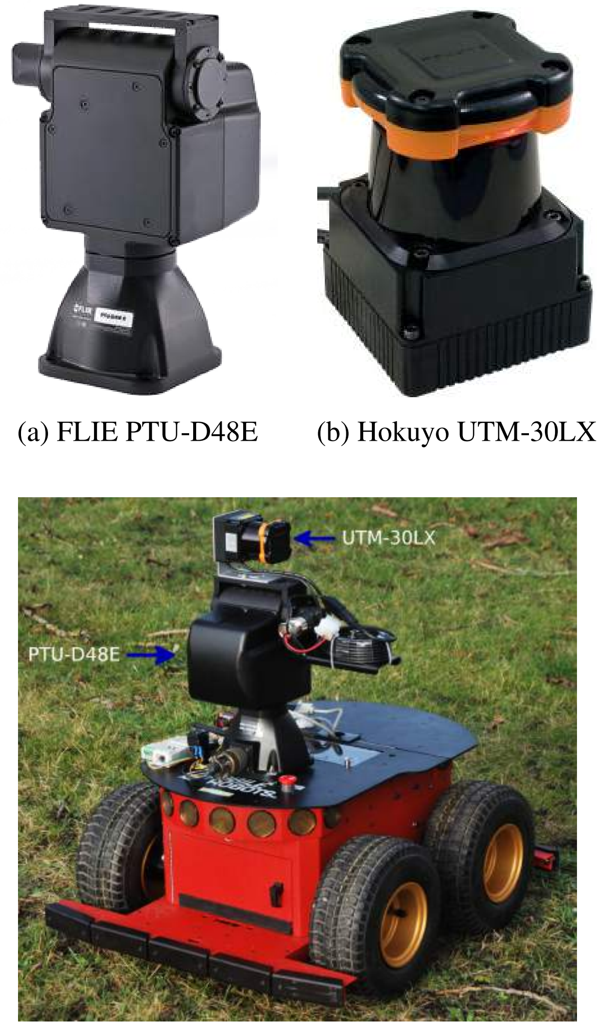
Probabilistic Hough Transform

As shown in Algorithm 1 and 2, the amount of calculation of HTD is directly related to the number of points in original scan. We use Probabilistic Hough Transform [74] to speed up the computation of HTDs.

Since Hough Transform is a voting procedure, the performance is slightly impaired when the subset of original data are used to compute Hough Transform, but the execution time could be shortened considerably. This Probabilistic Hough Transform is analysed in [74] and supported by the experiment results. In other words, it is unnecessary to compute the Hough Transform of every point in scans to obtain HTDs, and the computational complexity of HTDs is adjustable. In our implementation, we select the points uniformly at intervals of 3.

Increments of accumulator cells

A physical point represents a sinusoid surface in Hough space, and normally all the accumulator cells related with the sinusoid surface are incremented by**1**. Due to the inhomogeneity of the points cloud, the planes with equal numbers of points have variable size depending on their distances to the scanner. To solve this ambiguity, the accumulator cells are recommended to be incremented by the area of the points. The area of points could be calculated by its four surrounding closest points. If the scan numbers and the point numbers in the scans are available, the area calculation is straightforward and does not require the computationally expensive nearest neighbour search procedure. By the way, the area of point is not adopted in our implementation of HTD since it is time consuming.

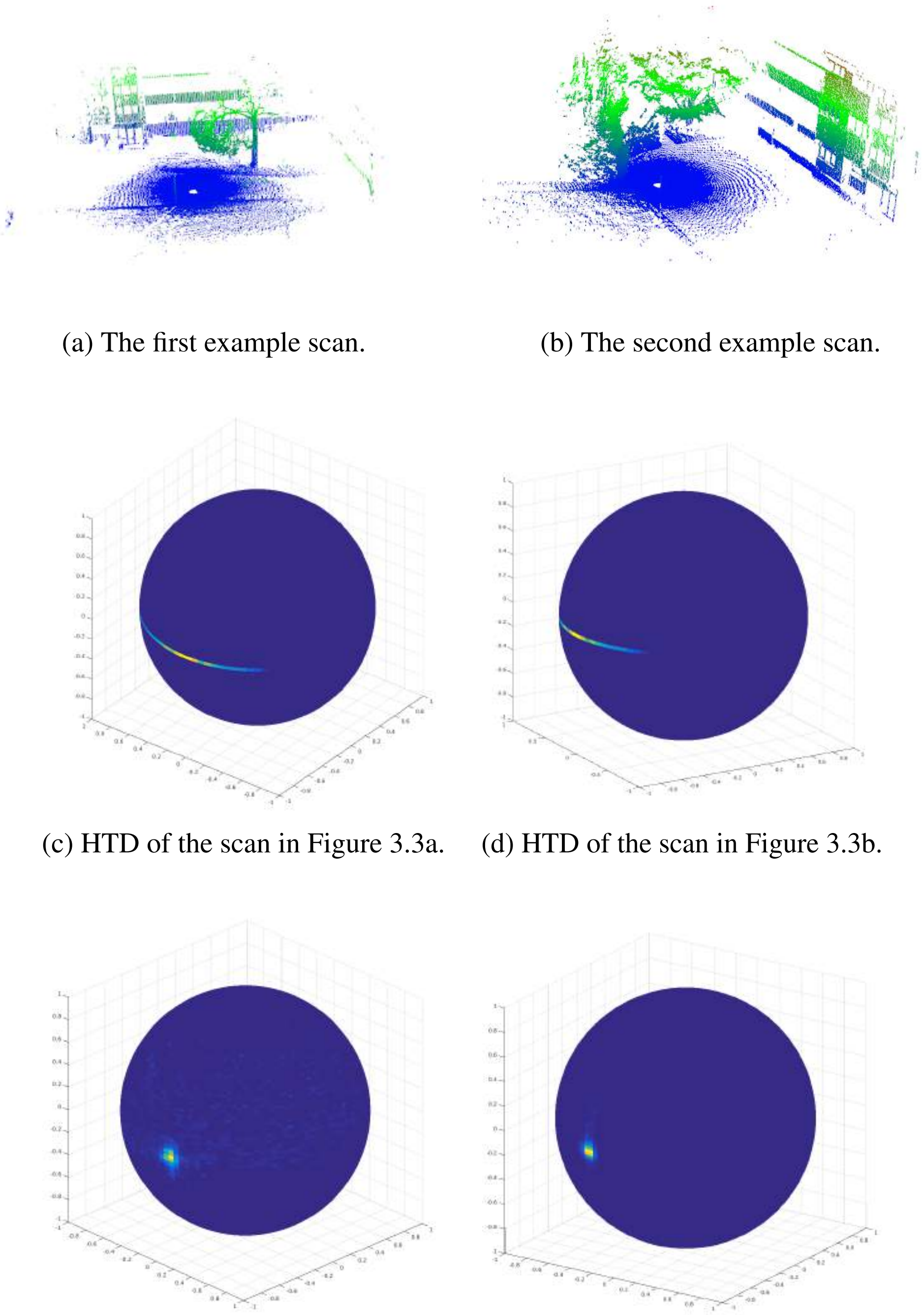


(a) FLIE PTU-D48E (b) Hokuyo UTM-30LX

(c) Platform

Fig. 3.2 Our custom built 3D perception platform

Inhomogeneity of point clouds



(a) The first example scan.

(b) The second example scan.

(c) HTD of the scan in Figure 3.3a.

(d) HTD of the scan in Figure 3.3b.

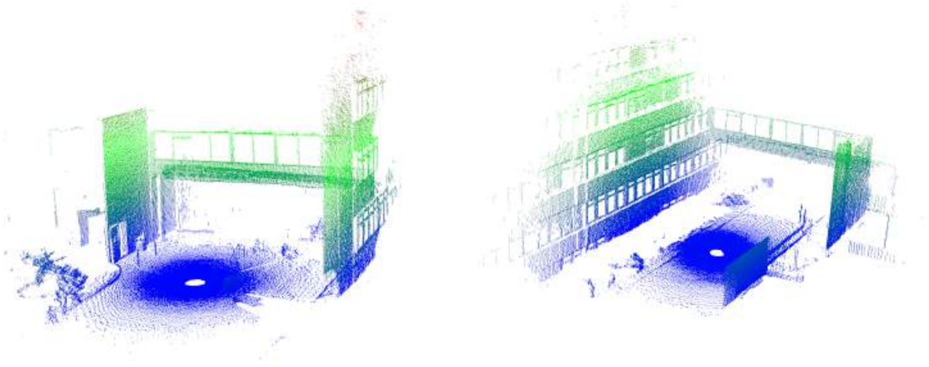
(e) HTD of the scan in Figure 3.3a based on oriented points

(f) HTD of the scan in Figure 3.3b based on oriented points

Fig. 3.3 Examples of Hough Transform Descriptor (HTD) of scans captured by our custom-built platform

Owing to the scan pattern, the density of point clouds is inhomogeneous. Usually the ground points closer to the scanner are excessively dense compare to others. Sometimes, this undue imbalance will overplay the role of the ground and could not fully reveal the structure of objects above the ground, especially when there are sparse constructions in the scene, as the Figure 3.1 shows. And the conformation of the objects is crucial to determine the transformation parameters between scans. To solve this problem, part of points on the ground are excluded when compute HTD.

3.3.3 Examples

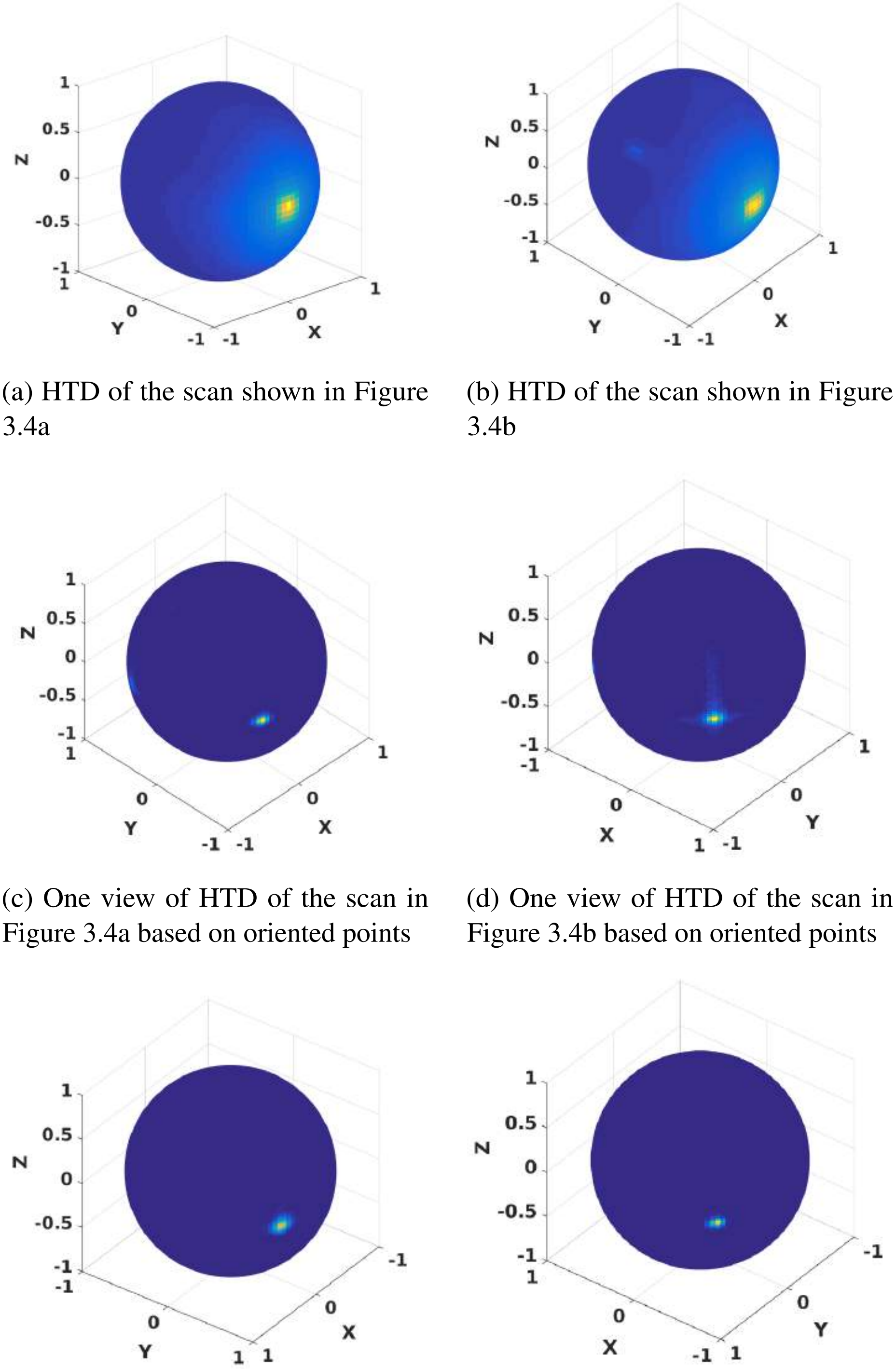


(a) (b)

Fig. 3.4 Example scans captured by our custom-built platform to prove the superiority of HTDs based on oriented points

Several scans obtained by our custom-built 3D perception platform are used to show the HTD of scans. Our custom-built platform is built upon a Pioneer 3-AT robot [1] which is a four wheel drive robotic platform, can be operated on rough-terrain outdoor, and has an on-board computer. The platform is equipped with an aLRF for 3D perception, which is constructed by a FLIR PTU-D48E PTU [47] and a Hokuyo UTM-30LX LRF [55], see Figure 3.2. UTM-30LX is a compact and accurate laser scanner designed for both indoor and outdoor applications, and it can detect objects within range from 0.1 to 30 m in a 270° FoV (up to 0.25° angular resolution). A relative low power consumption – 8.4 Watt (12 Volt 0.7 Ampere) – allows it to be used on mobile robotic platforms.

Two examples of scans captured by our custom-built platform and their corresponding HTDs are presented in Figure 3.3. As the Figure 3.3 shows, the HTDs of scans are sparse images. The peaks of HTD stand for the planes in original scan. And since the square of distances between points and origin are summed, the peaks are extreme large comparing to



(a) HTD of the scan shown in Figure 3.4a

(b) HTD of the scan shown in Figure 3.4b

(c) One view of HTD of the scan in Figure 3.4a based on oriented points

(d) One view of HTD of the scan in Figure 3.4b based on oriented points

(e) Another view of HTD of the scan in Figure 3.4a based on oriented points

(f) Another view of HTD of the scan in Figure 3.4b based on oriented points

Fig. 3.5 Examples of Hough Transform Descriptor (HTD) of scans shown in Figure 3.4 to show the difference between HTD based on direction-less and oriented points.

other points. Our proposed novel scan registration method determines the rotation between scans by estimation the rotation between their corresponding HTDs. Although the method does not depend on the planes in theory, the planes in original scans would certainly reinforce the robustness of this registration method.

Since the scans present in Figure 3.3 contain only one plane, another two scans captured by our custom-built platform are presented in Figure 3.4 aiming to show the difference between HTD based on the direction-less points and oriented points. The HTDs of the scans proposed in Figure 3.4 are depicted in Figure 3.5. Two planes in scans are parallel, so it is reasonable there is only one peak in HTDs based on direction-less points as Figure 3.5a&3.5b show. However, as shown in Figure 3.5c&3.5e and Figure 3.5d&3.5f the HTDs based on oriented points have two peaks. It is worthy to pay attention to the direction difference of axis of Figure 3.5c&3.5e and Figure 3.5d&3.5f. Moreover, we find that the coordinates of the two peaks in HTD based on oriented points of scan in Figure 3.4a are $[-0.4274, -0.9037, 0.02454]$ and $[0.4274, 0.9037, 0.02454]$, while the coordinates of the two peaks in HTD based on oriented points of scan in Figure 3.4b are $[0.7410, -0.6716, -0.02454]$ and $[-0.7407, 0.6714, 0.02454]$. This means the HTDs regard the two parallel planes have opposite direction. It is because we adopt the consistency to determine the normal direction of points. In this way, the HTDs based on oriented points are more descriptive than the HTDs based on direction-less points. Actually, the computation of HTDs based on oriented points are more efficient.

3.4 Scan registration based on HTD

Consider two scans s_1, s_2 with the relationship:

$$s_2 = R \bullet s_1 + t + \varepsilon \tag{3.7}$$

where $R \in SO(3)$, $t \in \mathbb{R}^3$, ε is the noise, and their Hough Transform descriptors are $\mathbf{HTD}_1$ and $\mathbf{HTD}_2$, their translation-invariant descriptors are $\mathbf{TID}_1$ and $\mathbf{TID}_2$. Our algorithm first recovers the R in the TID domain of the two scans; and then applies the obtained R to the HTDs to solve the translation parameters.

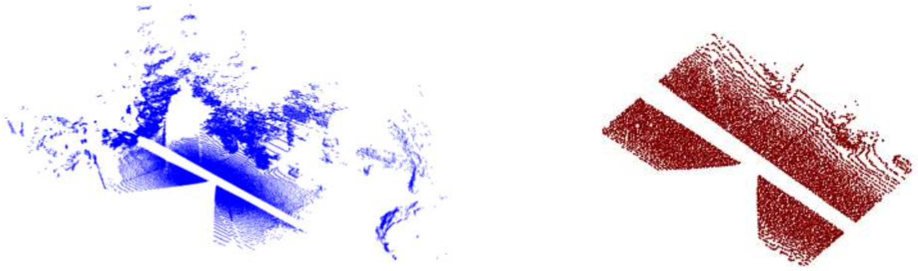
3.4.1 Rotation determination

Obtained the HTDs of two scans, the rotation between the original scans could be estimated by aligning their corresponding HTDs. Theoretically, the Spherical Harmonic Transform could be used to recover the rotation between two spherical functions defined on unit sphere. The technique for pattern matching defined on S^2 through Generalized Convolution Theorem based on the Spherical Harmonic Transform is originated by the researchers in applied mathematics area [40][63], but is attracting more and more attention in pattern recognition [70][96] and medical image processing societies [129][86]. The Generalized Convolution Theorem and Spherical Harmonic Transform is described in section 4.3.1 in detail. But the HTD defined on unit sphere is too spare as Figure 3.3 shows and the Generalized Convolution Theorem plays poorly in these situations.

The estimation of rotation between spherical graphs could be decoupled into determination of rotation axis and rotation angle according to Euler's Rotation Theorem. The rotation axis could be recovered by minimizing the 3D Angular Difference Function (ADF) defined in [71], but the performance of this method deteriorates in the case that the scans have interference and occlusion.

Rotation determination method based on ground surface

A little trick is invented in our previous paper [134] to employ the ground surface to determine the rotation axis, but this technique only validate in robotic mapping scenarios where the ground surface is available.



(a) An example scan from *Gazebo Winter* [112]

(b) The corresponding loose ground points extracted by our algorithm.

Fig. 3.6 Example of ground extraction

The rotation R between two scans could be decomposed into R_1 paralleling two ground surfaces and R_2 compensating the roll about the normal vectors of ground surfaces. Projecting this proposition into the TID domain, the rotation R_1 paralleling the two peaks of TIDs stand for normal vectors of ground surfaces, and the rotation R_2 compensating the rotation of TIDs about the two peaks.

Let the g_1 and g_2 be normal vectors of ground surfaces, the R_1 and R_2 could be easily described by the axis-angle representation:

1. the rotation axis of R_1 is $g_1 \times g_2$ and the rotation angle is the angle between g_1 and g_2;
2. the rotation axis of R_2 is g_2 and the rotation angle of R_2 is symbolized by δ.

Determination of R_1

As to the determination of ground direction, the density of points is greater closer to the scanner because of the scan pattern of the scanner, as shown in Figure 3.6. We develop a method to estimate the normal vector of the ground surface based on this prior knowledge, which needs less extra calculation.

In our ground surface estimation method, the xy-plane segment closer to the scanner is divided into square grids, the dimension of which is a compromise of accuracy and efficiency. Normally the range of this plane segment could be a quarter of the whole xy-plane. And then the 3D points, whose (x,y) values are within the scope of the plane segment, are projected into the cells according to their (x,y) coordinates. For each cell, the point of the smallest z value is stored as loose ground surface points; in this way, the scan is classified into loose ground points and non-ground points. The loose ground points of the scan depicted in Figure 3.6a are shown in Figure 3.6b. Our method could not be used to segment the precise ground surface, since it extracts not only the points belonging to the ground surface but also the bottoms of objects. But our method is computational efficient and capable to deal with the slanted surfaces.

Theoretically, computing the TID of the loose ground points, the normal vector of the ground surface is determined by the main peak of TID. After that, the computation of TID could continue by calculating the Hough Transform of the non-ground points; hence only the operation to separate the loose ground points costs extra computation. But in practise, we found that the normal vector determined by the main peak of TID is sometimes correct with its z direction but false with its x or y direction. Usually, the second or third large local maximum of TID corresponds to correct direction of ground surface. Therefore, we adopt the RANSAC-based (*RANdom SAmple Consensus*) [44] data fitting algorithm to estimate the direction of the ground surface in our code. In addition, we believe this is the same reason why the HSM3D produces several candidate results to improve the robustness [25].

Obtaining the direction of ground surface g_1 and g_2, the R_1 paralleling the ground surfaces could be calculated.

Determination of R_2

There are several methods to determine the rotation angle δ using the TIDs [25]. We convert the estimation of δ to a translation recovering problem by projecting the TID defined on $\mathbb{S}^2$ onto its corresponding cylinder surface $\mathbb{C}^2$ whose axis parallels the normal vector of the ground surface, as Figure 3.7 outlines. In this way, the radial translation of the $\mathbb{C}^2$ corresponds to the rotation angle δ of $\mathbb{S}^2$, and it could be settled by the cross-correlation techniques, for instance Phase Only Matched Filter (POMF).

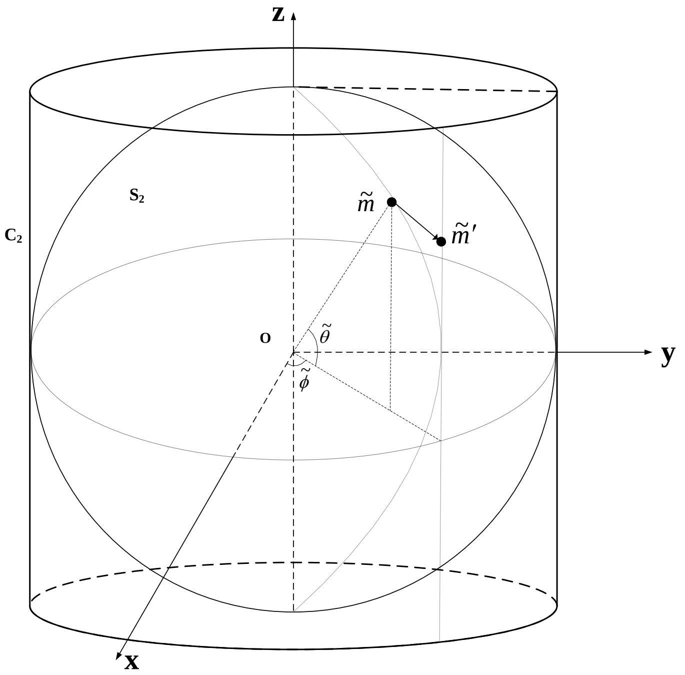
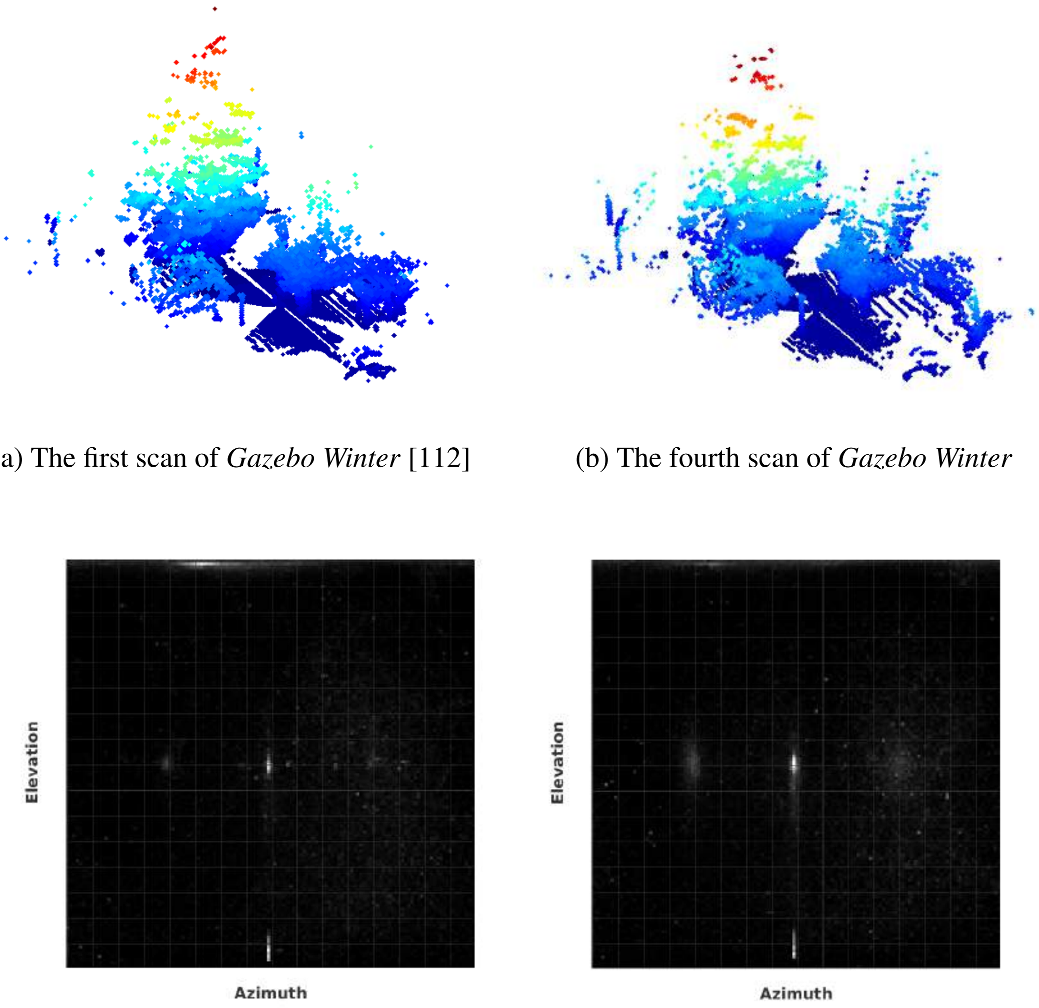


Fig. 3.7 Mapping from a spherical surface onto its corresponding cylinder surface.

As shown in Figure 3.7, let $\mathbb{S}^2$ be a spherical surface and $\mathbb{C}^2$ be its corresponding cylinder surface. Consider a point $\tilde{m}$ on $\mathbb{S}^2$, its the spherical coordinate is $(\tilde{\theta}, \tilde{\phi}, 1)$, and its Cartesian coordinate is $[\cos(\tilde{\theta})\cos(\tilde{\phi}), \cos(\tilde{\theta})\sin(\tilde{\phi}), \sin(\tilde{\theta})]$. And then, re-sampling the Cartesian space by the cylinder coordinate system, the cylindrical coordinate of $\tilde{m}$ could be derived: $[\sin(\tilde{\theta}), \tilde{\phi}, \cos(\tilde{\theta})]$. Finally, map $\tilde{m}$ onto $\mathbb{C}^2$ through normalizing its radial distance to the axis of the cylinder. In such a way, a one-to-one projection $\tilde{m}'$ of $\tilde{m}$ is gained:

$$\begin{aligned}
&\tilde{m} = (\tilde{\theta}, \tilde{\phi}, 1) \\
&\xleftrightarrow{\text{Cartesian resample}} (\cos(\tilde{\theta})\cos(\tilde{\phi}), \cos(\tilde{\theta})\sin(\tilde{\phi}), \sin(\tilde{\theta})) \\
&\xleftrightarrow{\text{Cylindrical resample}} (\cos(\tilde{\theta}), \tilde{\phi}, \sin(\tilde{\theta})) \\
&\xRightarrow{\text{radial distance normalization}} \tilde{m}' = (1, \tilde{\phi}, \sin(\tilde{\theta}))
\end{aligned}$$

After mapping the points of $\mathbb{S}^2$ onto $\mathbb{C}^2$, the rotation angle δ of $\mathbb{S}^2$ around the g_2 is converted into the displacement of the cylinder surface along the tangential direction.



(a) The first scan of *Gazebo Winter* [112]

(b) The fourth scan of *Gazebo Winter*

(c) Ground-calibrated TIDs of first scan

(d) Ground-calibrated TIDs of fourth scan

Fig. 3.8 Examples of ground-calibrated TIDs projected on cylinder surface after R_1 compensation.

Two examples of ground-calibrated TIDs projected on cylinder surface after R_1 compensation are shown in Figure 3.8. It could be seen that there is only tangential shift left after the compensation using R_1. Achieving the cylinder surfaces, the determination of R_2 is converted to a uni-dimensional signal registration issue. But in our implementation, the two dimensional POMF is adopted since the shift along the axis could be applied to validate whether the R_1 is solid or not.

The registration method using ground surface to determine the rotation is outlined in Algorithm 3. This ground surface-based rotation determination method is theoretical complete when ground surfaces are available and solid. But the dependence on ground surface is its inherent defect.

Algorithm 3: The registration method based on HTD using ground surface to determine the rotation

```
Input  : Point clouds $S_1$, $S_2$
Output : Rotation matrix $R$ and translation vector $t$
▷ Estimate the directions of grounds: $g_1, g_2$
$\mathbf{GS}_1 \leftarrow \mathbf{GroundExtraction}(S_1)$ ;
$\mathbf{GS}_2 \leftarrow \mathbf{GroundExtraction}(S_2)$ ;
$g_1 \leftarrow \mathbf{RANSAC}(\mathbf{GS}_1)$ ;
$g_2 \leftarrow \mathbf{RANSAC}(\mathbf{GS}_2)$ ;
▷ Calculate $R_1$ based on the directions of grounds
$R_1 \leftarrow \mathbf{Euler'sRotationTheorem}(g_1, g_2)$ ;
$S_2^{'} \leftarrow R_1 * S_2$ ;
▷ Compute the HTDs & TIDs of $S_1$, $S_2^{'}$
$\mathbf{HTD}_1 \leftarrow \mathbf{HoughTransform}(S_1)$ ;
$\mathbf{HTD}_2 \leftarrow \mathbf{HoughTransform}(S_2^{'})$ ;
$\mathbf{TID}_1 \leftarrow \sum_\rho(\mathbf{HTD}_1)$ ;
$\mathbf{TID}_2 \leftarrow \sum_\rho(\mathbf{HTD}_2)$ ;
▷ Estimate the rotation $R_2$
$\mathbf{TID}_1^{'} \leftarrow \mathbf{ProjectToCylinder}(\mathbf{TID}_1)$ ;
$\mathbf{TID}_2^{'} \leftarrow \mathbf{ProjectToCylinder}(\mathbf{TID}_2)$ ;
$\delta \leftarrow \mathbf{POMF}(\mathbf{TID}_1^{'}, \mathbf{TID}_2^{'})$ ;
$R_2 \leftarrow \mathbf{Euler'sRotationTheorem} < g_2, \delta >$ ;
$R \leftarrow R_1 * R_2$ ;
▷ Recover the translation
$\overline{S_2} \leftarrow R_2 * S_2^{'}$ ;
$P_1, v_1 \leftarrow \mathbf{Rasterize}(S_1)$ ;
$P_2, v_2 \leftarrow \mathbf{Rasterize}(S_2^{'})$ ;
$t_x, t_y \leftarrow \mathbf{2D\_POMF}(P_1, P_2)$ ;
$t_z \leftarrow \mathbf{1D\_POMF}(v_1, v_2)$ ;
```

General rotation determination method

Considering the shortcoming of ground surface-based rotation determination methods, we develop a more general method. In this kind of rotation determination method, the rotation is divided into two steps: the yaw determination and the roll & pitch determination. Overall speaking, we resample the TIDs to convert the yaw/roll/pitch to offsets of resampled TIDs, and use POMF to recover the offsets.

Yaw Determination

The yaw angle is determined just like the determination of R_2 in ground surface-based method described in section 3.4.1. Project the TIDs onto cylinders just like depicted in Figure 3.7, and in this way the radial translation of $\mathbb{C}^2$ corresponds to yaw angle of spherical functions. And the radial translation of $\mathbb{C}^2$ could be recovered by POMF.

It is worth noting that the roll and pitch are treated as undesirable noise in yaw angle determination. That means this method also has its own inherent defect: it could not handle the situations when roll and pitch offsets between scans are too large.

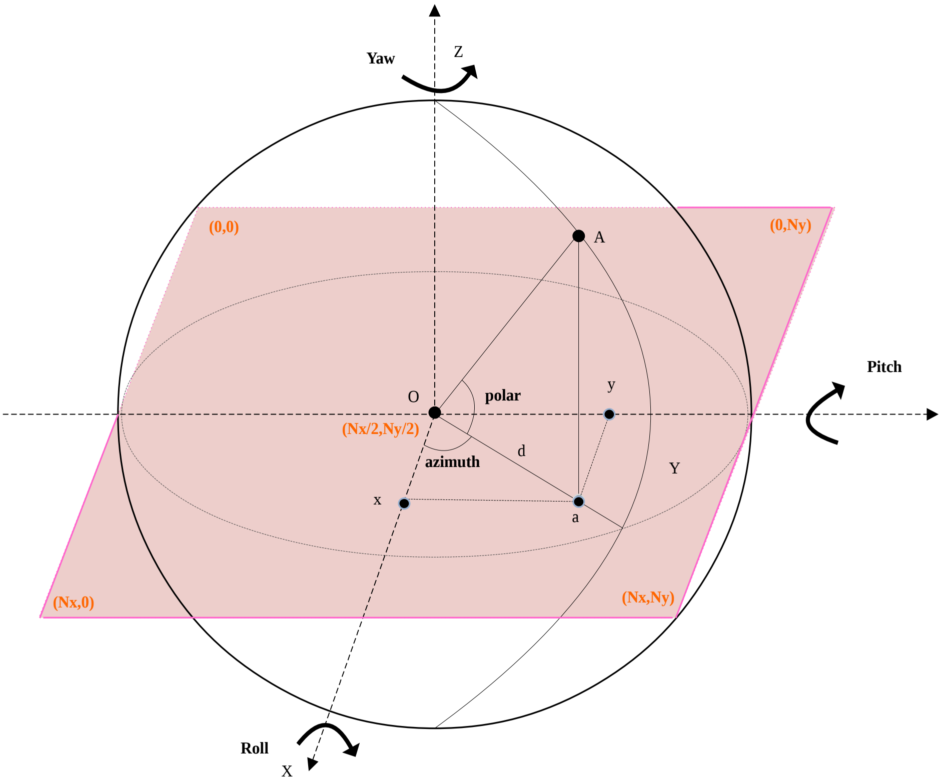


Fig. 3.9 Project the upper spherical surface onto its corresponding plane.

Roll & Pitch Determination

Before roll & pitch determination, first apply the achieving yaw angle to rerotate the TIDs on sphere, which is the basis for the following unwrap procedure.

In roll & pitch determination step, we just unwrap one half sphere to determine the roll & pitch. It is natural to select the half which contains rich information. So we compute the entropies of the two halves separately and select the half whose entropy is large, and project the TID on this half of sphere onto a plane.

Assume we project the TID on the above half sphere onto a plane, just like Figure 3.9 shows, according to the following equations:

$$d = \sqrt{(y - \frac{N_y}{2})^2 + (x - \frac{N_x}{2})^2} \tag{3.8}$$

$$\frac{d}{r} = \frac{polar}{\pi/2} \tag{3.9}$$

$$\tan(azimuth) = \frac{y - \frac{N_y}{2}}{x - \frac{N_x}{2}} \tag{3.10}$$

In this way, the determination is converted into the translation of the unwrapped plane: the roll angle corresponds to the translation of the unwrapped plane along positive y axis, and the pitch angle is related to the translation of the unwrapped plane along positive$x$ axis. Moreover if we project the TID on the bottom half sphere onto the unwrapped plane, the roll angle corresponds to the offset along negative y axis of plane, and the pitch angle is related to the offset along negative x axis of plane.

This kind of registration method is outlined in Algorithm 4. It is worth to mention that the sphere rotation in our implementation is based on the public library about spherical harmonic transforms SpharmonicKit [40].

3.4.2 Translation recovery

Once the rotation R is determined correctly, there is only translation between two scans after applying the R to the scans. We apply the POMF [57][48] to estimate the shift between two scans. The POMF employs the fact that two shifted signals carry the shift information within the phase of their Fourier spectrum and decouples the local signal energy from the signal structure.

Let $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$ symbolize two N-dimensional shifted signals where $\boldsymbol{x} = \{x_1, x_2, \ldots, x_n\}$, while $F_1(\boldsymbol{\alpha})$ and $F_2(\boldsymbol{\alpha})$ are their corresponding Fourier spectra. The shift between two translated duplicates could be determined by the following equations:

$$S(\boldsymbol{\alpha}) = \frac{\overline{\mathscr{F}_1(\boldsymbol{\alpha})}}{|\mathscr{F}_1(\boldsymbol{\alpha})|} \bullet \frac{\mathscr{F}_2(\boldsymbol{\alpha})}{|\mathscr{F}_2(\boldsymbol{\alpha})|} \tag{3.11}$$

$$s(\boldsymbol{x}) = \mathscr{F}^{-1}\{S(\boldsymbol{\alpha})\} \tag{3.12}$$

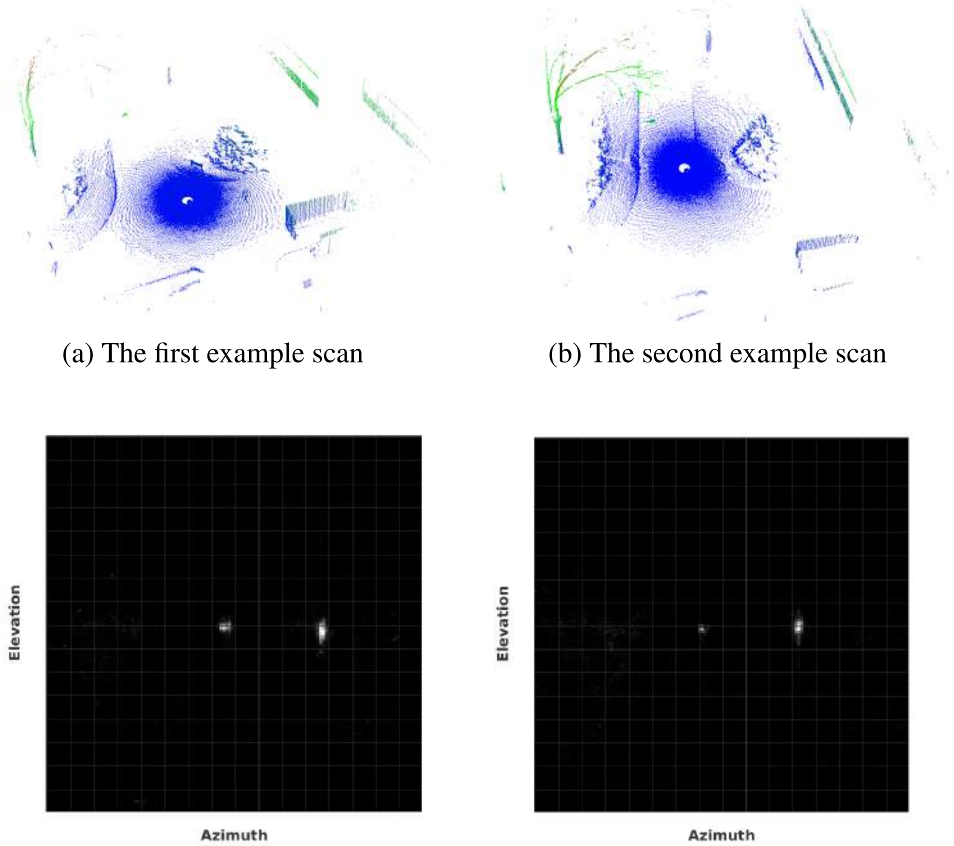
$$\boldsymbol{x}_t = \arg\max_{\boldsymbol{x}} s(\boldsymbol{x}) \tag{3.13}$$

$\boldsymbol{x}_t$ is the displacement between $f_1(\boldsymbol{x})$ and $f_2(\boldsymbol{x})$. In theory, it could be used in arbitrary dimensional signal registration problems. Ideally, the $s(\boldsymbol{x})$ contains a Dirac peak, but the Dirac pulse deteriorates in practice due to the noise and the partial overlap. The Fourier Transform calculated in translation determination is the common Cartesian Fourier Transform (compared to the Spherical Harmonic Transform proposed in Section 4.3.1).

Algorithm 4: A general registration method based on HTD.

```
Input  : Point clouds $S_1$, $S_2$
Output : Rotation matrix $R$ and translation vector $t$
▷ Compute the HTDs & TIDs of $S_1$, $S_2$
$\mathbf{HTD}_1 \leftarrow \mathbf{HoughTransform}(S_1)$ ;
$\mathbf{HTD}_2 \leftarrow \mathbf{HoughTransform}(S_2)$ ;
$\mathbf{TID}_1 \leftarrow \sum_\rho(\mathbf{HTD}_1)$ ;
$\mathbf{TID}_2 \leftarrow \sum_\rho(\mathbf{HTD}_2)$ ;
▷ Estimate the yaw angle
$\mathbf{TID}_1^{'} \leftarrow \mathbf{ProjectToCylinder}(\mathbf{TID}_1)$ ;
$\mathbf{TID}_2^{'} \leftarrow \mathbf{ProjectToCylinder}(\mathbf{TID}_2)$ ;
$\mathbf{Yaw} \leftarrow \mathbf{POMF}(\mathbf{TID}_1^{'}, \mathbf{TID}_2^{'})$ ;
$R_1 \leftarrow \mathbf{Euler'sRotationTheorem} < \mathbf{Yaw} >$ ;
▷ Estimate the roll & pitch angle
$\overline{\mathbf{TID}_2} \leftarrow \mathbf{SphereRotation}(\mathbf{TID}_2, \mathbf{R}_1)$ ;
$\mathbf{TID}_1^{''} \leftarrow \mathbf{Unwrap}(\mathbf{TID}_1)$ ;
$\mathbf{TID}_2^{''} \leftarrow \mathbf{Unwrap}(\overline{\mathbf{TID}_2})$ ;
$(\mathbf{Roll}, \mathbf{Pitch}) \leftarrow \mathbf{POMF}(\mathbf{TID}_1^{''}, \mathbf{TID}_2^{''})$ ;
$R_2 \leftarrow \mathbf{Euler'sRotationTheorem} < \mathbf{Roll}, \mathbf{Pitch} >$ ;
▷ Recover the translation
$R = R_1 * R_2$ ;
$S_2^{'} = R * S_2$ ;
$P_1, v_1 \leftarrow \mathbf{Rasterize}(S_1)$ ;
$P_2, v_2 \leftarrow \mathbf{Rasterize}(S_2^{'})$ ;
$t_x, t_y \leftarrow \mathbf{2D\_POMF}(P_1, P_2)$ ;
$t_z \leftarrow \mathbf{1D\_POMF}(v_1, v_2)$ ;
```

Before applying the POMF or 3D FFT, it is necessary to rasterize the original 3D surface into volume grids. In general, the way to rasterize the 3D surface is assigning voxels the value of 1 if they are occupied by the surface, otherwise their values are set to 0. The Euclidean Distance Transform is applied to 3D surface in [70], and then the value at each voxel is given by the negatively exponentiated Euclidean Distance Transform of the point located in it.



(a) The first example scan

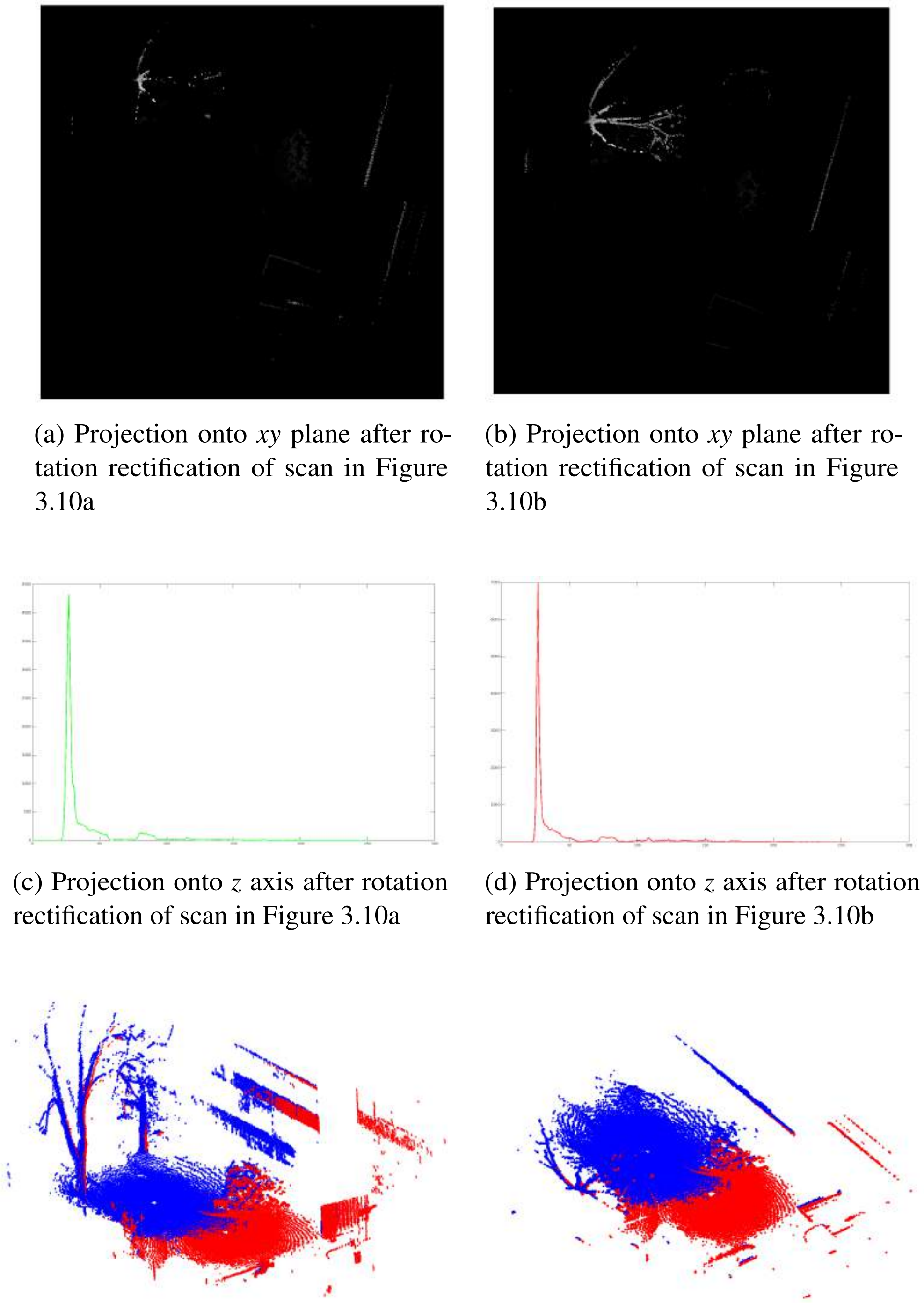
(b) The second example scan

(c) Ground-calibrated TIDs projected on cylinder of scan in Figure 3.10a

(d) Ground-calibrated TIDs projected on cylinder of scan in Figure 3.10b

Fig. 3.10 Example scans employed to prove effectiveness of our translation recovery algorithm

The 3D POMF could be adopted to estimate the translation between two shifted scans straightforwardly, or in another way, it is also feasible to integrate the scan along the x, y, z direction and apply the 1D POMF to calculate the translation separately. In our algorithm, we divide the translation recovery into two steps: recover the offset in(x, y) direction and in z direction. For two scans after rotation rectification, we project the scans onto its (x, y) plane to use 2D POMF to estimate the offset in x and y direction. In the projection procedure, the xy pixel stores the maximum z value of all the points whose (x, y) locate in this pixel, since



(a) Projection onto xy plane after rotation rectification of scan in Figure 3.10a

(b) Projection onto xy plane after rotation rectification of scan in Figure 3.10b

(c) Projection onto z axis after rotation rectification of scan in Figure 3.10a

(d) Projection onto z axis after rotation rectification of scan in Figure 3.10b

(e) Side view of the registration result of scans in Figure 3.10a&3.10b

(f) Top view of the registration result of scans in Figure 3.10a&3.10b

Fig. 3.11 Example proving effectiveness of our translation recovery algorithm

the variation of the original data is significant for the FFT-based techniques. Meanwhile project the scans onto its z axis to use 1D POMF to calculate the offset in z direction. The number of points belongs to the specific z interval is stored. It is worthy to point out that the projection onto (x,y) axis and z axis is accomplished in one iteration, rather than two separate iteration of points in scans.

In our implementation of POMF, we adopt the functions in FFTW [50] to calculate the forward and backward of 1D/2D FFT. An registration example to prove the effectiveness of our translation recovery method is presented in Figure 3.10 and Figure 3.11 . We can see that there is large translation between two scans, and our translation recovery method could solve the translation effectively.

3.5 Experiments and results

In this section, the Hough Transform Descriptor (HTD) is applied in scan registration, and compared with the existing and representative scan registration algorithms based on the public available datasets and the data captured by our custom-built platform. The reason why we adopt the public available dataset is that it is convenient for other researchers to compare our algorithm with their own algorithms.

Experiments have been carried on standard personal desktop computers. Our scan registration algorithm is implemented in C++, and the code has been published online, please refer to Appendix A. The Point Cloud Library (PCL) [107] is utilized for reading and writing point clouds, estimating the normals of points and 3D visualization. The linear algebra library Eigen [53], a C++ template library for linear algebra, is employed for matrix processing. The Discrete Fourier Transform library FFTW [50] is adopted in translation recovery procedure to compute the forward and backward Fourier Transform. The *std::chrono::high_resolution_clock* class in **C++ 11** is employed to record the runtime.

In section 3.5.1, we review three representative scan registration algorithms, which are employed to compare with the HTD-based scan registration method. In section 3.5.2, we introduce the "Barcelona Robot Lab Dataset" which is captured in UPC Nord Campus in Barcelona and propose the experiment results based on this dataset. We introduce the dataset recorded at University of Hamburg, Informatik campus by our custom-built platform and propose the experiment results based on this dataset in section 3.5.3. Due to the space limitation, we cannot present all the results. Indeed, some typical results have been selected for explanation.

3.5.1 Related work

Iterative Closest Point (ICP)

Iterative Closest Point (ICP) [30][12][111][118] is the most famous and widely used registration method. ICP aims to minimize the difference between two point clouds based on an iterative procedure. In this kind of algorithms, the given initial transformation matrix is refined and the point clouds are transformed iteratively to minimize the distance from one scan to another.

Typically, the ICP algorithms contain four steps:

1. **Data Association:** for each point in source scan, find its closest point in target scan;
2. **Estimate Transformation:** estimate the transformation matrix based on the point pairs built in step 1;
3. **Error Minimization:** transform the source scan using the achieved transform matrix estimated in step 2;
4. **Check:** check if termination criteria is fulfilled, if not, go to step 1.

In data association, usually the Euclidean distance is used and kd-tree is employed to accelerate the search procedure [43]. Further, the common distance metrics are point-to-point [12] and point-to-plane [30].

PCL is a good example implementing ICP algorithm. It contains different kinds of ICP algorithms. In this experiment, we use the *pcl::IterativeClosestPointWithNormals* class in PCL which uses a transformation estimated based on point-to-plane distances.

Fast Point Feature Histograms (FPFH)

FPFH-based scan registration is a representative local feature-based registration method. FPFH is a simplified version of Point Feature Histograms (PFH), and it reduces the computational complexity but retains most of the discriminative power of PFH. FPFH is calculated based on the oriented points, but it does not make use of the intensity or color information.

Generally speaking, FPFH calculation of a scan includes three steps:

1. for every point p_i in the scan, collect its all neighbouring points within a sphere P_i^N;

2. for every point pair (p_i,p_j) where $p_j \in P_i^N$, the point whose normal has the smaller angle to the vector $\overrightarrow{p_i - p_j}$ is called source point p_s, the other one is named target point p_t; define a Darboux *uvn* frame ($u = n_s, v = (p_t - p_s) \times u, w = u \times v$) and calculate three features which together to express the mean curvature at p_t:

   $$\alpha = v \cdot n_j$$
   $$\phi = \frac{u \cdot (p_t - p_s)}{\|p_t - p_s\|}$$
   $$\theta = arctan(w \cdot n_t, u \cdot n_t)$$

   The three features are combined and put into the equivalent histogram bin. The histograms are called Simplified Point Feature Histogram (SPFH). Note that ever point p_i has its own SPFH.

3. After achieving SPFH of every point, for each point p_i redetermine its k neighbours and use the neighbouring SPFH values to weight the final histogram of p_i:

   $$\mathbf{FPFH}(p_i) = \mathbf{SPFH}(p_i) + \frac{1}{k}\sum_{j=1}^{k}\frac{1}{\omega_k} \cdot \mathbf{SPFH}(p_j)$$

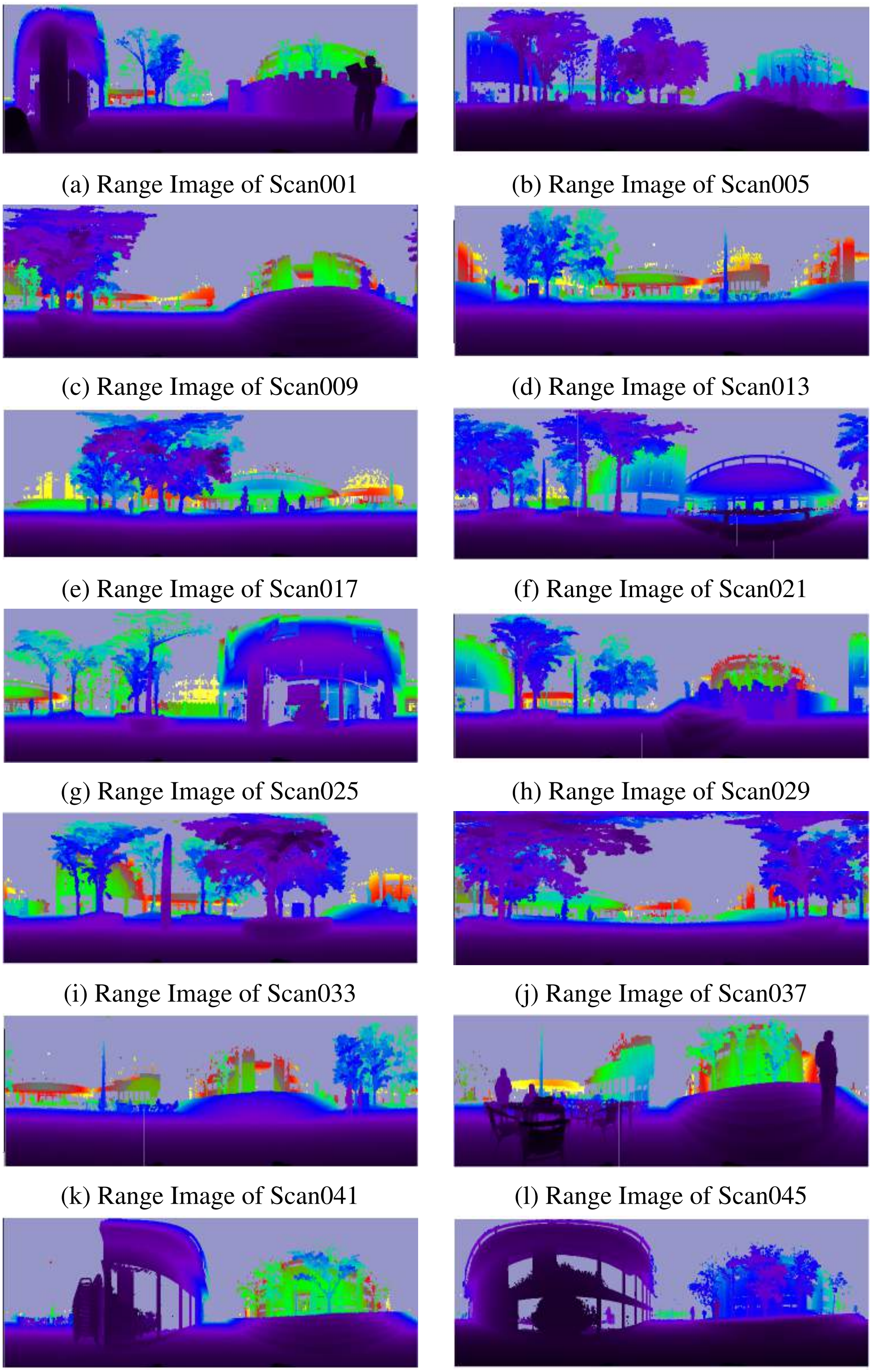
FPFH is also implemented in PCL. We use the *pcl::FPFHEstimation* class in PCL to compute FPFH in this experiment.

Plane-based registration algorithm

We adopt the most recent plane-based scan registration algorithm [163] as a representative of plane-based registration techniques in this experiment to compare with our HTD-based scan registration methods. This scan registration method contains three steps:

1. extract planar segments from scans utilizing a cached-octree region growing (CORG) method, which does not require the point clouds to be organized;
2. calculate the area of each planar segments;
3. find corresponding planar segments between overlapping point clouds and calculate the transformation matrix from determined correspondence.

In Step 3, the transformation is searched globally as to maximize a spherical correlation like metric, by enumerating solutions derived from potential segment correspondences. C++ implementation of this algorithm is available on *https://github.com/junhaoxiao/TAMS-Planar-Surface-Based-Perception.git*.



(a) Range Image of Scan001

(b) Range Image of Scan005

(c) Range Image of Scan009

(d) Range Image of Scan013

(e) Range Image of Scan017

(f) Range Image of Scan021

(g) Range Image of Scan025

(h) Range Image of Scan029

(i) Range Image of Scan033

(j) Range Image of Scan037

(k) Range Image of Scan041

(l) Range Image of Scan045

(m) Range Image of Scan049

(n) Range Image of Scan053

Fig. 3.12 Range images of the selected scans in "Barcelona Robot Lab Dataset"

3.5.2 The "Barcelona Robot Lab Dataset"

The "Barcelona Robot Lab" dataset covers about 10000 m^2 of the UPC Nord Campus in Barcelona, which is intended for use in mobile robotics and computer vision research. Multiple sensor channels are provided in the dataset, including odometry information, compass data, unorganized point clouds, on-board robot imagery as well as imagery from a camera sensor network. This dataset contains 400 scans with about 195000 points each. The point clouds are captured by Hokuyo UTM-30LX scanner mounted in a slip-ring, just like our custom-built platform. The "Barcelona Robot Lab" dataset is public available on: *http://www.iri.upc.edu/research/webprojects/pau/datasets/BRL/index.php*.

In this experiment, we select 14 scans in the first 53 scans by the interval of four, in this way, there are large offsets and less overlaps between the successive scans. We adopt these 14 scans to prove that our HTD-based scan registration technique could handle the scans share small overlaps. The range images of the selected 14 scans are shown in Figure 3.12. Further, the range images are generated by *pcl::RangeImageSpherical* Class and visualized by *pcl::visualization::RangeImageVisualizer* Class.

Performance of HTD-based algorithm

The registration results by our HTD-based scan registration algorithm are proposed in Table 3.1. Our scan registration algorithm succeeds in 11 scan pairs, but fails when align the 21th scan to 17th scan and 41th scan to 37th scan. But if we add 18th & 19th and 39th scans into the registration chain, the registration algorithm performs well, just like the bottom part of Table 3.1 depicts.

Figures 3.13 and 3.14 show the central four scan pairs, 25& 21, 29& 25, 33& 29, 37& 33. It could be seen from the original scan pairs that how large the rotation offsets between the scans are, while the large translation could be seen from the registration results. For the 25& 21, 29& 25, 33& 29 scan pairs, there are about 90 degree rotations between them. And our HTD-based scan registration algorithm could handle them easily and stably. For the 37& 33 scan pair, there is large offset between them, and our transformation recovery method could estimate the translation based on the little overlap.

Please note that the registration results shown in Figures 3.13 and 3.14 are the results of our HTD-based registration method without the refinement by local registration methods. Compare with other global registration algorithms, such as FPFH-based registration method [119], plane-based registration method [163], and SEI-based registration algorithm proposed in Section 4.3, this HTD-based registration usually produces more precise registration results.

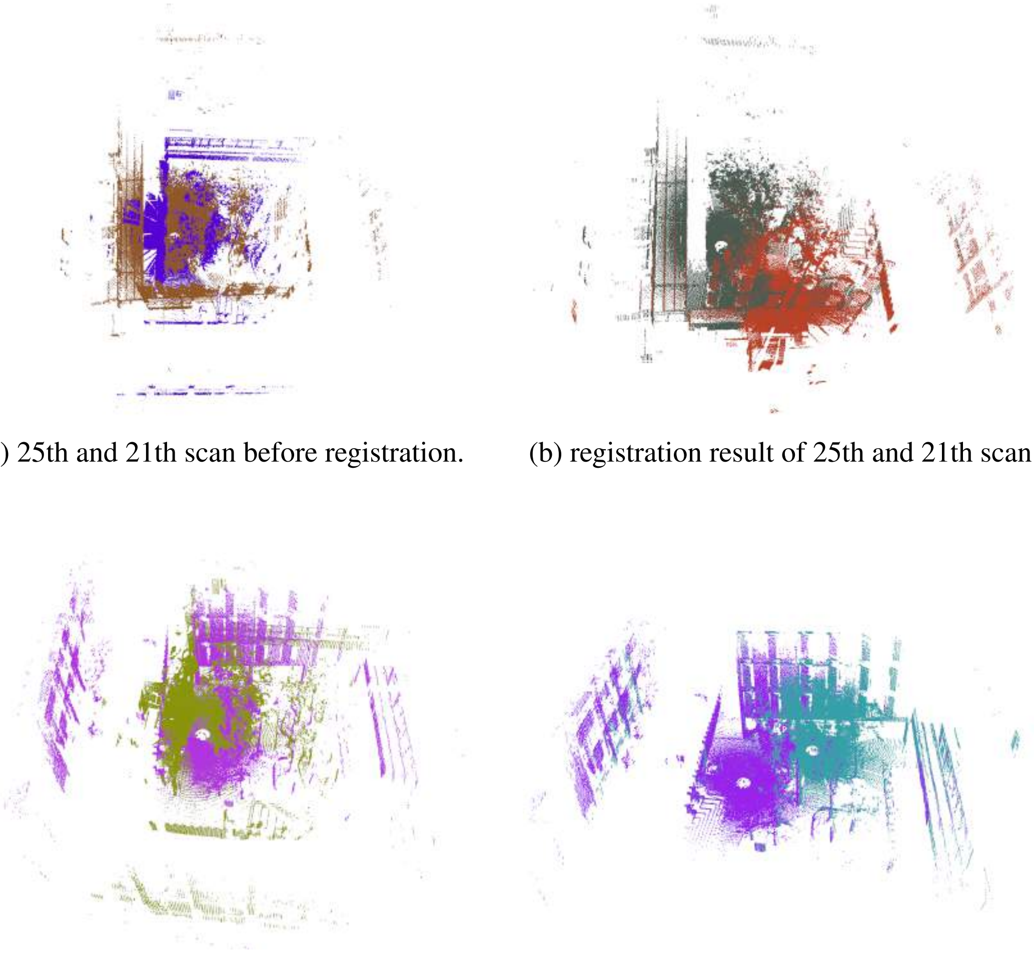
Table 3.1 Registration results of our HTD-based scan registration algorithm applied to "Barcelona Robot Lab Dataset".

Scan Pairs	Rotation (radian)			Translation (m)		
	Roll	Pitch	Yaw	X	Y	Z
5 → 1	0.0141	-0.0361	0.3456	7.93056	2.08306	-0.178802
9 → 5	0.0107	-0.0081	0.4095	6.88793	-4.42686	-0.0607769
13 → 9	0.0080	0.0153	1.3060	5.00036	4.13597	-0.0601081
17 → 13	0.0200	-0.0017	1.0877	5.0177	4.17567	-0.101578
21 → **17**	—	—	—	—	—	—
25 → 21	0.0710	0.0048	1.6151	6.58132	7.18973	-0.311634
29 → 25	0.0191	0.0075	1.4624	6.82892	5.88536	-0.0647581
33 → 29	0.0011	-0.0240	1.3753	2.894	4.03988	-0.0834897
37 → 33	0.0179	-0.0139	1.1721	5.79207	-4.92743	-0.188376
41 → **37**	—	—	—	—	—	—
45 → 41	0.0085	-0.0011	0.9090	6.8145	3.88789	-0.105301
49 → 45	0.0254	0.0198	0.1382	8.15788	0.55841	0.047969
53 → 49	0.0019	-0.0032	0.0148	11.1342	-0.756214	-0.181826
18 → **17**	0.0210	0.0141	0.1891	2.76644	0.434501	-0.0781493
19 → 18	0.0322	-0.0136	0.2172	2.74701	0.25058	-0.0594296
21 → 19	0.0140	-0.0293	0.2206	5.51585	0.481793	-0.122603
39 → **37**	0.0015	0.0362	0.6663	2.59516	-1.05736	0.0456518
41 → 39	0.0186	-0.0297	0.3887	5.15237	-2.13771	-0.202215

The registration is implemented on the standard personal desktop computer with the following details:

- **Memory:** 7.7GiB
- **Processor:** Intel® Core™ i5-3750 CPU @ 3.4GHZ
- **System:** 64-bit Ubuntu 12.04 LTS

Specifically, with regard to efficiency of the algorithm, the processing time without resampling of the original scans is listed in Table 3.2. We could see from Table 3.2 that the runtime of our algorithm is quite stable, and most time (more than half) is used to compute the normal of points which could be implemented offline and separately before



(a) 25th and 21th scan before registration.

(b) registration result of 25th and 21th scan.

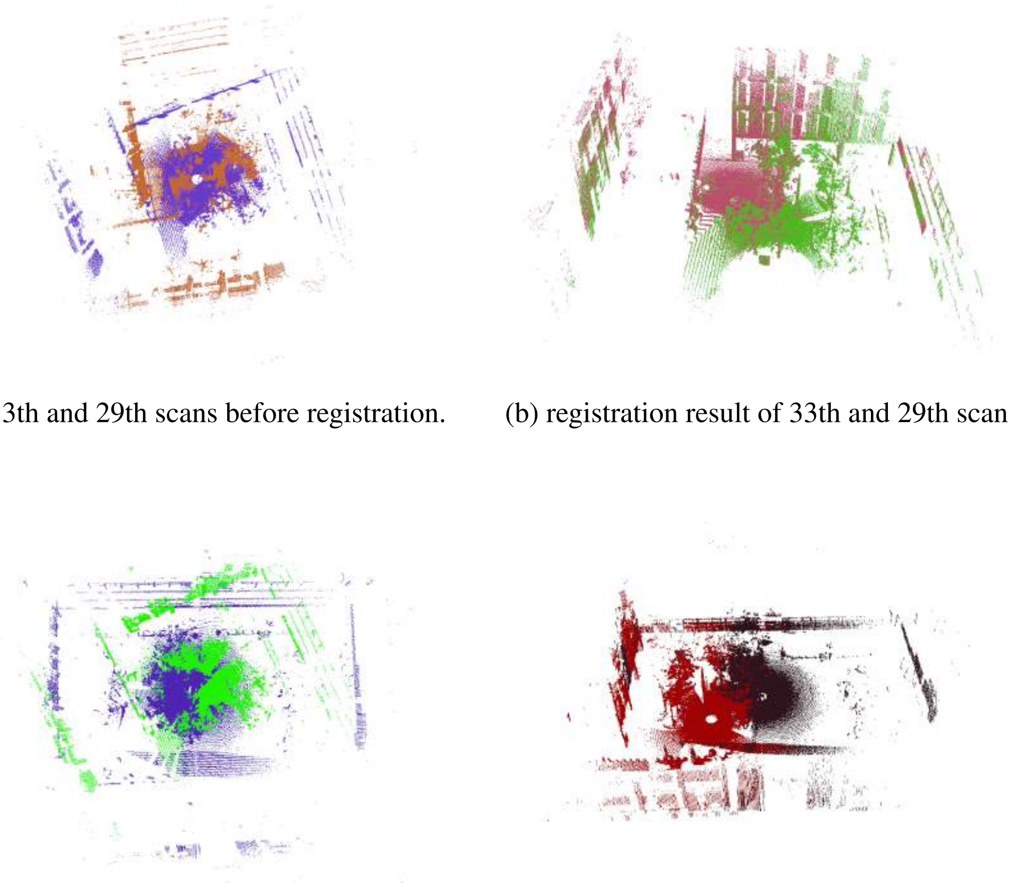
(c) 29th and 25th scan before registration.

(d) registration result of 29th and 25th scan.

Fig. 3.13 Examples of registration results of "Barcelona Robot Lab Dataset" by our HTD-based scan registration algorithm (I).

online registration. Particularly, the normal of points in former scan could be calculated while the robot captures the next scan. The time used for normal estimation and HTD calculation listed in Table 3.2 contains the time required by both the object and scene scans in each pair. Actually in the successive registration chain, it is only necessary to calculate one time. For example, in the procedure aligning 17th scan to 13th scan, the algorithm only needs to compute the normals and HTD of the 17th scan, since the normal and HTD of 13th scan has been calculated in the former registration procedure. It is the same to projecting point clouds onto *xy* plane and *z* axis in translation recovery procedure.

It is worthy to point out that we also present the runtime of the algorithm although it fails in 21& 17 and 41& 37, since it is also important how long the registration methods take to tell that they could not achieve the goal.



(a) 33th and 29th scans before registration.

(b) registration result of 33th and 29th scan.

(c) 37th and 33th scans before registration.

(d) registration result of 37th and 33th scan.

Fig. 3.14 Examples of registration results of "Barcelona Robot Lab Dataset" by our HTD-based scan registration algorithm (II).

Comparison with the state-of-the-art

In order to prove the superiority of our registration algorithm, we also compare our HTD-based scan registration method with the state-of-the-art methods introduced in Section 3.5.1. The code implementing ICP algorithm based on *pcl::IterativeClosestPointWithNormals* class in PCL and FPFH algorithm based on *pcl::FPFHEstimation* class in PCL are also public available on the Internet, please refer to Appendix A.

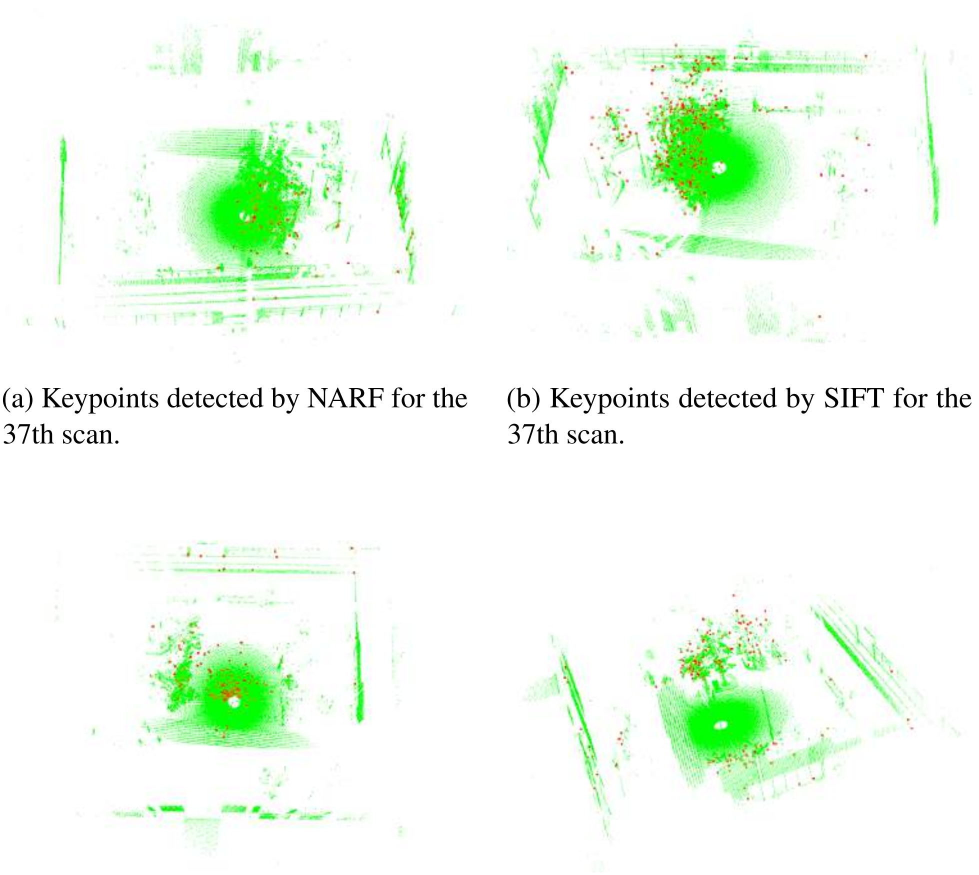
FPFH algorithm combining the keypoints detection methods NARF and 3D SIFT is also accessible in our code, but it is not adopted in this comparison experiments. Instead, we use the uniformly resampled points as the keypoints, which takes more time but performs much better. The examples of the keypoints detected by NARF and 3D SIFT are shown in

Table 3.2 Processing time of our HTD-based scan registration algorithm applied to "Barcelona Robot Lab Dataset" (without resampling).

Scan Pairs	Time per step (s)				Total (s)
	Normal Estimation	HTD Calculation	Rotation Determination	Translation Recovery	
5 → 1	1.380134	0.380134	0.015455	0.672082	2.447805
9 → 5	1.361336	0.384292	0.009453	0.743060	2.498141
13 → 9	1.210614	0.335550	0.009496	0.630288	2.185948
17 → 13	1.172073	0.330593	0.009377	0.576633	2.088676
21 → 17 (FAIL)	1.397406	0.419999	0.009307	0.706259	2.532971
25 → 21	1.495730	0.453491	0.009284	0.649925	2.608430
29 → 25	1.378997	0.396082	0.009294	0.667856	2.452229
33 → 29	1.420831	0.395692	0.009355	0.650712	2.476590
37 → 33	1.605091	0.453687	0.009324	0.728680	2.796782
41 → 37 (FAIL)	1.358607	0.373290	0.009588	0.638435	2.379920
45 → 41	1.030726	0.284181	0.009396	0.498335	1.822638
49 → 45	1.174090	0.321975	0.009434	0.589456	2.094955
53 → 49	1.354634	0.379299	0.009353	0.754029	2.497315
18 → **17**	1.589441	0.446769	0.009356	0.758911	2.804477
19 → 18	1.934572	0.534620	0.009469	0.830013	3.308674
21 → 19	1.736309	0.510100	0.009504	0.787791	3.043704
39 → **37**	1.433463	0.396889	0.009715	0.710188	2.550255
41 → 39	1.060626	0.294575	0.009387	0.588731	1.953319

Figure 3.15. The keypoints detection methods could reduce the runtime of FPFH registration algorithm dramatically, but the success ratio of FPFH algorithm decreases seriously. Besides, it is unfair for FPFH in the comparison experiment to apply any algorithms which could possible reduce its performance.

The performances of the four registration algorithms are presented in Table 3.3. For ICP method, the *max correspondence distance* is set to be 5.0 m, which seems to be quite large,



(a) Keypoints detected by NARF for the 37th scan.

(b) Keypoints detected by SIFT for the 37th scan.

(c) Keypoints detected by NARF for the 41th scan.

(d) Keypoints detected by SIFT for the 41th scan.

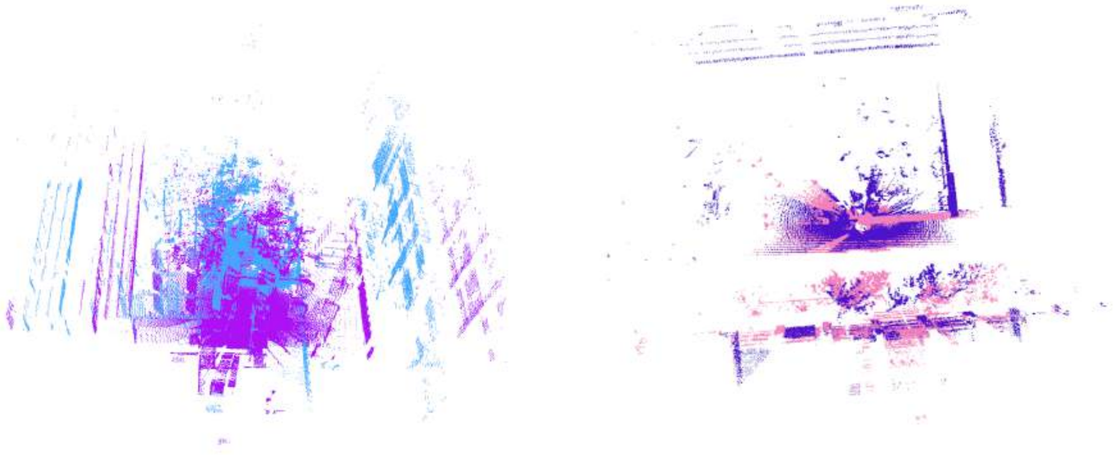
Fig. 3.15 Example of keypoints detected by NARF and 3D SIFT.

but this is determined by plenty of trials. Because we selected the scans by interval of 4, there are large offsets between scan pairs, small values of the *max correspondence distance* do not work, actually ICP fails all the scan pairs when this value is set to be 1.0 m. FPFH-based registration algorithm performs well for object registration, but works poor for outdoor scan registration tasks since there are too many similar features in large scene scans. With respect to the plane-based algorithm, it tells that it could not find enough planes in 25th & 21th and 37th & 33 scan pairs. But it gives wrong results for 29th & 25th and 49th & 45th scan pairs, as Figure 3.16 shows.

In order to compare the efficiency of the four registration algorithms, their runtime on the same personal desktop computer is depicted in Figure 3.17. As expected, the ICP and FPFH algorithms need too much more time than HTD-based and plane-based algorithms, and present large fluctuation. ICP and FPFH-based scan algorithms play well for surface alignment of handy objects, but they work really poor in outdoor robotic mapping applications.

Table 3.3 Performances of four registration algorithms applied to "Barcelona Robot Lab Dataset" (without resampling).

Scan Pairs	HTD	ICP	FPFH	plane-based
5 → 1	✓	✗	✓	✓
9 → 5	✓	✗	✓	✓
13 → 9	✓	✓	✗	✓
17 → 13	✓	✓	✓	✓
21 → 17	✗	✗	✗	✗
25 → 21	✓	✗	✗	✓
29 → 25	✓	✗	✗	✗
33 → 29	✓	✗	✓	✓
37 → 33	✓	✗	✗	✗
41 → 37	✗	✗	✗	✓
45 → 41	✓	✓	✗	✓
49 → 45	✓	✗	✗	✗
53 → 49	✓	✗	✗	✓
Success ratio	**0.846154**	0.230769	0.307692	0.692308



(a) Wrong registration result of 29th and 25th scan pair.

(b) Wrong registration result of 49th and 45th scan pair.

Fig. 3.16 Wrong registration results given by plane-based registration algorithm [163].

Admittedly, it is unfair for ICP to be compared in the scenarios where the scan pairs have large offsets, since it is a local registration algorithm and the key of its success is the good initial guess and close enough correspondences. But we adopt ICP in this comparison experiment because it is a quite famous registration method. The variation of runtime is a internal defect of ICP, and it usually needs more time when it could not find enough close correspondences. Maybe the low efficiency of FPFH-based algorithm is partially evocable by the absence of keypoints detection, but its combination with NARF or 3D SIFT also requires much more time than our HTD-based registration algorithm. And just like the ICP method, the variation of runtime is also a internal defect of FPFH-based algorithm. The plane-based registration method [163] is designed for the outdoor robotic mapping in the scenarios with plenty of planes. It performs quite well in this experiment and needs slightly more processing time than our HTD-based algorithm. The algorithm will stop if it could not find enough corresponding planes, which is helpful to save time for the failed instances. The plane dependence of this kind of algorithm is a internal defect, which will be presented in Section 3.5.3. Our HTD-based algorithm could give results in about two seconds for most scan pairs, and its runtime is quite stable.

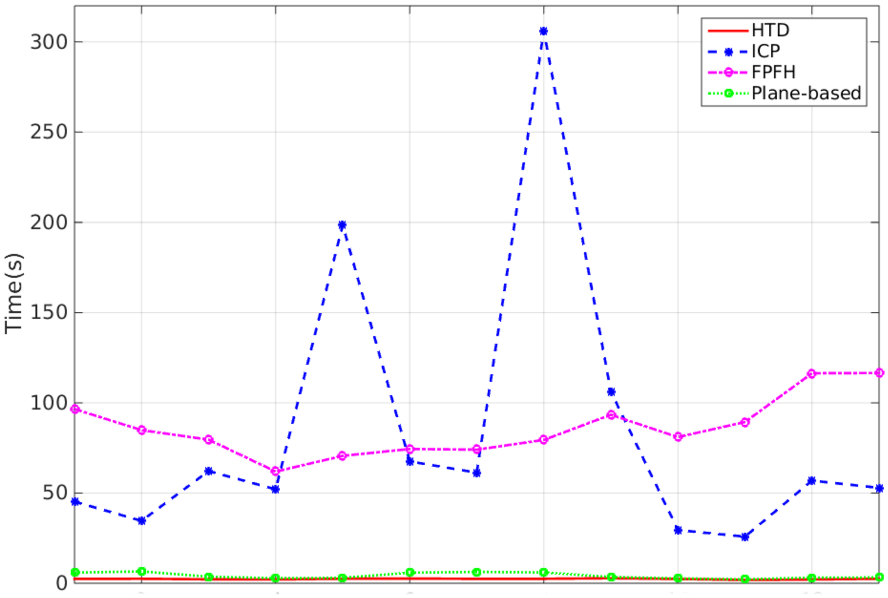
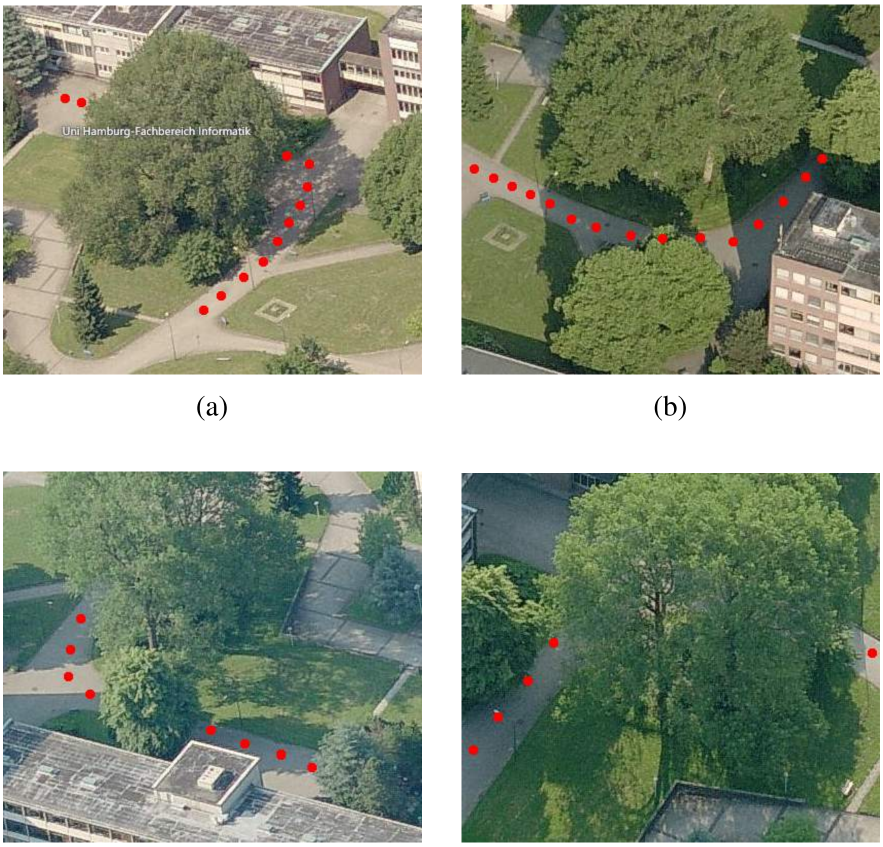


Fig. 3.17 Runtime of four registration algorithms applied to "Barcelona Robot Lab Dataset"(without resample)

3.5.3 The dataset captured by our custom-built platform



(a)

(b)

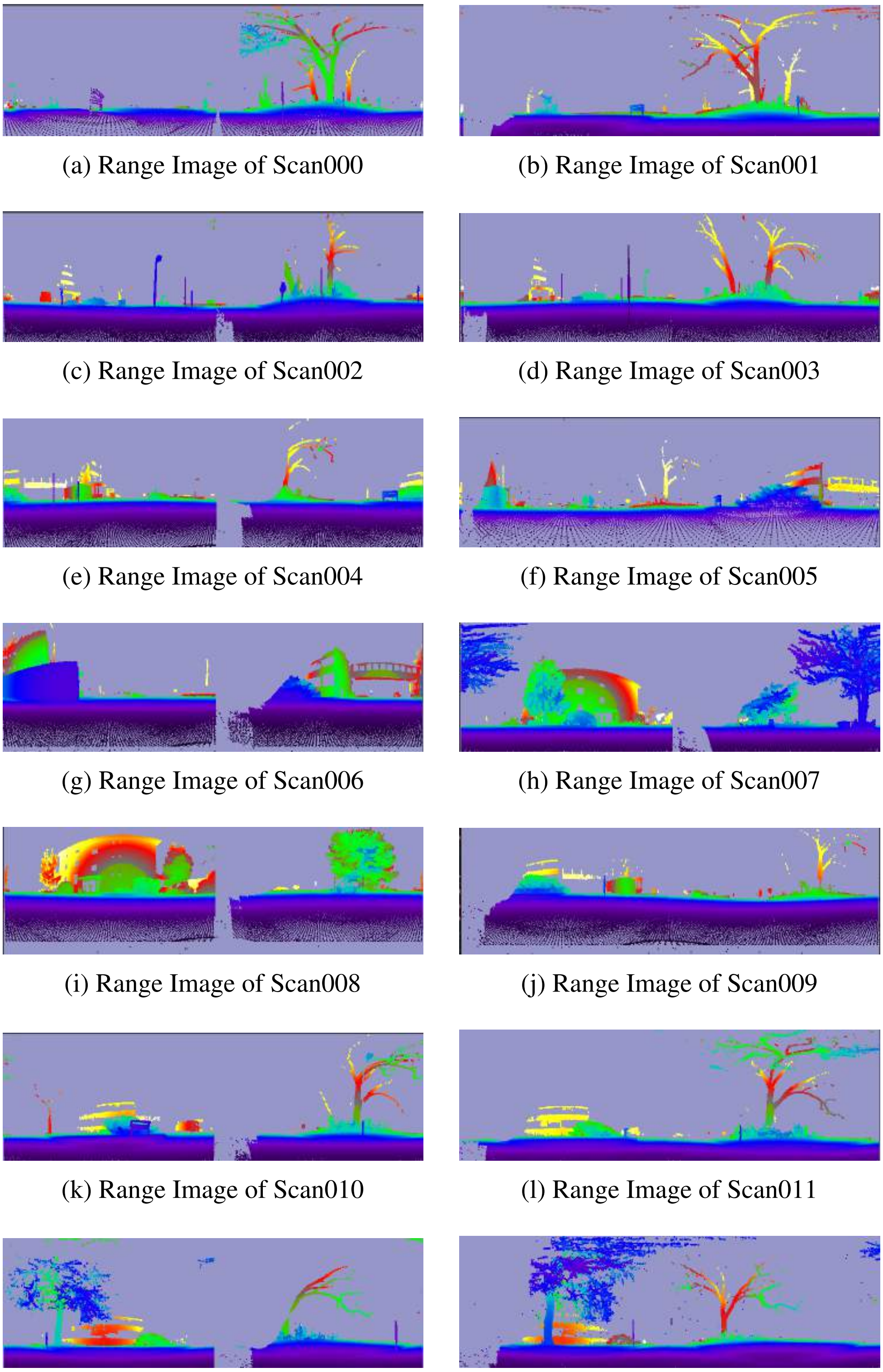
(c)

(d)

Fig. 3.18 Different views of location where our datasets are captured by our custom-built platform (From Bing Map).

We also validate our HTD-based scan registration method with respect to the dataset captured by our custom-built platform in University of Hamburg, Informatik campus. Each scan contains about 420000 points. Different views of location where the datasets are captured are shown in Figure 3.18. It is a pity that the pictures in Figure 3.18 are from "Bing Map" in summer, while the scans are captured in winter. The range images of scans are shown in Figures 3.19 and 3.20. The range images are generated by *pcl::RangeImageSpherical* Class and visualized by *pcl::visualization::RangeImageVisualizer* Class.

It can be seen that there are less planes in the scenes, so the plane-based registration method [163] performs poor. And compare with the dataset presented in Section 3.5.2, there are less offsets between successive scan pairs. In this way, the ICP method works better although still not good enough as our HTD-based method.



(a) Range Image of Scan000

(b) Range Image of Scan001

(c) Range Image of Scan002

(d) Range Image of Scan003

(e) Range Image of Scan004

(f) Range Image of Scan005

(g) Range Image of Scan006

(h) Range Image of Scan007

(i) Range Image of Scan008

(j) Range Image of Scan009

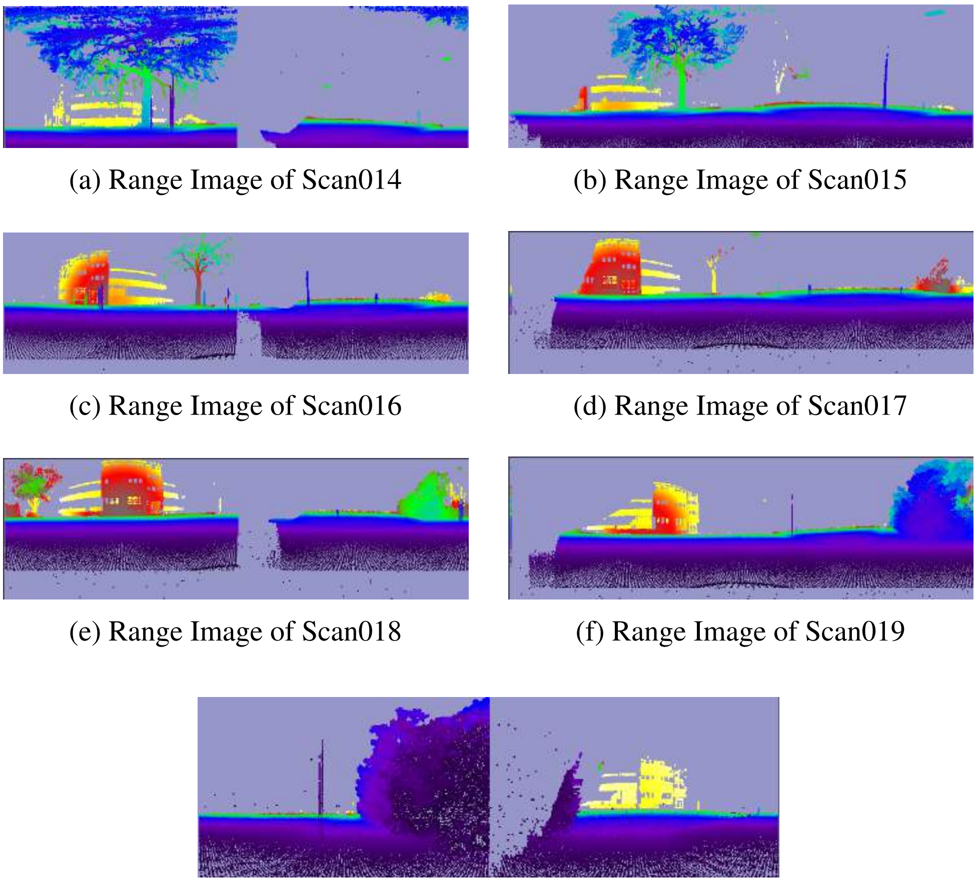
(k) Range Image of Scan010

(l) Range Image of Scan011

(m) Range Image of Scan012

(n) Range Image of Scan013

Fig. 3.19 Range Images of the scans captured by our custom-built platform (I)



(a) Range Image of Scan014

(b) Range Image of Scan015

(c) Range Image of Scan016

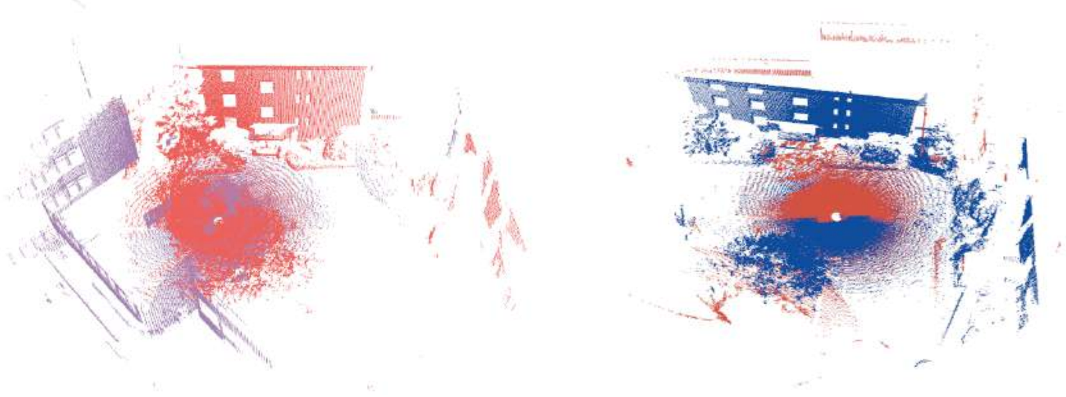
(d) Range Image of Scan017

(e) Range Image of Scan018

(f) Range Image of Scan019

(g) Range Image of Scan020

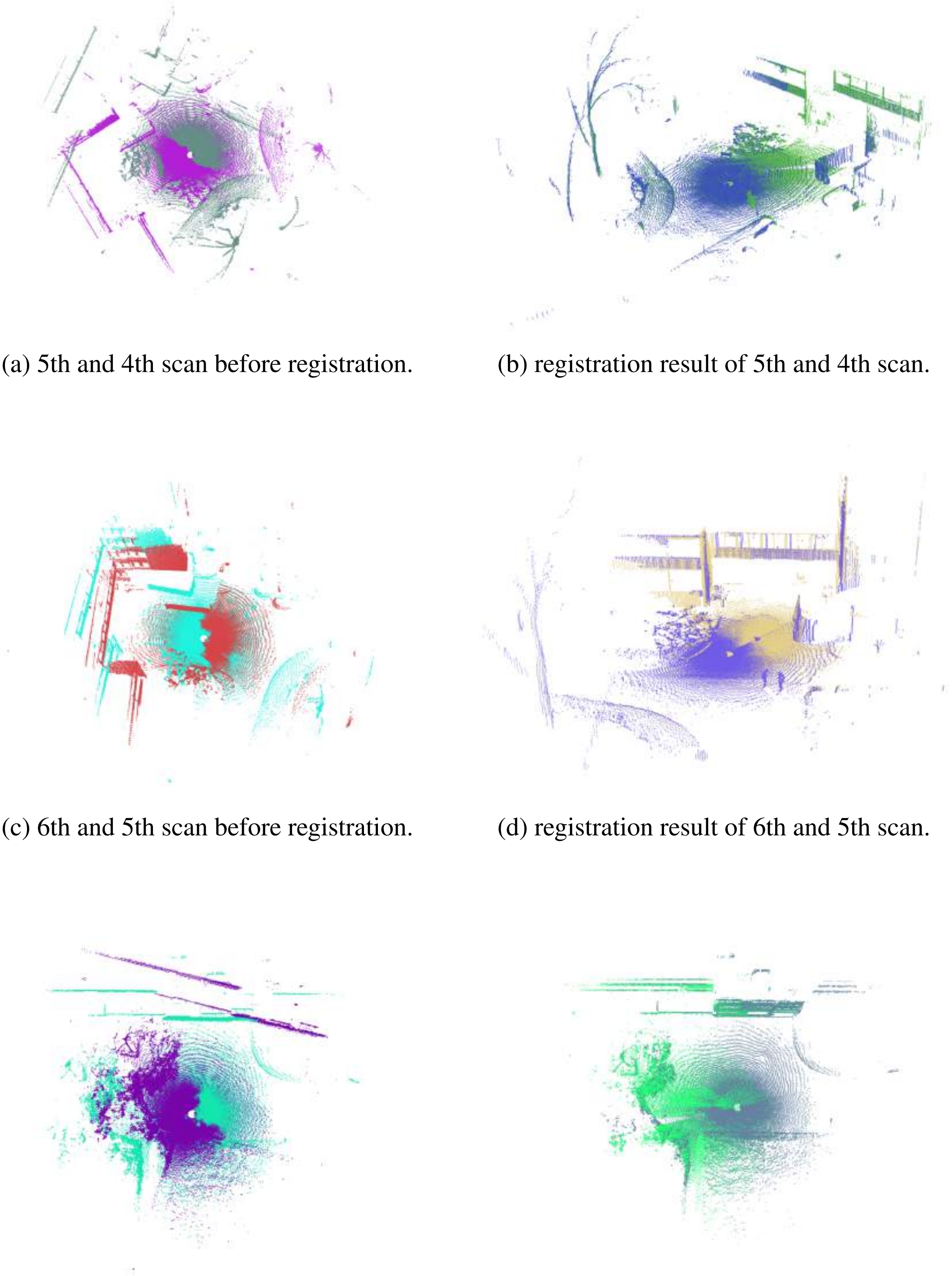
Fig. 3.20 Range Images of the scans captured by our custom-built platform (II)



(a) 7th and 6th scan before registration.

(b) 9th and 8th scan before registration.

Fig. 3.21 Two scan pairs captured by our custom-built platform which our HTD-based algorithm fails.



(a) 5th and 4th scan before registration.

(b) registration result of 5th and 4th scan.

(c) 6th and 5th scan before registration.

(d) registration result of 6th and 5th scan.

(e) 19th and 18th scan before registration.

(f) registration result of 19th and 18th scan.

Fig. 3.22 Examples of registration results of our HTD-based scan registration algorithm with respect to the datasets captured by our custom-built platform.

Performance of HTD-based algorithm

Table 3.4 Registration results of our HTD-based scan registration algorithm applied to our own dataset with resample: [0.1m, 0.1m, 0.1m]).

Scan Pairs	Rotation (radian)			Translation (m)		
	Roll	Pitch	Yaw	X	Y	Z
1 → 0	0.0140	0.0103	0.0260	-5.61854	-0.399192	0.699518
2 → 1	0.0357	-0.0137	0.4084	-5.93066	-2.48076	-0.0228423
3 → 2	0.0430	0.0697	0.0698	-5.30540	-0.139687	-0.0124463
4 → 3	0.0057	-0.0081	0.0533	-5.05042	0.0907421	-0.0522128
5 → 4	0.0247	-0.0270	0.9960	-4.42280	2.78186	0.0550576
6 → 5	0.0080	0.0138	0.0594	-4.16427	0.0646951	-0.0150615
7 → 6	—	—	—	—	—	—
8 → 7	0.0259	-0.0163	0.0474	9.47040	-0.566739	-0.152669
9 → 8	—	—	—	—	—	—
10 → 9	0.0376	0.0269	0.1718	-4.93201	-0.532898	0.00270474
11 → 10	0.0281	0.0010	0.2707	-4.7311	0.900582	-0.0392536
12 → 11	0.0565	-0.0174	0.0610	-4.37175	0.0409229	0.0506028
13 → 12	0.0083	-0.0009	0.0237	-4.74231	0.124113	-0.0172713
14 → 13	0.0129	0.0352	0.0657	-4.60302	0.0418059	-0.0402376
15 → 14	0.0023	0.0047	0.0187	-3.92494	-0.159744	-0.0582144
16 → 15	0.0035	-0.0423	0.0569	-4.34661	-0.201696	0.0865352
17 → 16	0.0009	0.0146	0.0033	-4.68631	-0.0810165	0.0109183
18 → 17	0.0143	0.0331	0.1023	-4.28723	-0.357048	-0.0580377
19 → 18	0.0164	-0.0322	0.2342	-5.19029	-1.23454	0.0449379
20 → 19	0.0787	0.0425	2.8318	-1.52083	-3.94592	-0.152487

The registration results by our HTD-based scan registration algorithm are proposed in Table 3.4. Our scan registration algorithm succeeds in 18 scan pairs, but fails when align 7th scan to 6th scan and 9th scan to 8th scan. These two failed scan pairs are shown in Figure 3.21. It can be seen that is is almost impossible to align these two scan pairs based on hardly any overlaps.

Three successful example scan pairs, 5th & 4th, 6th & 5th, 19th & 18th, are presented in Figure 3.22. The registration results are the results of our HTD-based registration algorithm without the refinement of any local registration methods. It can be seen that our registration method produces more precise registration results compared with other global registration methods.

Table 3.5 Processing time of our HTD-based scan registration algorithm applied to our own dataset with resample: [0.1m, 0.1m, 0.1m]).

Scan Pairs	Time per step (ms)				Total (s)
	Normal Estimation	HTD Calculation	Rotation Determination	Translation Recovery	
1 → 0	0.161178	0.046802	0.002685	0.161318	0.371983
2 → 1	0.15699	0.045501	0.009327	0.172119	0.383937
3 → 2	0.160137	0.047712	0.002578	0.177229	0.387656
4 → 3	0.169064	0.049445	0.002606	0.19017	0.411285
5 → 4	0.171743	0.050793	0.002841	0.180044	0.405421
6 → 5	0.189796	0.055884	0.009741	0.220836	0.476257
7 → 6 (FAIL)	0.305909	0.089268	0.002597	0.239586	0.63736
8 → 7	0.402581	0.117009	0.009449	0.2618	0.790839
9 → 8 (FAIL)	0.267917	0.077392	0.002599	0.259295	0.607203
10 → 9	0.144578	0.042653	0.002568	0.184877	0.374676
11 → 10	0.164629	0.048999	0.002576	0.169021	0.385225
12 → 11	0.188119	0.055871	0.002633	0.195521	0.442144
13 → 12	0.216315	0.061738	0.002596	0.179709	0.460358
14 → 13	0.225881	0.064858	0.002626	0.169989	0.463354
15 → 14	0.208199	0.060861	0.00905	0.181504	0.459614
16 → 15	0.181902	0.054074	0.002572	0.153685	0.392233
17 → 16	0.182583	0.054017	0.002752	0.164876	0.404228
18 → 17	0.205481	0.064176	0.002709	0.177487	0.449853
19 → 18	0.261451	0.076225	0.002658	0.18162	0.521954
20 → 19	0.217773	0.063314	0.002618	0.174134	0.457839

In this experiment, our HTD-based registration algorithm is also implemented on the personal desktop computer depicted in Section 3.5.2. With respect to the efficiency of the algorithm, the runtime of our algorithm is listed in Table 3.5. Before registration, the original scans are resampled with 0.1 meter leaf size. So the runtime in this experiment is much smaller than in the experiment described in Section 3.5.2. For most scan pairs, our HTD-based registration algorithm could give result in less than 0.5 seconds, and its runtime is quite stable.

Table 3.6 Performances of four registration algorithms applied to our own dataset with resample: [0.1m, 0.1m, 0.1m]).

Scan Pairs	HTD	ICP	FPFH	plane-based
$1 \to 0$	✓	✓	✓	✗
$2 \to 1$	✓	✓	✗	✗
$3 \to 2$	✓	✓	✗	✓
$4 \to 3$	✓	✓	✗	✓
$5 \to 4$	✓	✓	✗	✓
$6 \to 5$	✓	✗	✓	✗
$7 \to 6$	✗	✗	✗	✗
$8 \to 7$	✓	✓	✗	✓
$9 \to 8$	✗	✗	✗	✗
$10 \to 9$	✓	✓	✗	✗
$11 \to 10$	✓	✓	✓	✗
$12 \to 11$	✓	✓	✓	✗
$13 \to 12$	✓	✓	✓	✗
$14 \to 13$	✓	✗	✗	✗
$15 \to 14$	✓	✓	✗	✓
$16 \to 15$	✓	✓	✗	✗
$17 \to 16$	✓	✓	✓	✗
$18 \to 17$	✓	✗	✗	✗
$19 \to 18$	✓	✓	✗	✗
$20 \to 19$	✓	✗	✓	✗
Success ratio(%)	**90.0**	70.0	35.0	25.0

Comparison with the state-of-the-art

In order to prove the superiority of our registration algorithm, we also compare our HTD-based scan registration method with the state-of-the-art methods introduced in Section 3.5.1. For convenience of other researchers to compare with their own registration algorithm, the dataset and the code implementation our HTD-based registration method, ICP algorithm based on *pcl::IterativeClosestPointWithNormals* class and FPFH-based registration algorithm based on *pcl::FPFHEstimation* class in PCL are public available on the Internet, please refer to Appendix A. Again, FPFH algorithm combining the keypoints detection methods NARF and 3D SIFT is also accessible in our code, but it is not adopted in this comparison experiment. Instead, we use the uniformly resampled points as the keypoints, which takes more time but performs much better.

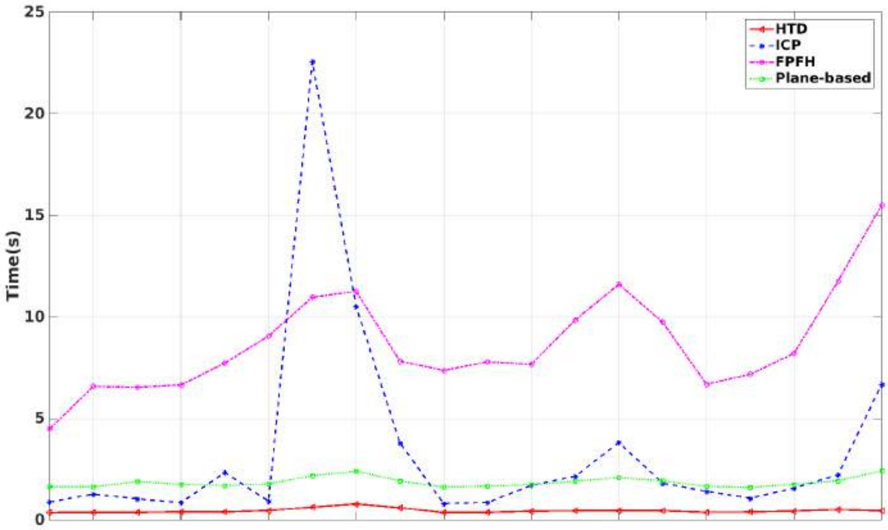


Fig. 3.23 Runtime of four registration algorithms applied to our dataset with resample: [0.1m, 0.1m, 0.1m]).

The performance of the four registration algorithms are presented in Table 3.6. For ICP method, the *max correspondence distance* is set to be 5.0 meter, which is determined by plenty of trials. After resampling the original point cloud with leaf size 0.1 meters, the FPFH-based algorithm resample the scan with leaf size 0.5 meters as the keypoints. ICP method performs better than in the experiment depicted in Section 3.5.2, since there are less offsets between successive scans. FPFH-based registration still works poor for large scale outdoor scenes as before. And the plane-based registration method does not perform as well as in the "Barcelona Robot Lab Dataset", since the scans in this experiment are not plane-rich. Actually, it is unfair for the plane-based registration method to be used in plane less scenarios, since it is designed for plane-rich scenarios.

In order to compare the efficiency of the four registration algorithms with respect to this dataset, their runtime on the same personal desktop computer is depicted in Figure 3.23. As before, the ICP and FPFH-based registration algorithms require too much time than our HTD-based technique, and show large variation. No matter succeed or not, the runtime of plane-based method [163] is less than ICP and FPFH-based registration algorithm. When the plane-based registration method could not find enough corresponding large planar patches in the scan pairs, it stops and tells that it could not work out the registration task. This property prevents the plane-based method wasting runtime. And just as in the past, our HTD-based algorithm needs least time in the four registration algorithms.

3.6 Summary

In this chapter, we propose the Hough Transform Descriptor of 3D point clouds, and develop a feature-less global registration method based on Hough Transform Descriptor. The novel registration method estimates rotation and translation information successively. The Hough Transform Descriptor only maintains the rotation information of the original data, so the rotation matrix is recovered by matching the corresponding Hough Transform Descriptor. After rotation recovery, rerotate the original scans according to the determined rotation matrix, and determine the translation by the POMF technique. The proposed registration method is validated with regards to two datasets, one is the "Barcelona Robot Lab Dataset" which is public available on the Internet, another one is the dataset captured by our custom-build platform in University of Hamburg. The experiment results show that our novel registration method present good performance and could handle the scan pairs with large offsets and partial overlaps. Furthermore, the runtime of our novel registration method is short and quite stable, which is important for practical application.

CHAPTER 4

Spherical Entropy Image

> To study and not think is a waste. To think and not study is dangerous.
>
> Confucius

Contents

4.1 Introduction

SHAPE descriptors collate the information stored in 3D surfaces so that the surfaces can be represented compactly and compared efficiently. Shape descriptors are the foundation of several advanced 3D data processing techniques, such as 3D object recognition, 3D shape retrieval and 3D surface alignment. The existing shape descriptors could be categorised into *Signature*-based and *Histogram*-based methods. *Signature*-based methods, such as Point Signature [32], 3D Point's Fingerprint [137] and local depth SIFT (LD-SIFT) [36], describe the 3D surface neighbourhood of a given point (hereinafter support) by encoding one or more geometric measurements computed individually at each point of a subset of the support. *Signature*-based methods are potentially highly descriptive thanks to the use of spatially well-localized information. *Histogram*-based methods, such as Spin Image [66][65], 3D Shape Context [51] and Fast Point Feature Histograms [119], describe the support by encoding counters of local geometrical entities into histograms according to a specific classification criteria. In broad terms, *Signature*-based methods are potentially highly descriptive thanks to the use of spatially well-localized information, whereas *Histogram*-based methods trade off descriptive power for robustness by compressing geometric structure into bins.

This chapter proposes a novel shape descriptor aiming to inherit the vantage of both *Signature*-based and *Histogram*-based methods, just like Signature of Histograms of OrienTations (SHOT) descriptor [150] [123] does. We propose a novel spherical descriptor named Spherical Entropy Image (SEI). Technically speaking, SEI is more compact than SHOT, since it calculates entropy based on histograms. Entropy preserves the robustness of histogram, and the compactness of entropy allows partitioning the support more densely to improve the descriptive power. SEI could be used as a global or local descriptor depending on either viewer-centred or object-centred coordinate system [41] is adopted. Viewer-centred representations describe surface data with respect to a coordinate system dependent on the viewer of the surface. Object-centred representations describe the object surface in a coordinate system fixed to the object. Incorporating with viewer-centred coordinate system, SEI could be used for 3D surface registration as a global descriptor. While integrating with object-centred coordinate system, SEI could be employed as a local shape descriptor for feature matching.

The shape descriptor SEI proposed in this chapter has the following characteristics:

✓ it is a hybrid structure between *Signature*-based and *Histogram*-based methods aiming at a favourable balance between descriptive power and robustness.

✓ as for the signature structure, it adopts a spherical grid that encompasses partitions along the azimuth and elevation axes, without partitions along radial axis.

✓ as for histograms, it encodes the depth of points into histograms, and further computes the entropy of histogram for the sake of compactness. Besides, entropy achieves robustness to point density variations.

✓ it is not dependent on the normals of points, since normal estimation is time consuming and not so stable as we expect.

4.2 Spherical Entropy Image (SEI)

4.2.1 Motivation

To begin with, we analyse the theoretical requirements of 3D shape descriptors. The purpose of shape descriptor is to describe the overall structure of the original data in a compact way and fulfil the following requirements:

✓ ***Structural discrimination:*** the descriptor should capture the structural information of the original data solely, which means different structures are supposed to result in different values of structural descriptors.

✓ ***Locality preservation:*** similar structures should be mapped to similar values, and in other words, slight changes in structure should lead to slight shifts in the values of structural representation. It is critical for the robustness to noise.

✓ ***Independent of point density:*** the density of points within the same structure may be variable in different scans. The shape descriptor should be related to permutation of points but independent of the density.

✓ ***Robust to outliers:*** notice that the outliers are the points far from the majority of 3D point clouds, and their portion is limited. So the value of structural representation should approximate 0 when the probability of its related points approach 0.

Generally, the *signature*-based shape descriptors have powerful structural discrimination, while the *histogram*-based shape descriptors have strong capability of locality preservation. So SHOT inherits both descriptive power of *signature*-based methods and robustness of *histogram*-based methods. It encodes information about the surface within a spherical support structure. This support is divided into 32 bins, with 8 divisions along the azimuth, 2 along the elevation and 2 along the radius. For every bin, a one-dimensional local histogram is computed. The local geometrical entity chosen for histogram is angle between the normal of the key-point and the current point within that bins. After all local histograms have been computed, they are stitched together into a final descriptor.

We continue the trend of SHOT and desire to preserve the vantage of *Signature*-based and *Histogram*-based methods. Rather than stitch histograms directly, we compute entropy of histogram, since entropy inherits the robustness of histogram but is more compact. Thanks to the entropy, we could divide the support more densely than SHOT. Further we notice that there must be only one point along the specific azimuth and elevation combination, so we discard the partition along radial direction and choose the depth of point as the local geometric entity for histogram. In this way, we could partition the azimuth and elevation very densely so as to achieve very strong descriptive power.

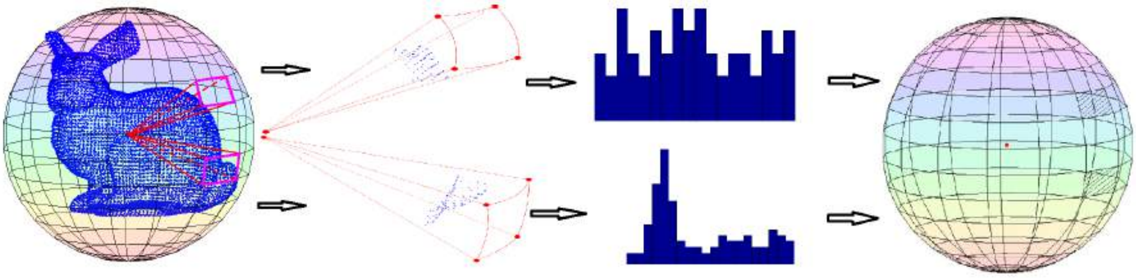


Fig. 4.1 Schematic illustration of Spherical Entropy Image (SEI). The calculation of SEI includes three steps: a) divide the scan/support into several pathes; b) build the histograms and estimate the probability density function (PDF); c) compute entropy based on the PDF densely.

4.2.2 Algorithm description

The original data is divided into bins by equally spaced boundaries in the azimuth and elevation dimensions. In this paper, the way of division is projecting the points into the spherical grids corresponding to regular sampling along the lines of longitude and latitude. As shown in Figure 4.1, the 3D patch is an analogous square pyramid taking the origin of the scan as the vertex, but the bottom surface is not a plane but a spherical grid [133].

The assessment of information content, prevalent in information theory discipline, measures up to the above principles related to 3D shape descriptor. The depth of points belonging to the same 3D patch could be interpreted as the observations of a random variable R, then the information entropy $E(R)$ of the variable R is regarded as the value of structural representation of the patch:

$$E(R) = -\sum_{d \in D} p(R=d) \cdot log\{p(R=d)\} \tag{4.1}$$

where D is the set of possible values of R, and p is its Probability Density Function (PDF).

The process of calculating entropy is illustrated in Figure 4.1:

1. divide the 3D point cloud into several patches according to azimuth and elevation angles of points;
2. for each patch, considering the depth of points as the observations of a random variable R, normalize the depth of points, build the histogram and compute the PDF of variable R;
3. compute the entropy of variable R based on the probability distribution.

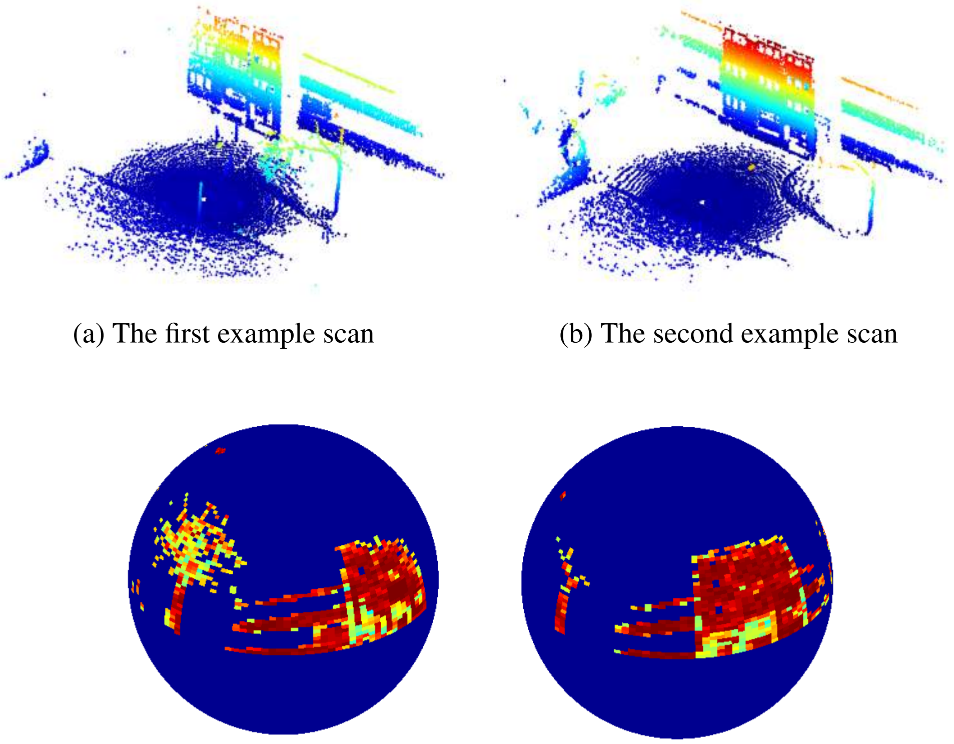
The structural representation of the 3D point cloud is achieved by computing the entropy of patches in a dense manner. We name this type of representation as Spherical Entropy Image (SEI). Two examples of SEI are presented in Figure 4.2.

4.2.3 Discussion

The SEI could be used as the global or local descriptor in many 3D data processing tasks, such as shape retrieve, object recognition and scan registration. There are several tips deserving more attention when compute the SEI.

Dimension of spherical grids

When acquire the SEIs of the scans, the dimension of the spherical grids is a tradeoff between precision and efficiency. The tradeoff is extremely common in engineering problems: greater precision requires more calculation and decreases the efficiency, while higher efficiency and less amount of calculation would lower the precision.



(a) The first example scan

(b) The second example scan

(c) SEI of the first scan

(d) SEI of the second scan

Fig. 4.2 Examples of Spherical Entropy Image (SEI) of scans captured by our custom-built platform

However, the dilemma of SEI is peculiar since it involves the entropy calculation. It is well known that the entropy is calculated on the probability of the variable, so the more samples the better the entropy describes the variable. If the number of points in a patch is too small, there would be no statistical property of the depth of points and hence the entropy is meaningless. Consequently, the dimension of the spherical grids could not be too high. Furthermore, when SEI is adopted in surface alignment as a global shape descriptor, it is only employed to estimate the initial estimation, and the crude alignment would be refined by local registration methods. From this point of view, the dimension of the spherical grids is unnecessary to be too high.

Entropy computation

Firstly, in order to make full use of the range of the histogram, we normalize the depth of points in the same patch. Secondly, the entropy is based on the statistics, which means sufficient samples are crucial to entropy calculation. But after the division of the scan/support, the points located in a single patch is numbered. To solve this predicament, we utilize the *kernel-based Parzen-window* [104] technique during building the histogram, since it yields more robust PDFs for a small number of observations. Thirdly, there are many other options for entropy calculation. Within the algorithm proposed in this paper, we regard the depth of point as a random variable R and estimate the entropy related to it. The variable R could also be interpreted as angle between the surface normals of points, the mutual distance of points or their admixture and so on.

Outlier removal

HTD proposed in Section 3.3 is a voting procedure, so the outlier removal is unnecessary for calculation HTD. While outlier removal is quite important for SEI, since the values of outliers have effect on the values of entropies. In our implementation, the PCL class *pcl::StatisticalOutlierRemoval* is employed to remove outliers, which uses point neighbourhood statistics to filter outlier data [120]. This outlier removal algorithm contains two steps: firstly compute the average distance between each point and its nearest k neighbours, and then determine a threshold based on the mean and standard deviation of all these distances; secondly iterate the points and remove the points whose average neighbour distance is above the threshold.

Translation dependence

The entropy calculation is translation-invariant, since it is merely related to the probability of the depth of points but independent of value of the depth. What is noteworthy is that even though the entropy evaluation is translation-invariant, the SEI is not translation-invariant due to the fact that the way to divide scans is translation dependent. It is because the sphere is divided into bins by equally spaced boundaries in the azimuth and elevation dimensions when compute SEIs. In this case, the object far away from the center of sphere covers less part of the spherical surface, while the object closer to the center occupies larger area. Thus when SEI is employed as a global shape descriptor, the *translation normalization* of the scan is essential before calculating SEI. Actually the translation normalization is theoretically incomplete, especially when the scans are partially overlapped. The theoretically complete solution is computing SEIs based on the spectral magnitudes of 3D Fast Fourier Transform (FFT) of scans. To reduce the adverse effect of translation dependence, it is necessary to compute 3D FFT of the original data and calculate the SEI based on the spectral magnitude of FFT. When SEI is used as a local shape descriptor, the translation dependence is not a problem since the local reference frame is estimated in this case.

4.3 SEI applied in scan registration

SEI could be employed to develop a global feature-less registration algorithm. Generally, the original 3D data is depicted by SEI and the transformation between the original data could be solved by matching the corresponding SEIs. This section proposes a global feature-less scan registration strategy based on the SEI and the Generalized Convolution Theorem. The 3D rotation is estimated by the Generalized Convolution Theorem based on Spherical Harmonic Transform of SEIs. After that, Phase Only Matched Filter (POMF) is adopted for translation recovery. No particular features in the input data are prerequisite to this scan registration method. Unlike the feature-based methods, the performance of this registration method does not reply on specific parameters.

4.3.1 Mathematical background

Although the reason why the Spherical Harmonic analysis techniques could be used to match patterns defined on 2D sphere has been explained in [40][63] detailedly, it is written by mathematicians and there are too many obscure mathematical equations which are difficult to understand by engineers. In this part, we express the profound principles of Spherical Harmonic analysis behind the obscure mathematical equations in plain language, explain the reason why the Spherical Harmonic analysis techniques could be used for matching patterns defined on 2D sphere.

Spherical Harmonic Transform (SHT)

Fourier Analysis is extremely significant in signal processing and pattern recognition, since it decomposes the function into a linear combination of sinusoidal basis functions. In other words, the Fourier Analysis maps a function to a set of coefficients of basis functions. Admittedly, there are infinite ways to decompose the signals, and the reason why sinusoids are adopted is that they are eigenfunctions of the *Laplacian operator*, hence they maintain fidelity to most real systems. The basis functions of traditional Fourier Analysis are induced by the Laplacian operator in Cartesian coordinate system. By the same token, the Laplacian operator also has effective forms in other coordinate systems, e.g., polar and spherical coordinates. *The SHT is connected with Cartesian Fourier Transform by the Laplacian operator.*

The angular part of spherical Laplacian operator's eigenfunctions are named *spherical harmonic* functions $Y_l^m : S^2 \mapsto \mathbb{C}$, where S^2 stands for the unit 2D sphere and $\mathbb{C}$ symbolizes the set of complex number.

$$Y_l^m(\vartheta,\varphi)=\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m(cos\vartheta)e^{im\varphi} \tag{4.2}$$

where (ϑ,φ) are the spherical coordinates; l, m are integers, $l>0, |m|<l$. The l is called the degree of spherical harmonics. For the l-th degree, there are $2l+1$ spherical harmonics basis functions indexed in the range of $-l\leq m\leq l$. P_l^m is associated Legendre polynomial, and $\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m(x)$ is called normalized associated Legendre polynomial. Some care must be taken in identifying the notational convention being used. In Y_l^m, ϑ is taken as the polar (colatitudinal) coordinate with $\vartheta\in[0,\pi]$, and φ as the azimuthal (longitudinal) coordinate with $\varphi\in[0,2\pi)$.

Similar to the Cartesian Fourier basis, spherical harmonics represent the different frequency components of spherical functions. In the following, we list some important properties of spherical harmonics $Y_l^m(\vartheta,\varphi)$, all of which are validated based on Mathematica [162].

Properties of Spherical Harmonics

1. $P_l^{-m}(x)=(-1)^m\frac{(l-m)!}{(l+m)!}P_l^m(x)$;
2. Based on Property 1, we could deduce:

$$Y_l^{-m}(\vartheta,\varphi)=(-1)^m\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m(cos\vartheta)e^{i(-m)\varphi} \tag{4.3}$$

3. Based on *super* Property 2 (equation 4.3) we could get:
   (a) $Y_l^{-m}(\vartheta,\varphi)=(-1)^m\overline{Y_l^m(\vartheta,\varphi)}$
   (b) $Real\{Y_l^{-m}(\vartheta,\varphi\}=(-1)^mReal\{Y_l^m(\vartheta,\varphi)\}$
   (c) $Imaginary\{Y_l^{-m}(\vartheta,\varphi\}=(-1)^{m+1}Imaginary\{Y_l^m(\vartheta,\varphi)\}$
   (d) $|Y_l^{-m}(\vartheta,\varphi_1)|=|Y_l^m(\vartheta,\varphi_2)|$

where $|Y_l^{-m}(\vartheta,\varphi_1)|=|Y_l^m(\vartheta,\varphi_2)|$ is an important property of spherical harmonic, which means that the complex modulus of Y_l^m is unrelated to the azimuth angles.

Let $L^2(S^2)$ denote the space of square integrate functions defined on sphere S^2. The spherical harmonics of degree l span a $(2l+1)$ dimensional subspace of $L^2(S^2)$. Spherical harmonics of different degrees are orthogonal to each other. Furthermore, the spherical harmonic functions provide an complete orthonormal basis for $L^2(S^2)$. In other words, any function $f(\vartheta,\varphi)\in L^2(S^2)$ could be expanded as a linear combination of spherical harmonics:

$$f(\vartheta,\varphi)=\sum_{l=0}^{\infty}\sum_{m=-l}^{l} f_l^m Y_l^m(\vartheta,\varphi) \tag{4.4}$$

$$f_l^m=\int_0^{2\pi} d\varphi \int_0^{\pi} d\vartheta sin\vartheta f(\vartheta,\varphi)\overline{Y_l^m(\vartheta,\varphi)} \tag{4.5}$$

where the overline stands for the complex conjugate. Equation (4.4) is named Spherical Harmonic expansion or inverse Spherical Harmonic Transform, and f_l^m is commonly called the spherical harmonic coefficients of $f(\vartheta,\varphi)$. Equation (4.5) is the Spherical Harmonic Transform of $f(\vartheta,\varphi)$. In Cartesian Fourier Transform, the translation of signal does not change the magnitude of Fourier coefficients. Integrating equation 4.4 with property 3d, we could obtain that the change of φ in function $f(\vartheta,\varphi)$ does NOT change the magnitude of Spherical Harmonic Coefficients.

It is obvious that the SHT has similar formula with the Cartesian Fourier Transform. Moreover, *the SHT is rotation friendly and plays a vital role in matching patterns on S^2.*

Generalized Convolution Theorem

It is well known that it is possible to detect the translated duplicates of a pattern in an image by convolving the image with the pattern. And this convolution could be converted to point-wise product via Fourier Transform. In other words, the convolution in time domain equals point-wise multiplication in frequency domain, then the original problem could be solved in frequency domain much more efficiently, just like POMF does. Furthermore, this convolution theorem could be generalized to the functions defined onS^2. *Efficient spherical convolution, aided by a fast Spherical Harmonic Transform and its inverse, contributes to the registration of graphs on S^2.*

Following the tradition, let $SO(3)$ denote the rotation group in 3D space, represented by the 3×3 matrices with determinant one. Given a function f_1 on the sphere, and its rotated version f_2 for a rotation $g\in SO(3)$: $f_2=\wedge(g)\cdot f_1$. Registration of the two functions could be achieved by correlating functions:

$$C(g)=\int_{S^2} f_2(\Omega)\bullet\overline{\wedge(g)\cdot f_1(\Omega)}d\Omega \tag{4.6}$$

and the g maximizing the integral (4.6) is the rotation between two functions. However, evaluating $C(g)$ for all possible rotations is a terrific time-consuming task.

Spherical Harmonic Transform could be adopted to determine the maximum g efficiently based on rotation-invariant characteristic of spherical harmonic:

$$\wedge(g)Y_l^m(\Omega) = \sum_{|k|\leq l} D_{km}^l(g)Y_l^k(\Omega) \tag{4.7}$$

where $D_{km}^l(g)$ are called *Wigner-D function*, and they are the irreducible unitary representations of $SO(3)$. In some sense, the $D_{km}^l(g)$ could be interpreted as the k-th component of $\wedge(g)$ acting on $Y_l^m(\Omega)$. Formula (4.7) signifies that *the rotated cousins of a spherical harmonic could be expressed as a linear combination of spherical harmonics with the same degree*. According to equation (4.4), the Spherical Harmonic expansions of f_1 and f_2 are:

$$f_1(\Omega) = \sum_l \sum_{|m|\leq l} a_l^m Y_l^m(\Omega) \tag{4.8}$$

$$f_2(\Omega) = \sum_{l'} \sum_{|m'|\leq l'} b_{l'}^{m'} Y_{l'}^{m'}(\Omega) \tag{4.9}$$

Substituting equations (4.8) and (4.9) into equation (4.6) and utilizing the *"Separation of Variables"* technique and the orthogonality between spherical harmonics, the correlation function could be rewritten as:

$$C(g) = \sum_l \sum_{|m|\leq l} \sum_{|m'|\leq l} a_l^m \overline{b_l^{m'}} \bullet \int_{S^2} Y_l^{m'}(\Omega)\ \overline{\wedge(g)Y_l^m(\Omega)} d\Omega \tag{4.10}$$

Recall the rotation-invariant property of the spherical harmonic expressed by equation (4.7), $\overline{\wedge(g)Y_l^m(\Omega)}$ could be replaced:

$$C(g) = \sum_l \sum_{|m|\leq l} \sum_{|m'|\leq l} a_l^m \overline{b_l^{m'}} \bullet \int_{S^2} Y_l^{m'}(\Omega)\ \overline{\sum_{|k|\leq l} D_{km}^l(g)Y_l^k(\Omega)} d\Omega \tag{4.11}$$

$$= \sum_l \sum_{|m|\leq l} \sum_{|m'|\leq l} a_l^m \overline{b_l^{m'}} \bullet \overline{\sum_{|k|\leq l} D_{km}^l(g)} \int_{S^2} Y_l^{m'}(\Omega)\ \overline{Y_l^k(\Omega)} d\Omega \tag{4.12}$$

All over again, based on the orthogonality of spherical harmonics, the integral and the summation on k in equation (4.12) could be zapped:

$$C(g) = \sum_l \sum_{|m|\leq l} \sum_{|m'|\leq l} a_l^m \overline{b_l^{m'}} \cdot \overline{D_{m'm}^l(g)} \tag{4.13}$$

The detailed deduction could be found in [40][63]. It is mainly on the strength of the *"Separation of Variables"* technique and orthogonality of spherical harmonics, and makes full use of the two criteria over and over again.

As prescribed in equation (4.13), the convolution in equation (4.6) is converted to point-wise multiplication, and the correlation $C(g)$ concerning the whole series of g could be evaluated uniformly and efficiently based on the spherical Harmonic coefficients. In this way, the rotation g maximizing $C(g)$ could be easily found. We use the General Convolution Theorem to estimate the rotation between two spherical representations of scan pairs in our algorithm.

Algorithm 5: Global feature-less scan registration method based on Spherical Entropy Image (SEI) as a global shape descriptor (based on 3D FFT).

```
Input  : Points Cloud $S_1, S_2$
Output : Transformation matrix $(R, T)$ between $S_1, S_2$
▷ Remove outliers of the original scans
$\mathbf{RS}_1 \leftarrow \mathrm{OutlierRemoval}(S_1)$ ;
$\mathbf{RS}_2 \leftarrow \mathrm{OutlierRemoval}(S_2)|$ ;
▷ Compute the magnitude of 3D FFT of scans
   to resist the translation dependence of SEI,
   since the SEI is dependent on the translation
$\mathbf{MF}_1 \leftarrow |\mathrm{3DFFT}(\mathrm{RS}_1)|$ ;
$\mathbf{MF}_2 \leftarrow |\mathrm{3DFFT}(\mathrm{RS}_2)|$ ;
▷ Calculate the Spherical Entropy Images(SEI)
$\mathbf{SEI}_1 \leftarrow \mathrm{SEICreation}(\mathrm{MF}_1)$ ;
$\mathbf{SEI}_2 \leftarrow \mathrm{SEICreation}(\mathrm{MF}_2)$ ;
▷ Estimate the Spherical Harmonic coefficients of SEIs
$\mathbf{SH}_1 \leftarrow \mathrm{SphericalHarmonic}(\mathrm{SEI}_1)$ ;
$\mathbf{SH}_1 \leftarrow \mathrm{SphericalHarmonic}(\mathrm{SEI}_2)$ ;
▷ Recover the rotation via Generalized Convolution Theorem [63]
$\mathbf{R} \leftarrow \mathrm{GeneralizedConvolutionTheorem}(\mathrm{SH}_1, \mathrm{SH}_2)$ ;
▷ Rotate the original scans according to $\mathbf{R}$
$\mathbf{S}'_1, \mathbf{S}'_2 \leftarrow \mathbf{R}(S_1, S_2)$ ;
▷ Rasterize the point clouds into xy planes and z axis
$P_1, v_1 \leftarrow \mathbf{Rasterize}(S'_1)$ ;
$P_2, v_2 \leftarrow \mathbf{Rasterize}(S'_2)$ ;
▷ Adopt the POMF to determine the translation
$t_x, t_y \leftarrow \mathbf{2D\_POMF}(P_1, P_2)$ ;
$t_z \leftarrow \mathbf{1D\_POMF}(v_1, v_2)$ ;
$\mathbf{T} = (t_x, t_y, t_z)$ ;
```

4.3.2 Scan registration algorithm based on SEI

Obtained the SEIs of two scans, the rotation between the original scans could be estimated by aligning their corresponding SEIs. The estimation of rotation between spherical graphs could be decoupled into determination of rotation axis and rotation angle according to *Euler's Rotation Theorem*. The rotation axis could be recovered by minimizing the 3D Angular Difference Function (ADF) defined in [71], but the performance of this method deteriorates in the case that the scans have interference and occlusion. A little trick is invented in [134] to employ the ground surface to determine the rotation axis, but this technique only validates in robotic mapping scenarios where the ground surface is available.

The technique for pattern matching defined on S^2 through Generalized Convolution Theorem is originated by the researchers in applied mathematics area [40][63], but is attracting more and more attention in pattern recognition [70][96] and medical image processing societies [129][86]. We apply this technique to recover the rotation in our algorithm.

Overall, the rotation recovery of scans contains four steps:

1. remove outliers and then acquire the SEIs of two scans;
2. implement SHT of two SEIs;
3. combine the Spherical Harmonic coefficients and calculate the correlation functions;
4. find the maximum value of the correlation functions.

We use the freely available toolkit SOFT [64] to obtain the Spherical Harmonic coefficients.

Once the rotation is determined correctly, there is only translation between rerotated scans. The POMF techniques presented in Section 3.4.2 is employed to estimate the shift between two rerotated scans.

Resistance to translation dependence

As discussed in Section 4.2.3, the SEIs are dependent on translation. How to reduce the adverse effect of translation when determine the rotation matrix is the key of success of SEI-based registration algorithm.

One solution is to render the point clouds into volumes and calculate the 3D FFT of volumes, since the magnitude of 3D FFT is translation-invariant. The summary of this kind of global feature-less scan registration strategy based on SEI is shown in Algorithm 5. This solution is theoretically complete but computational expensive. The code implementing this solution is available on the Internet, please refer to Appendix A. Unfortunately, this solution needs more runtime and its performance is not satisfied, at least not so well as the second solution.

The second solution is to borrow the iteration idea. Essentially, the rotation determination and the translation recovery in SEI-based registration algorithm is a **Chicken-and-Egg** problem. To recover the translation, it is necessary to determine the rotation firstly. But in order to determine the rotation matrix accurately, the translation rectification is helpful. This is a typical Chicken-and-Egg problem. We adopt the iteration idea to solve this dilemma. And an example is presented in Figure 4.3. Naturally, sometime it is unnecessary to iterate for some pairs, and commonly two times iteration is enough. The procedure of the global feature-less scan registration strategy based on SEI and iteration technique is summarized in Algorithm 6.

4.4 SEI applied as a local shape descriptor

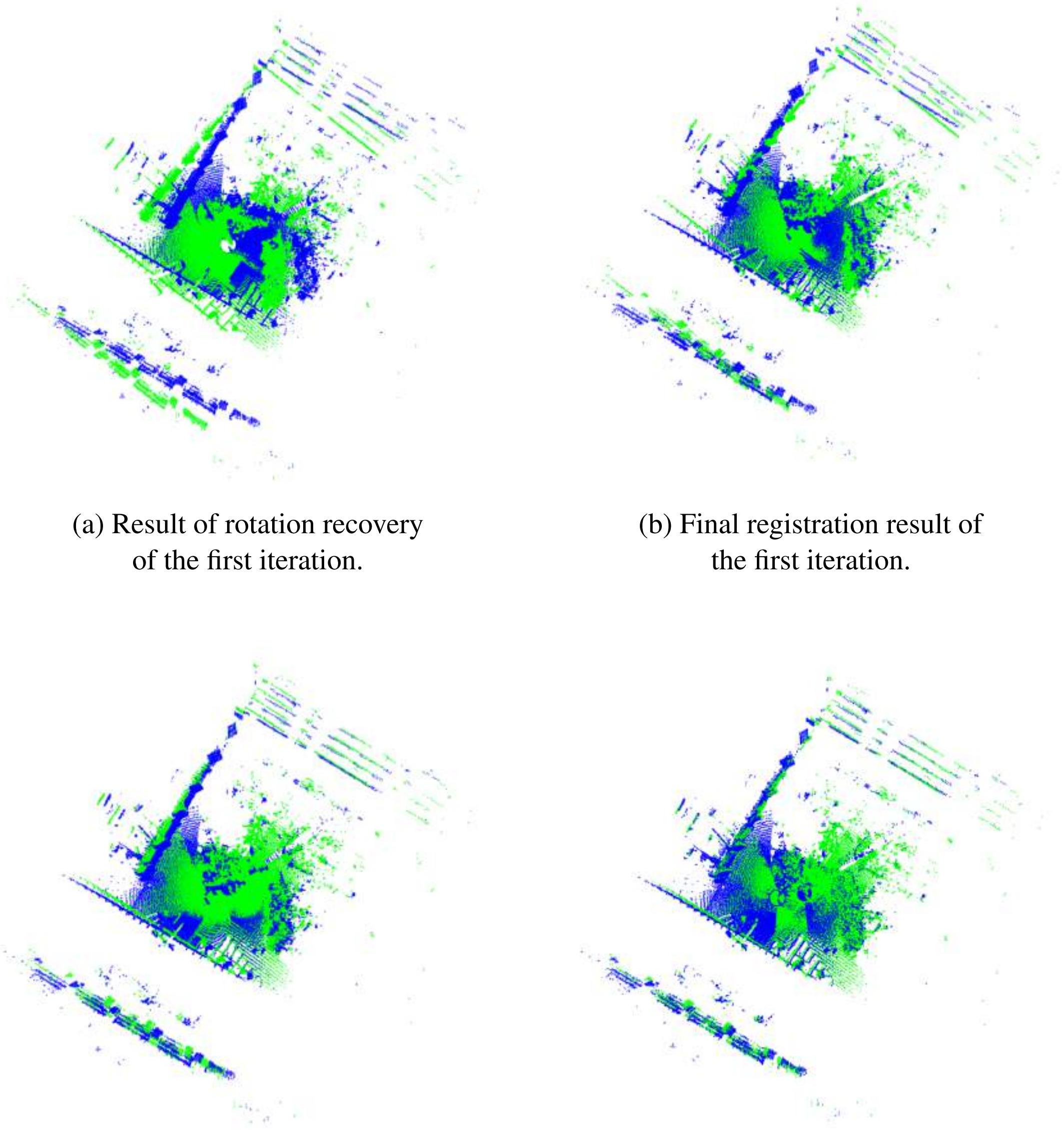
Beside being used as a global shape descriptor in scan registration, SEI could be used as a local shape descriptor in 3D object recognition. But in order to employ SEI as a local shape descriptor, it is necessary to achieve the rotation-invariant version of SEI. SEI combining with the repeatable local coordinate system proposed in [150][18] could be used as local shape descriptor in 3D model-based object recognition task. Besides that, the property of SHT of spherical functions could be used to obtain the rotation-invariant version of SEI.

4.4.1 Rotation-invariant version of SEI

Rotation-invariant SEI based on local reference frame

Just like all the other signature-based shape descriptors, SEI requires the repeatable Local Reference Frame (LRF) when it is used as a local shape descriptor. The building of LRF is challenging since it should be invariant to translations, rotations and robust to noise and clutter. But it is significant to the performance of shape descriptors, which is firstly investigated in [150].

With regard to the LRF of the point clouds, Principle Component Analysis (PCA) [67] is traditionally the first choice to achieve this. PCA is mathematically defined as orthogonal linear transformation which transforms the data into a new coordinates system such that the



(a) Result of rotation recovery of the first iteration.

(b) Final registration result of the first iteration.

(c) Result of rotation recovery of the second iteration.

(d) Final rotation result of the second iteration.

Fig. 4.3 Examples to prove effectiveness of the iteration framework adopted by our SEI-based registration algorithm employing 393th and 392th scans in "Barcelona Robot Lab Dataset".

greatest variance by some projection of the data comes to lie on the first coordinate called the first principal component; the second greatest variance on the second coordinate, and so on. PCA can be thought as revealing the internal structure of the data in a way that best explains the variance in the data. And PCA is usually implemented through Singular Value Decomposition (SVD) of the covariance matrix of point coordinates within the support of

Algorithm 6: Global feature-less scan registration method based on Spherical Entropy Image (SEI) as a global shape descriptor (based on iteration).

```
Input  : Points Cloud $S_1, S_2$
Output : The transformed $S_1$ aligned with $S_2$
▷ Remove outliers of the original scans
$\mathbf{S}_1 \leftarrow$ OutlierRemoval($S_1$) ;
$\mathbf{S}_2 \leftarrow$ OutlierRemoval($S_2$) ;
▷ Calculate Spherical Entropy Images(SEI) of scene
$\mathbf{SEI}_2 \leftarrow$ SEICreation($\mathrm{S}_2$) ;
▷ Estimate Spherical Harmonic coefficients of scene
$\mathbf{SH}_2 \leftarrow$ SphericalHarmonic($\mathrm{SEI}_2$) ;
▷ Rasterize the scene into xy planes and z axis
$\mathbf{P}_2, \mathbf{v}_2 \leftarrow$ Rasterize($S_2$) ;
▷ Iteration Part
for Iteration: 0 to $n$ do
    ▷ Calculate SEI of object
    $\mathbf{SEI}_1 \leftarrow$ SEICreation($\mathrm{S}_1$) ;
    ▷ Estimate Spherical Harmonic coefficients of object
    $\mathbf{SH}_1 \leftarrow$ SphericalHarmonic($\mathrm{SEI}_1$) ;
    ▷ Recover the rotation via Generalized Convolution Theorem [63]
    $\mathbf{R} \leftarrow$ GeneralizedConvolutionTheorem($\mathrm{SH}_1, \mathrm{SH}_2$) ;
    ▷ Rotate the original scan according to $\mathbf{R}$
    $\mathbf{S_1} \leftarrow \mathbf{R}(S_1)$ ;
    ▷ Rasterize the object scan into xy planes and z axis
    $\mathbf{P}_1, \mathbf{v}_1 \leftarrow$ Rasterize($S_1$) ;
    ▷ Adopt the POMF to determine the translation
    $\mathbf{t}_x, \mathbf{t}_y \leftarrow$ 2D_POMF($P_1, P_2$) ;
    $\mathbf{t}_z \leftarrow$ 1D_POMF($v_1, v_2$) ;
    $\mathbf{T} = (t_x, t_y, t_z)$ ;
    ▷ Translate the object scan
    $\mathbf{S_1} \leftarrow \mathbf{T}(S_1)$ ;
end
```

query point. The main problem with the LRF based on PCA is that the signs of coordinate system are not determinate. The PCA itself provides no means for assessing the sign of each axis.

A repeatable, unambiguous and unique LRF is proposed in [150]. It addresses sign ambiguity using the technique presented in [18]. As Figure 4.4 sketches, the direction of axis is determined so as it has same direction with majority of the data vectors, so the black real line in Figure 4.4 is selected as the coordinate axis, while the dashed line is discarded. We adopt this kind of LRF when use SEI as a local shape descriptor, so we recall its procedure briefly as follows:

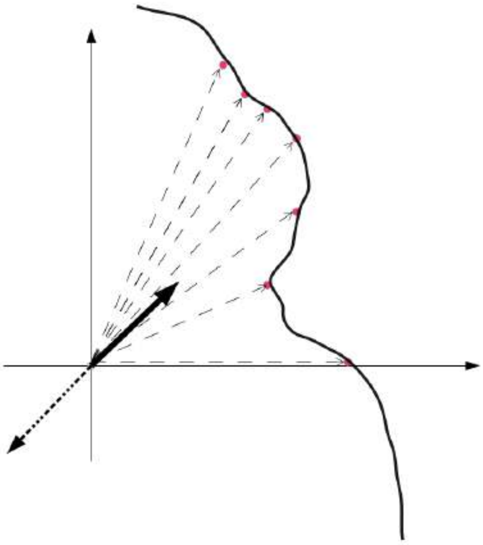


Fig. 4.4 The sketch showing how to select the direction of local reference frame.

1. compute the covariance matrix M as a weighted linear combination based on all points laying within the spherical support of radius R:

$$M = \frac{1}{\sum_{i:d_i \leq R}(R-d_i)} \sum_{i:d_i \leq R} (R-d_i)(p_i-p)(p_i-p)^T$$

   where p is the feature point and $d_i = \| p_i - p \|_2$;

2. calculate the eigenvectors of M and refer to the three eigenvectors in decreasing eigenvalue order as the x^+, y^+, z^+ axis respectively, and denote their opposite vectors as x^-, y^-, z^-;

3. the sign of x axis is determined as: compute the inner products of point vectors and x^+/x^- and count the number of positive inner products respectively. If x^+ has more positive inner products than x^-, choose x^+ as the x axis and vice versa. The sign of z axis is disambiguated as x axis, finally the y axis is obtained as $z \times x$.

In the standard Cartesian coordinate system, the azimuth θ and polar ϕ could be easily computed: $\theta = tan^{-1}\left(\frac{y}{x}\right)$, $\phi = cos^{-1}\left(\frac{z}{r}\right)$. While for the new LRF, the azimuth and polar angles have to be calculated in a more general way:

A. project the point p_i onto the xy plane of the LRF, compute the vector $\overrightarrow{p_i}$ from origin to projection and normalize it;

B. compute the angle between the normalized $\overrightarrow{P}$ and the x axis to achieve azimuth θ;

C. calculate the angle between the vector from origin to point and z axis to achieve polar ϕ.

To conclude, the summary of computing SEIs as a local shape descriptor based on the local reference frame is depicted in Algorithm 7.

Rotation-invariant SEI based on Spherical Harmonics

As discussed in Section 4.3.1, a key property of SHT is that a rotation in the azimuthal direction results a phase shift in the frequency domain, so the amplitudes of the harmonics coefficients are invariant to the rotations along the azimuth direction. The rotation-invariant of SEI could be achieved based on this important property of Spherical Harmonic Transform.

Algorithm 7: Compute rotation-invariant Spherical Entropy Images (SEIs) on the keypoints of point clouds combining the local reference frame

```
Input  : Point cloud $\mathbf{S}$ and $n$ keypoints: $(\mathrm{KP}_1, \mathrm{KP}_2, \mathrm{KP}_3, \ldots, \mathrm{KP}_n)$
Output : $n$ SEIs: $(\mathrm{SEI}_1, \mathrm{SEI}_2, \mathrm{SEI}_3, \ldots, \mathrm{SEI}_n)$
for $KP_i$: $i \leftarrow 1$ to $n$ do
    ▷ acquire the support of $KP_i$
    $\mathbf{KPS}_i \leftarrow$ SearchForNeighbor($\mathbf{S}, \mathrm{KP}_i, Radius$);
    ▷ estimate the local reference frame(LRF)
    $\mathbf{LRF}_i \leftarrow$ EstimateLRF($\mathrm{KPS}_i$);
    ▷ compute the azimuth and elevation of points in LRF
    $\mathbf{KPS}_i^{'} \leftarrow$ ComputeSphericalCoordinate($\mathrm{KP}_i, \mathrm{KPS}_i, \mathrm{LRF}_i$);
    ▷ divide the points into m patches according
      to azimuth and elevation of point
    $(\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_m) \leftarrow$ Divide($\mathrm{KPS}_i^{'}$);
    for $P_j$: $j \leftarrow 1$ to $m$ do
        ▷ normalize the distances
        $\mathbf{V}_j \leftarrow$ FormulateVector($\mathrm{P}_j$);
        $\mathbf{V}_j^{'} \leftarrow$ Normalize($\mathbf{V}_j$);
        ▷ build the histogram
        $\mathbf{H}_j \leftarrow$ Histogram($\mathbf{V}_j^{'}$);
        ▷ smooth histogram using Parzen-window technique [104]
        $\mathbf{H}_j^{'} \leftarrow$ ParzenWindow($\mathbf{H}_j$);
        ▷ calculate the entropy
        $\mathbf{SEI}_i[j] \leftarrow$ Entropy($\mathbf{H}_j^{'}$);
    end
end
```

Algorithm 8: Compute rotation-invariant Spherical Entropy Images (SEIs) on the keypoints of point clouds based on Spherical Harmonic Transform (SHT)

```
Input   : Point cloud $\mathbf{S}$, $n$ keypoints: $(\mathrm{KP}_1, \mathrm{KP}_2, \mathrm{KP}_3, \ldots, \mathrm{KP}_n)$
          and bandwidth $B$
Output  : $n$ SEIs: $(\mathrm{SEI}_1, \mathrm{SEI}_2, \mathrm{SEI}_3, \ldots, \mathrm{SEI}_n)$
for $KP_i$: $i \leftarrow 1$ to $n$ do
    ▷ acquire the support of $KP_i$
    $\mathbf{KPS}_i \leftarrow$ SearchForNeighbor($\mathbf{S}, \mathrm{KP}_i, Radius$);
    ▷ estimate the normal of point $KP_i$
    $\mathbf{N}_i \leftarrow$ NormalEstimation($\mathrm{KP}_i$);
    ▷ build local reference frame (LRF)
    $z_i \leftarrow \mathbf{N}_i$ ;
    $x_i \leftarrow$ RandomNormalVector ;
    $y_i \leftarrow x_i \times z_i$ ;
    $\mathbf{LRF}_i \leftarrow (x_i, y_i, z_i)$;
    ▷ compute the azimuth and elevation of points in LRF
    $\mathbf{KPS}_i^{'} \leftarrow$ ComputeSphericalCoordinate($\mathrm{KP}_i, \mathrm{KPS}_i, \mathrm{LRF}_i$);
    ▷ divide the points into m patches according
      to azimuth and elevation of point
    $(\mathbf{P}_1, \mathbf{P}_2, \ldots, \mathbf{P}_m) \leftarrow$ Divide($\mathrm{KPS}_i^{'}$);
    for $P_j$: $j \leftarrow 1$ to $m$ do
        ▷ normalize the distances
        $\mathbf{V}_j \leftarrow$ FormulateVector($\mathrm{P}_j$);
        $\mathbf{V}_j^{'} \leftarrow$ Normalize($\mathbf{V}_j$);
        ▷ build the histogram
        $\mathbf{H}_j \leftarrow$ Histogram($\mathbf{V}_j^{'}$);
        ▷ smooth histogram using Parzen-window technique [104]
        $\mathbf{H}_j^{'} \leftarrow$ ParzenWindow($\mathbf{H}_j$);
        ▷ calculate the entropy
        $\mathbf{TempSEI}_i[j] \leftarrow$ Entropy($\mathbf{H}_j^{'}$);
    end
    ▷ compute the Spherical Harmonic Coefficients of $TempSEI_i$
    $\mathbf{SHC}_i \leftarrow$ SHT($\mathrm{TempSEI}_i, B$) ;
    $\mathbf{SEI}_i \leftarrow$ Amplitude(SHC)$_i$ ;
end
```

In order to employ this property of SHT to obtain rotation-invariant version of SEI, the z axis should be determined since the amplitudes of the harmonics coefficients are only invariant to the rotation along the azimuth direction. In our implementation, the normals of the query points are regarded as the z axis, and the *pcl::NormalEstimation* is adopted to compute the normals of points [107]. The normal estimation in PCL is based on PCA of a covariance matrix created from the nearest neighbours of the query point, and the eigenvector according to the smallest eigenvalue is regarded as the normal of the query point. And PCL employs a priori knowledge to select the orientation of the normal: all of points are captured from a single viewpoint, so PCL orients all normals towards to the viewpoint.

Computation of SEIs as a local shape descriptor based on the Spherical Harmonics is outlined in Algorithm 8. For any **real** function, $\|\mathbf{SHC}_l^m\| = \|\mathbf{SHC}_l^{-m}\|$, so we only store the spherical harmonic coefficients of $m \geq 0$. The final size of rotation-invariant version of SEI is $B \cdot (B+1)/2$. It is interesting that the dimensionality of the final feature is NOT related to the number of azimuth and elevation division.

4.4.2 3D object recognition based on SEI

3D object recognition aims to correctly identify objects in a scene and estimate their poses. It is a challenging task in complex scenes in the presence of clutter and occlusions. As for the 3D model-based object recognition, the 3D models of objects are constructed offline and stored in a model library. Commonly the 3D model of object is constructed by acquiring its range images from multiple viewpoints and registering them in a common coordinate system. In the online recognition stage, the range image of scene is converted into the similar representation as objects in model library and matched with the models in order to recognize the objects. The key and main challenge in 3D model-based object recognition is the effective shape descriptors and matching of correspondences between objects and scenes, especially when there is significant degree of occlusion and clutter in the scenes. Generalized Hough Transform technique [148][151], which is robust to wrong correspondences and occlusions, is commonly employed to find the objects in 3D object recognition task.

Generalized Hough Transform

The Generalized Hough Transform (GHT) introduced by Dana H.Ballard in 1981 [8] is the modification of the traditional Hough Transform aided by the principle of template matching, and aims to solve the dimensionality curse of the traditional Hough Transform. GHT extends the traditional Hough Transform to detection of arbitrary shapes by voting each feature for a specific position, orientation and scale factor of the shape. The principle supporting GHT is the truth that the problem of finding the model in the scene can be solved by calculating the position of model in the scene based on the feature correspondences between model and scene. So intrinsically, GHT is a template matching procedure accumulating the evidence of positions of models in the scene.

Algorithm 9: 3D model-based object recognition method using Spherical Entropy Image (SEI) as a local shape descriptor

```
Input  : Model database $M_1, M_2, M_3, \dots, M_n$ and scene $S$
Output : $n$ dimensional vector $E$ indicating the presence of models in the $S$, and their pose in
         scene $(R_1, t_1), (R_2, t_2), (R_3, t_3), \dots, (R_n, t_n)$
for $M_i$: $i \leftarrow 1$ to $n$ do
    ▷ downsample the point clouds uniformly to extract keypoints
    MKP$_i$ ← UniformDownsample(M$_i$);
    ▷ compute Spherical Entropy Images for keypoints
    MLRF$_i$ ← CalculateLocalReferenceFrame(M$_i$, MKP$_i$);
    MSEI$_i$ ← CalculateSEI(M$_i$, MKP$_i$, MLRF$_i$);
end
▷ downsample the point cloud uniformly to extract keypoints
SKP ← UniformDownsample(S);
▷ compute Spherical Entropy Images for keypoints
SLRF ← CalculateLocalReferenceFrame(S, SKP);
SSEI ← CalculateSEI(S, SKP, SLRF);
for $M_i$: $i \leftarrow 1$ to $n$ do
    ▷ find correspondences between model and scene
    FeaturePairs ← KdTree(MSEI$_i$, SSEI);
    ▷ correspondences cluttering based on Hough voting
    HoughPeak ← HoughVoting(FeatureParis);
    if $HoughPeak > threshold$ then
        $E_i = 1$;
        ▷ estimate the pose of object based on
         Absolute Orientation Algorithm
        $(R_i, t_i)$ ← AbsoluteOrientation(FeaturePairs, HoughPeak);
    else
        $E_i = 0$;
        $(R_i, t_i)$ = NaN;
    end
end
```

GHT converts the problem of finding model's position to an issue of finding the transformation parameters which maps the model to scene. Thus normally the Hough Space stores the transformation parameters instead of the shape parameters in GHT techniques. The position of the model in the scene could be settled as long as the transformation parameters are estimated. In spite of that, when GHT is extended into 3D domain to deal with generic rotations and translations, the Hough Space is 6-dimensional, which leads to a high computational cost of the voting process.

Most recent GHT designed for 3D object recognition under occlusion and clutter is proposed in [148][151], and only one 3D Hough space is built. The 3D Hough space stores the 3D position of the reference point of model indicated by features. The key of this kind of GHT is the employment of Local Reference Frame (LRF) related to the features and rigidity of models. Since the coordinates of the reference point under the LRFs related to the corresponding features are identical for rigid models, the coordinates of the reference point in the global reference frame of scene could be calculated known the relationship between LRFs and global reference frame. Specifically, for a feature in the model F_i^M, the coordinate of the reference point C_i^M in the LRF of F_i^M could be achieved if the relationship between the global reference frame of model and LRF of F_i^M were known. If the correspondence of F_i^M in scene F_j^S could be found, then the coordinate of the reference point C_j^S in the LRF of F_j^S is equal to the coordinate of the reference point in the LRF of F_i^M. Aided by the relationship between the global reference frame of scene and LRF of F_j^S, the coordinate of the reference point in global reference frame of scene could be achieved. And a 3D Hough space is built and the reference point in global reference frame of scene is stored in Hough space. The peaks of the Hough space indicate the presence of the models, and then the coherent correspondences related to the peaks are used to compute the pose of models in scene. The key to comprehend the procedure is the familiarity of transformation of reference frame. The 3D model-based object recognition algorithm using SEI as a local shape descriptor is outlined in Algorithm 9.

4.5 Experiments about SEI-based registration algorithm

In this section, the Spherical Entropy Image (SEI) is applied in scan registration, and compared with the existing and representative scan registration algorithms based on the public available dataset and the data captured by our custom-built platform. The reason why we adopt the public available dataset is that it is convenient for other researchers to compare our algorithm with their own algorithms.

Experiments have been carried on standard personal desktop computers. Our SEI-based feature-less scan registration algorithm is implemented in C++, and the code has been published online, please refer to Appendix A. The Point Cloud Library (PCL) [107] is utilized for reading and writing point clouds and 3D visualization. The linear algebra library Eigen [53], a C++ template library for linear algebra, is employed for matrix processing. The Discrete Fourier Transform library FFTW [50] is adopted in translation recovery procedure to compute the forward and backward Fourier Transform. The SOFT Package [64] is employed to obtain the Spherical Harmonic coefficients of SEIs. The *std::chrono::high_resolution_clock* class in **C++ 11** is employed to record the runtime.

This SEI-based registration algorithm involves much mathematical staff, so in section 4.5.1 we do some simulation to prove the theoretical completeness of this method. And then in section 4.5.2, two scans captured by our custom-built platform, which are quite challenging for registration algorithms, are used to prove the effectiveness of our algorithm. In section 4.5.3, we introduce the experiment results based on the selected scans of "Barcelona Robot Lab Dataset". We introduce the "Dragon" dataset in "Stanford 3D Scanning Repository", and propose the experiment results based on this dataset in section 4.5.4. Due to the space limitation, we cannot present all the results. Indeed, some typical results has been selected for explanation.

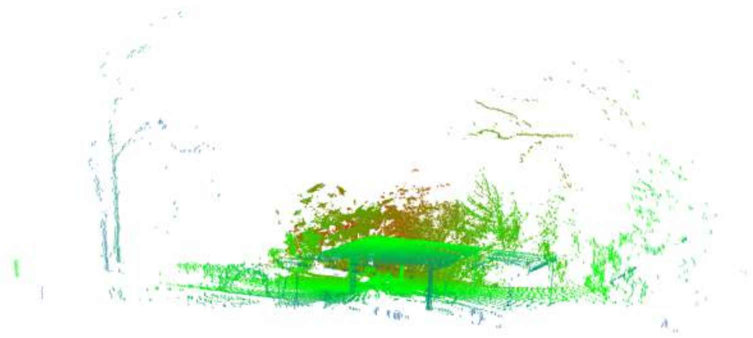


Fig. 4.5 A scan captured by our custom-built platform.

4.5.1 Simulation

We will show in the following experiments that our SEI-based registration algorithm works well for real data, but in this part, the goal is to validate the theoretical completeness of the algorithm in controlled circumstances. A scan captured by our custom-built platform, depicted in Figure 4.5, is used in this set of simulations to prove the theoretical completeness of SEI-based scan registration algorithm. The scan is rotated around x, y, z axis by 45, 60, 90, 125, 180, 225 degrees respectively and translated by the random distance in $[0, 10]$ meter. In this simulations, the dimension of spherical grids when calculate the SEIs is 64×64 and the bandwidth of SHT is 32. The SEI-based registration algorithm succeeds in the all 18 instances with no exceptions.

Figure 4.6 presents the correlation result of original scan and the scan rotated around y axis by 90 degree as an example. The joint of the slices in Figure 4.6 is the main peak of the correlation function. The main peaks in correlation functions are head and shoulders above others. The joint of the slices are the main peak.

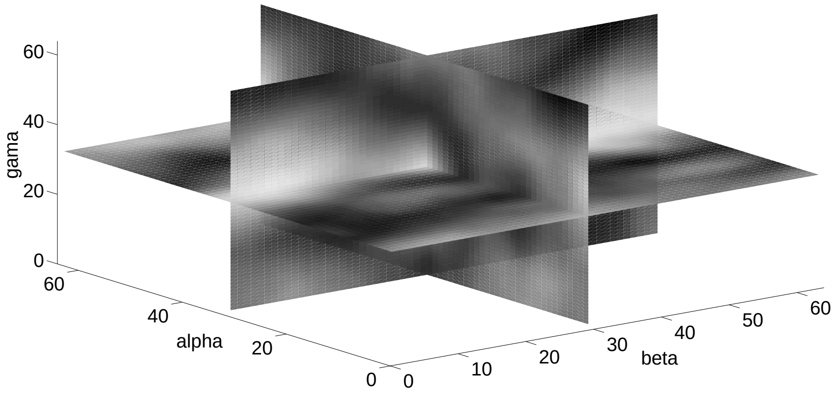
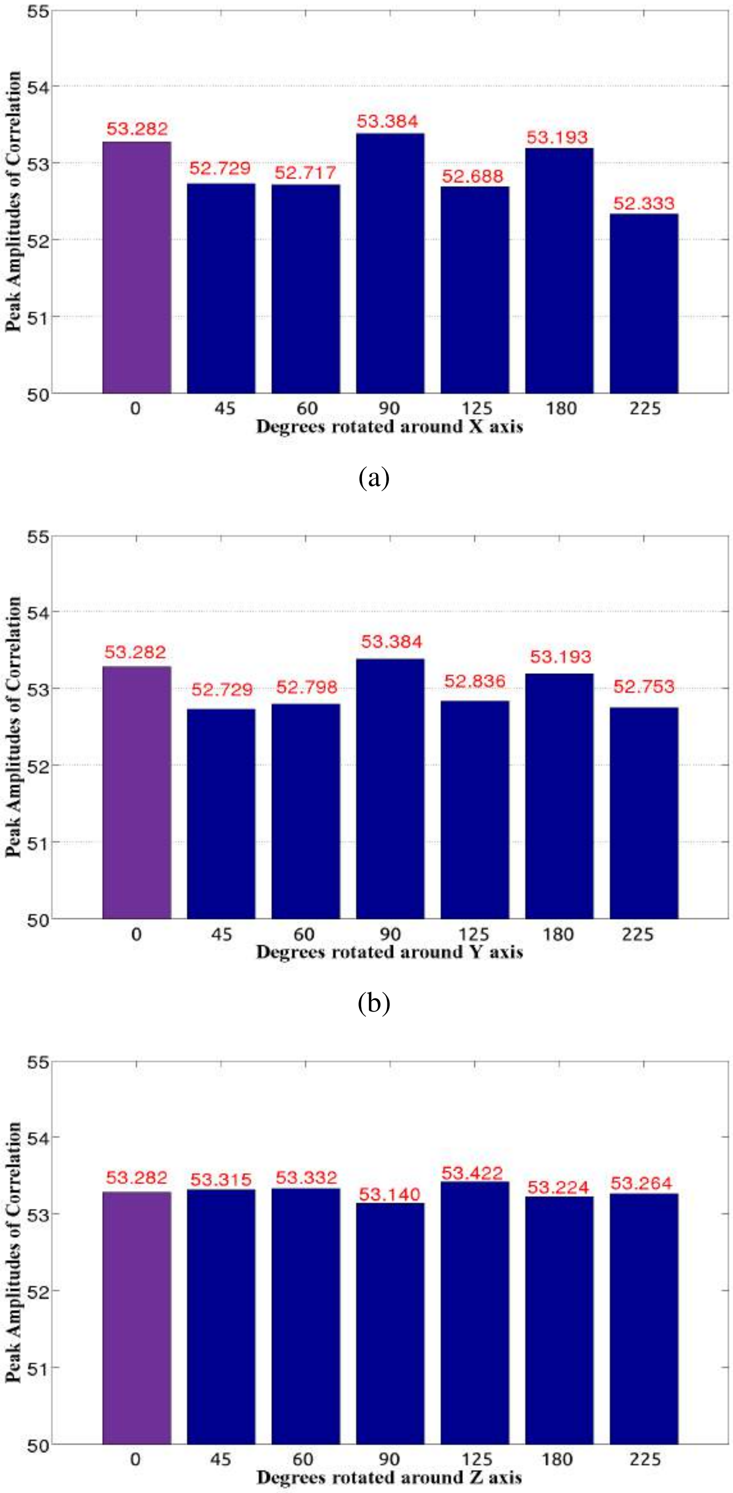


Fig. 4.6 Spherical correlation result of original scan and the scan rotated around y axis by 90 degree. We represent the rotation by the Euler Angle and the sequence of rotation axis is (z,y,z). *alpha* correspond to rotation angle around y, *beta* correspond to the first angle around z, and *gama* correspond to the second angle around z. The joint of the slices is the main peak.

Figure 4.7 depicts the peak amplitudes of the correlation results relative to that of the auto-correlation. The main peak amplitudes in the convolution procedure manifest scarcely any degeneration. If have to say, the peak amplitudes of correlation between the scan and its duplicates rotated around z axis are greater than that of x and y axis. Meanwhile, we found that the recovery of rotation around z axis is also more reliable than x and y in practical applications. This is because the samples of spherical function spread the whole range of longitudes but partial range of latitudes (exclude the poles) while compute the SHT. However this is not a limitation for robotic mapping applications, where there tend to be mainly large changes in yaw and little changes in roll and pitch between scans. This simulation proves that our algorithm performs well as regards arbitrarily large rotation without noise, in other words, the method is complete in theory.

4.5.2 Data captured by our custom-built platform

This experiment uses the scans captured by our custom-built platform while it turns around. The maximum range of the sensor is 30 meters. The whole platform is low-cost and the scans are very noisy since they are capture in rainy weather. The two scans presented in Figure 4.8a are challenging to align but typical in practical application. Registration of scans captured by the robots when they turn around at the end of trajectory or go around a corner is the real challenge in robotic mapping scenarios. Just like the scans depicted in Figure 4.8a, the



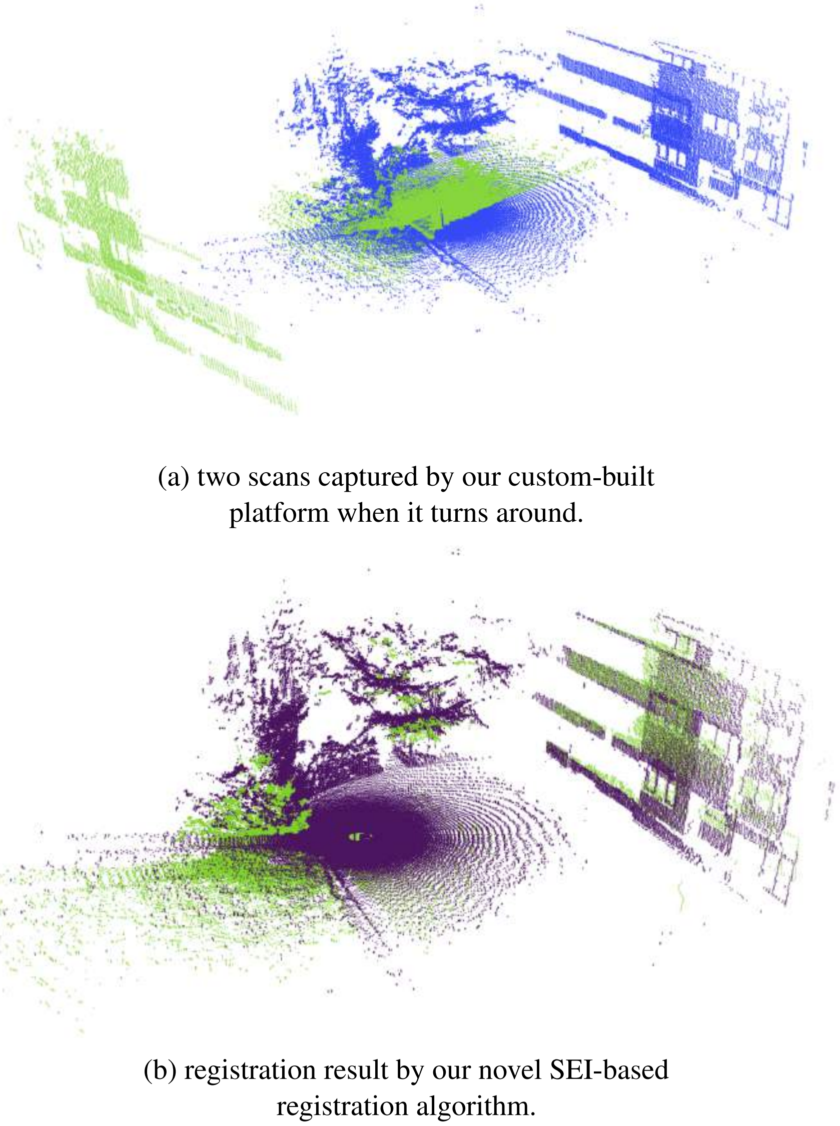
(a)

(b)

(c)

Fig. 4.7 The peak amplitudes of correlation in simulation, while the correlation result rotated by 0 degree is the auto-correlation result.

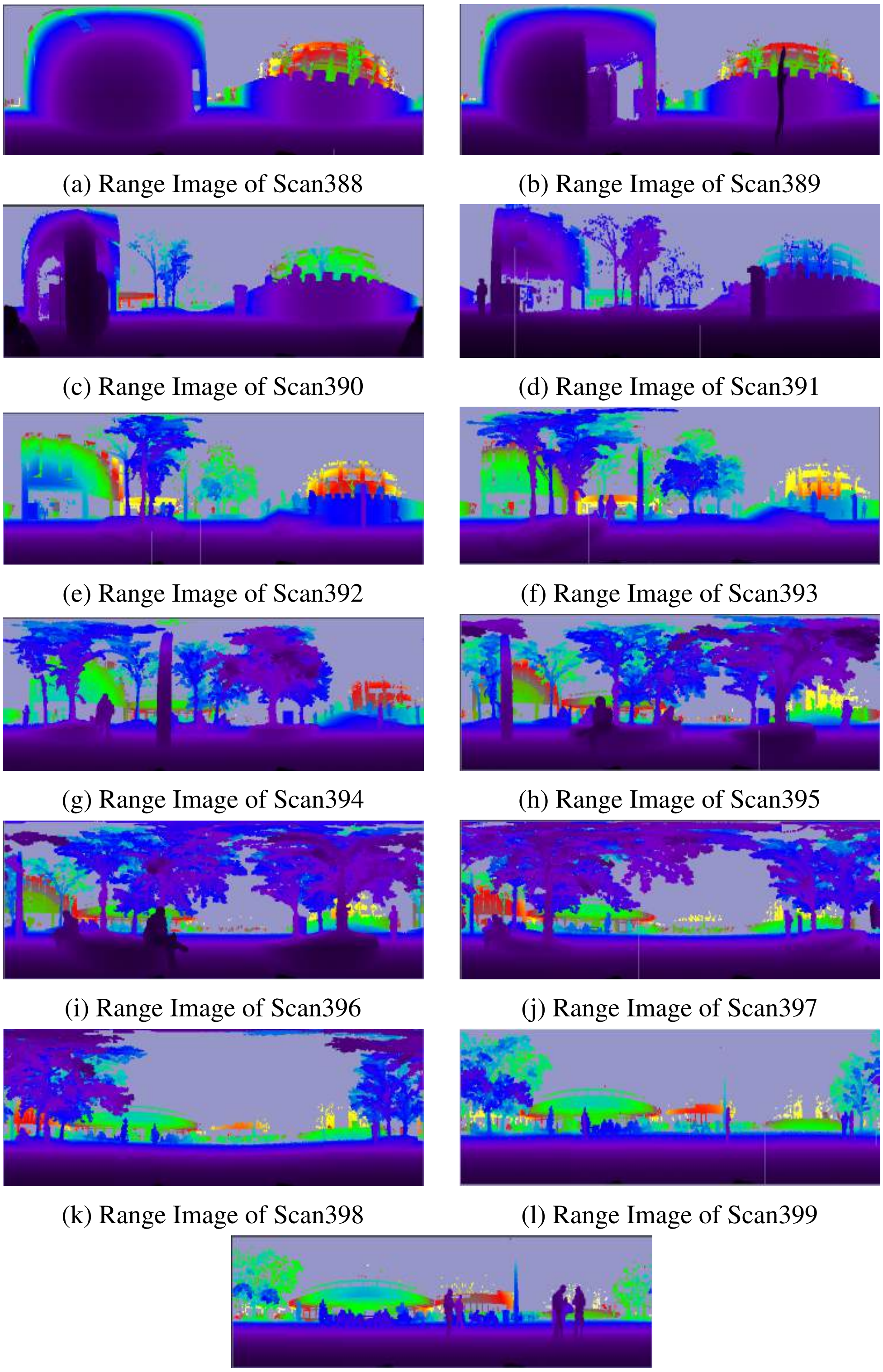
scans captured at the corners often tend to behave large offsets in yaw. However, as to these cases, the existing registration methods often fail. We apply the ICP, 3D-NDT, FPFH-based registration method in Point Cloud Library [107] and the latest plane-based method [163] to align the two scans in Figure 4.8a. All of them break down in this scenario. The registration result of our novel technique is presented in Figure 4.8b, the dimension of spherical grids when calculate the SEIs is 64×64 and the bandwidth of SHT is 32. The result shows that our novel method could handle scans with very large rotation in practice. This is a quite exciting result for the outdoor mobile robots aiming to build the maps of their work environments.



(a) two scans captured by our custom-built platform when it turns around.

(b) registration result by our novel SEI-based registration algorithm.

Fig. 4.8 Experiments involving two scans captured by our custom-built platform when it turns around.



(a) Range Image of Scan388

(b) Range Image of Scan389

(c) Range Image of Scan390

(d) Range Image of Scan391

(e) Range Image of Scan392

(f) Range Image of Scan393

(g) Range Image of Scan394

(h) Range Image of Scan395

(i) Range Image of Scan396

(j) Range Image of Scan397

(k) Range Image of Scan398

(l) Range Image of Scan399

(m) Range Image of Scan400

Fig. 4.9 Range Images of the selected scans in "Barcelona Robot Lab Dataset".

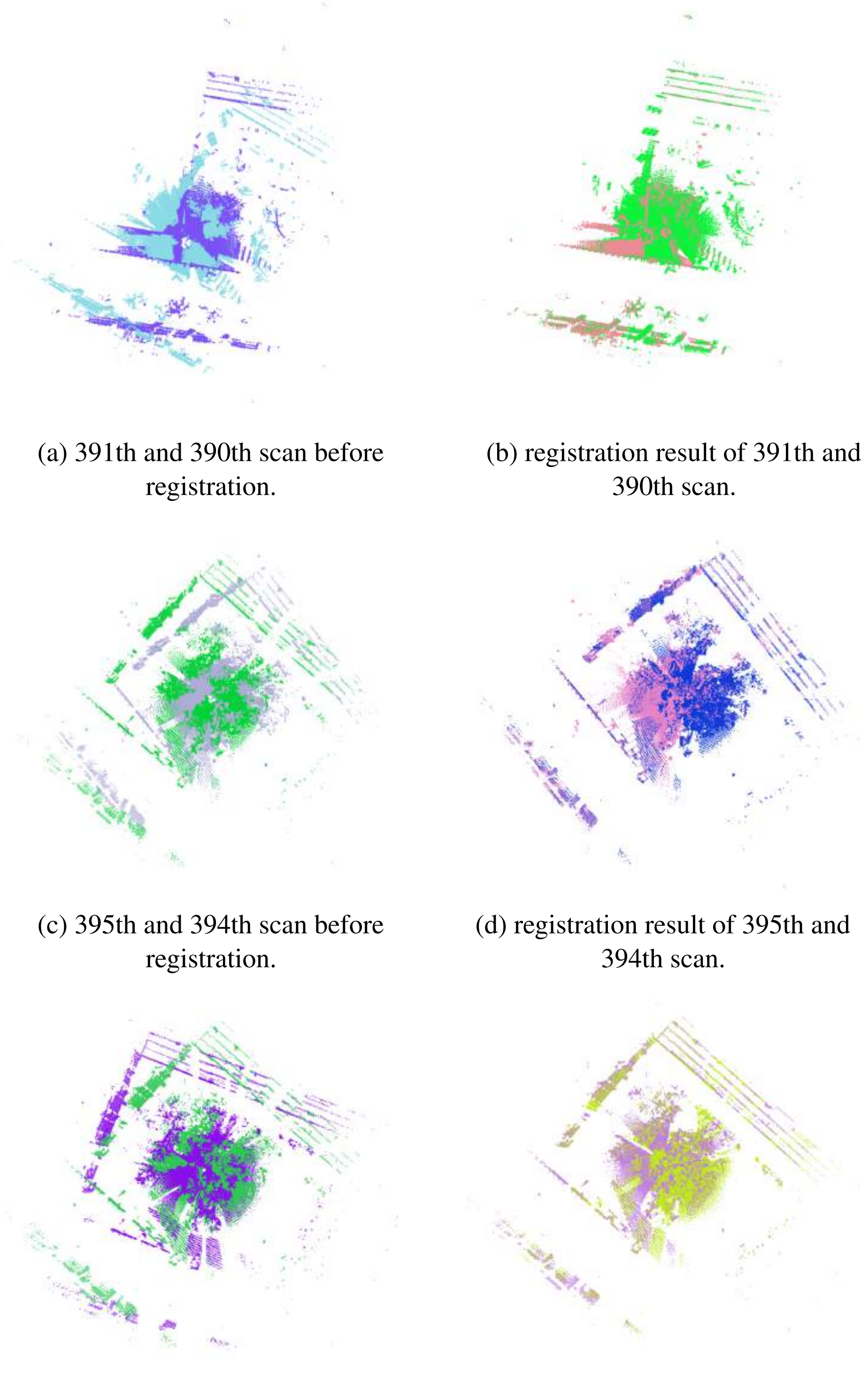
4.5.3 The "Barcelona Robot Lab Dataset"

We also employ the publicly available "Barcelona Robot Lab Dataset" to validate the effectiveness of our novel feature-less SEI-based registration algorithm. In this experiment, we use the last 13 scans in the dataset. The principle of selecting scans is that there should be large offsets between successive scans both in rotation and translation, since only this kind of scans could examine the effectiveness of both rotation determination and translation recovery of registration algorithm.

The range images of the selected 13 scans are depicted in Figure 4.9. As before, the range images are generated by *pcl::RangeImageSpherical* Class and visualized by *pcl::visualization::RangeImageVisualizer* Class.

Table 4.1 Registration results of our SEI-based scan registration algorithm applied to "Barcelona Robot Lab Dataset".

Scan Pairs	Rotation (radian)			Translation (m)		
	Roll	Pitch	Yaw	X	Y	Z
400→399	0.0059	-0.0133	0.0018	2.79232	0.103961	-0.086873
399→398	0.0068	0.0467	0.0087	2.84329	0.16036	-0.101913
398→397	0.0052	-0.0097	0.0018	2.80793	0.0485357	-0.0795451
397→396	0.000473	-0.0067	0.0038	2.77586	-0.0190202	-0.0625626
396→395	0.0029	-0.0112	0.3494	1.25819	-0.597171	-0.0271549
395→394	0.0027	0.0016	0.1732	2.79958	-0.160575	-0.0698585
394→393	0.0138	-0.0147	0.1891	2.74287	0.489009	-0.138933
393→392	0.0040	-0.0107	0.1772	2.87109	0.0213742	-0.0641087
392→391	0.0234	0.0225	0.0537	2.72459	-0.231774	-0.0951217
391→390	0.0208	-0.0220	0.3544	2.75221	0.514171	-0.0645027
390→389	0.0036	-0.0097	0.0027	4.05985	-0.530137	-0.0535576
389→388	0.0175	-0.096	0.0062	2.70263 0	-0.260445	-0.196451



(a) 391th and 390th scan before registration.

(b) registration result of 391th and 390th scan.

(c) 395th and 394th scan before registration.

(d) registration result of 395th and 394th scan.

(e) 396th and 395th scan before registration.

(f) registration result of 396th and 395th scan.

Fig. 4.10 Examples of registration results of "Barcelona Robot Lab Dataset" by our SEI-based scan registration algorithm.

Table 4.2 Processing time of our SEI-based scan registration algorithm applied to "Barcelona Robot Lab Dataset" (without resample).

	Time of First Iteration (s)				Time of Second Iteration (s)				
Scan Pairs	Generate SEI		Rotation	Translation	Generate SEI		Rotation	Translation	Total
	scene	object	Determination	Recovery	scene	object	Determination	Recovery	
400→399	0.098207	0.092583	0.381281	0.557346	0.0	0.077856	0.379776	0.472561	2.059610
399→398	0.127332	0.096020	0.381133	0.571251	0.0	0.095962	0.378689	0.549705	2.200092
398→397	0.160579	0.126263	0.383600	0.750119	0.0	0.096373	0.381492	0.795221	2.693647
397→396	0.167615	0.160188	0.386511	0.792298	0.0	0.133113	0.386311	0.729247	2.755283
396→395	0.158078	0.162675	0.381319	0.756967	0.0	0.158279	0.378327	0.798741	2.794386
395→394	0.140747	0.157486	0.379316	0.678922	0.0	0.123012	0.382918	0.730912	2.593313
394→393	0.129555	0.13625 0	0.385851	0.661151	0.0	0.116663	0.380816	0.669772	2.480058
393→392	0.116299	0.125362	0.383832	0.576671	0.0	0.108322	0.384039	0.567664	2.262189
392→391	0.113246	0.115384	0.377098	0.714632	0.0	0.117601	0.377837	0.686709	2.502507
391→390	0.114355	0.112792	0.377958	0.638937	0.0	0.088154	0.376452	0.609597	2.318245
390→389	0.1377	0.113446	0.384028	0.717251	0.0	0.083286	0.381519	0.673142	2.490372
389→388	0.141409	0.136310	0.383613	0.699412	0.0	0.120870	0.382092	0.660261	2.523967

Table 4.3 Processing time of our HTD-based scan registration algorithm applied to "Barcelona Robot Lab Dataset" (without resample).

Scan Pairs	Time per step (s)				Total (s)
	Normal Estimation	HTD Calculation	Rotation Determination	Translation Recovery	
400→399	0.998135	0.279086	0.002653	0.544034	1.823908
399→398	1.253396	0.340542	0.009601	0.558170	2.161709
398→397	1.635833	0.451452	0.009358	0.731865	2.828508
397→396	1.899219	0.523556	0.009649	0.786792	3.219216
396→395	1.885895	0.517856	0.010126	0.793302	3.207179
395→394	1.674075	0.460510	0.009283	0.741505	2.885373
394→393	1.470301	0.406040	0.009291	0.680845	2.566477
393→392	1.323154	0.375198	0.009354	0.674488	2.382194
392→391	1.235596	0.357126	0.009500	0.724661	2.326883
391→390	1.245816	0.359606	0.009937	0.697608	2.312967
390→389	1.451718	0.402257	0.009315	0.686720	2.550010
389→388	1.595896	0.460185	0.009457	0.661905	2.727443

Performance of SEI-based registration algorithm

The performance of our SEI-based registration algorithm is presented and compared to the state-of-the-art registration algorithms which are outlined in Section 3.5.1. In addition, the performance of HTD-based registration algorithm presented in **Chapter 3** is also presented for the sake of comparison.

The registration results by our SEI-based registration algorithm are proposed in Table 4.1. Our scan registration algorithm succeeds in all the 12 scan pairs without exception. Three example scan pairs and the registration results are presented in Figure 4.10. By the way, our HTD-based registration algorithm also succeeds in all the scan pairs.

The registration is implemented on the standard personal desktop computer with the following details:

- **Memory:** 7.7GiB

- **Processor:** Intel® Core™ i5-3750 CPU @ 3.4GHZ
- **System:** 64-bit Ubuntu 12.04 LTS

Specifically, with regard to the efficiency of the SEI-based registration algorithm, the processing time without resampling of the original scan is listed in Table 4.2. We could see from Table 4.2 that the runtime of our algorithm is quite stable, and all the registration could be finished in less than 3 seconds. Although some scan pairs could be aligned well without the second iteration, we still apply the iteration technique and record the runtime since it is impossible to know which scan pairs need the second iteration while which ones not before registration implemented. For the sake of comparison, the runtime of our HTD-based registration algorithm is proposed in Table 4.3. For some scan pairs, the SEI-based method needs less runtime, while for others, the HTD-based method requires less. It is difficult to say which algorithm has high efficiency since the runtime of the two algorithm is so close. If have to say, the HTD-based method requires more than 3 seconds for two scan registration tasks. But as pointed out in Section 3.5.2, the HTD-based method achieves more precise results than the SEI-based method.

Table 4.4 Performances of five registration algorithms applied to "Barcelona Robot Lab Dataset" (without resampling).

Scan Pairs	SEI	HTD	ICP	FPFH	plane-based
400→399	✓	✓	✓	✓	✗
399→398	✓	✓	✓	✗	✗
398→397	✓	✓	✓	✓	✓
397→396	✓	✓	✓	✗	✓
396→395	✓	✓	✓	✓	✓
395→394	✓	✓	✓	✓	✓
394→393	✓	✓	✓	✓	✓
393→392	✓	✓	✓	✓	✓
392→391	✓	✓	✓	✓	✓
391→390	✓	✓	✓	✓	✓
390→389	✓	✓	✗	✗	✓
389→388	✓	✓	✗	✓	✓
Success ratio (%)	**100.00**	**100.00**	83.33	75.00	83.33

Comparison with the state-of-the-art

In order to prove the superiority of our registration algorithm, we also compare our novel SEI-based registration algorithm with the state-of-the-art methods introduced in Section 3.5.1 and the HTD-based method proposed in **Chapter 3**. For convenience of other researchers to reproduce the registration results, the dataset in PCD format and the code implementing our SEI-based iteration registration algorithm, ICP and FPFH-based techniques are public available on the Internet, please refer to Appendix A. FPFH algorithm combining the keypoints detection methods NARF and 3D SIFT is also accessible in our code, but it is not employed in this comparison experiment. Instead, we use the uniformly resampled points as the keypoints, which takes much more time but achieve better registration results.

The performance of the five registration algorithms are presented in Table 4.4. For ICP method, the *max correspondence distance* is set to be 5.0 meter, which seems to be much large but is determined by plenty of trials. The FPFH-based algorithm resamples the scan with leaf size 0.5 meters as the keypoints. ICP method performs better than in the experiment depicted in Section 3.5.2, since there are less offsets between successive scans. Actually, all the five registration algorithms work pretty well with regard to this registration task.

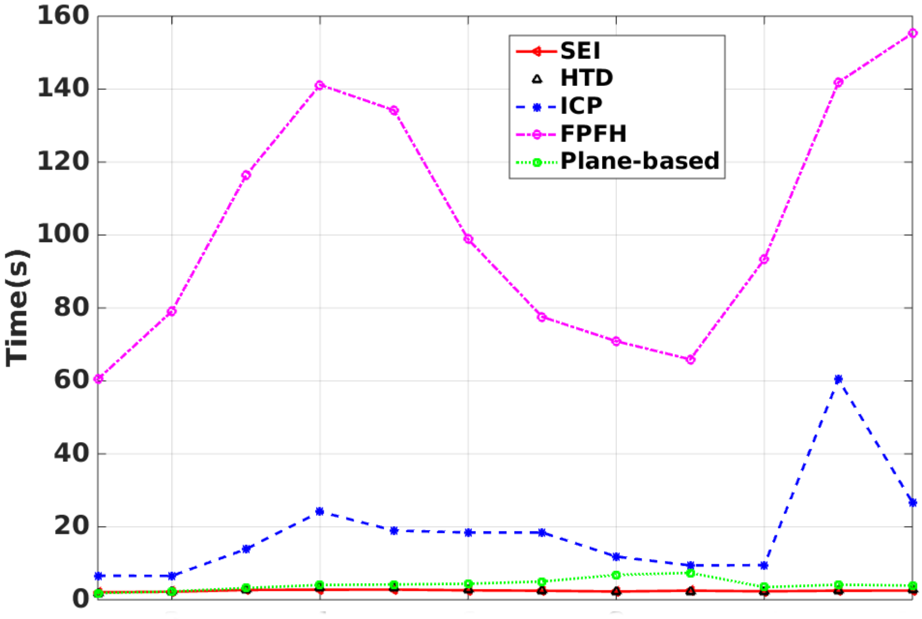


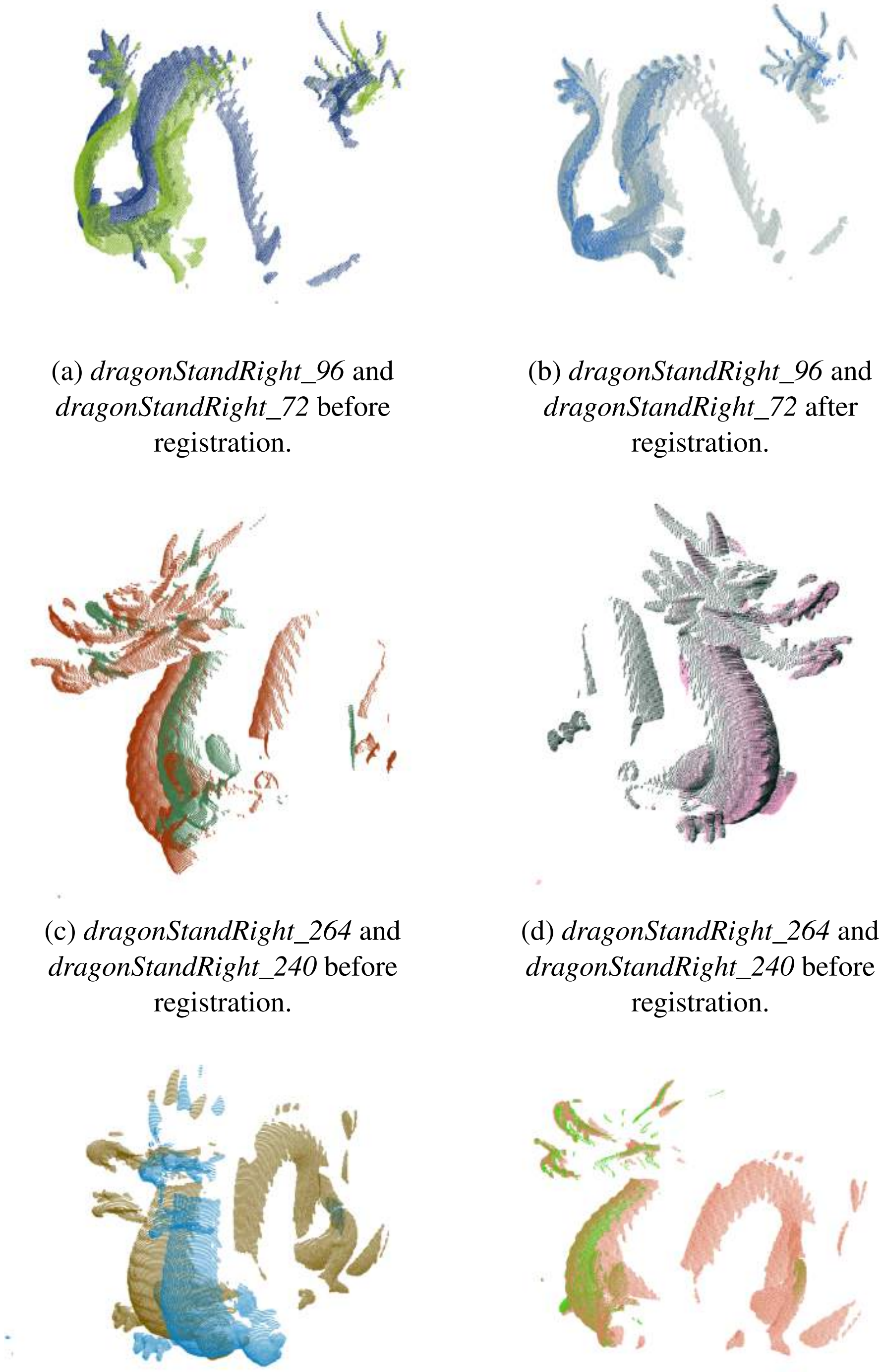
Fig. 4.11 Runtime of five registration algorithms applied to "Barcelona Robot Lab Dataset"(without resample).

In order to compare the efficiency of the five registration algorithms with respect to this dataset, their runtime on the same personal desktop computer is depicted in Figure 4.11. As before, the ICP and FPFH-based registration algorithms require too much more time than our HTD-based and SEI-based techniques, and show large variation. No matter succeed or not, the runtime of plane-based method [163] is less than ICP and FPFH-based registration algorithms and slightly more than our HTD-based and SEI-based methods. For the HTD-based and SEI-based methods, it is difficult to say which algorithm has high efficiency since the runtime of the two algorithm is so close. In the next experiment, we will present the registration task that our SEI-based registration performs well but the HTD-based method could not, which proves the superiority of SEI-based registration compared with HTD-based method.

4.5.4 *Dragon Stand* in "Stanford 3D Scan Scanning Repository"

Our SEI-based scan registration algorithm is a really feature-less registration method, and it could not only be used to build the maps for robots but also to align the surfaces of household objects. In this experiment, we adopt the range data of *Dragon Stand* in "Stanford 3D Scan Scanning Repository", which is famous in 3D scan processing community and publicly available on the Internet [133], to validate the ability our SEI-based method to align the surfaces of household objects without any specific features. By the way, ICP registration algorithm performs really well for this kind of applications as reported [12] [30] [118].

The 15 scans in *Dragon Stand* package are employed. The dataset in PCD file format is available on the Internet, please refer to Appendix A. Since the plane-based [163] and HTD-based registration algorithms could not handle this kind of applications, their registration results are not presented and discussed. There is almost no translations between the successive scans, so the SEI-based method does not need iteration to cope with the adverse effect of translation when estimate the rotation. The plain ICP algorithm and the normal-based ICP algorithm are both employed in this experiment. This registration task is difficult for global registration method, since there are less overlaps between successive scans, just as Figure 4.12 shows. While the less overlap is not a problem for local registration methods, the local methods could work well if and only if there are less offsets between scan pairs.



(a) *dragonStandRight_96* and *dragonStandRight_72* before registration.

(b) *dragonStandRight_96* and *dragonStandRight_72* after registration.

(c) *dragonStandRight_264* and *dragonStandRight_240* before registration.

(d) *dragonStandRight_264* and *dragonStandRight_240* before registration.

(e) *dragonStandRight_288* and *dragonStandRight_264* before registration.

(f) *dragonStandRight_288* and *dragonStandRight_264* before registration.

Fig. 4.12 Examples of *Dragon Stand* scans and the corresponding registration results by our SEI-based scan registration algorithm.

Table 4.5 Performances of four registration algorithms applied to *Dragon Stand* (without resampling).

Scan Pairs	SEI	Plain ICP	ICP based on Normal	FPFH
*dragonStandRight*24→*dragonStandRight*0	✓	✓	✓	✓
*dragonStandRight*48→*dragonStandRight*24	✓	✓	✓	✓
*dragonStandRight*72→*dragonStandRight*48	✓	✓	✓	✓
*dragonStandRight*96→*dragonStandRight* 72	✓	✗	✓	✗
*dragonStandRight*120→*dragonStandRight*96	✓	✓	✓	✗
*dragonStandRight*144→*dragonStandRight*120	✓	✓	✓	✗
*dragonStandRight*168→*dragonStandRight*144	✓	✓	✓	✗
*dragonStandRight*192→*dragonStandRight*168	✓	✓	✓	✓
*dragonStandRight*216→*dragonStandRight*192	✓	✓	✓	✗
*dragonStandRight*240→*dragonStandRight*216	✓	✓	✓	✓
*dragonStandRight*264→*dragonStandRight*240	✓	✓	✓	✗
*dragonStandRight*288→*dragonStandRight*264	✓	✓	✓	✓
*dragonStandRight*312→*dragonStandRight*288	✓	✓	✓	✓
*dragonStandRight*336→*dragonStandRight*312	✓	✓	✓	✗
*dragonStandRight*0→*dragonStandRight*336	✓	✓	✓	✓
Success ratio (%)	**100.00**	93.33	**100.00**	53.33

Table 4.6 Processing time of our SEI-based scan registration algorithm applied to *Dragon Stand* (without resample).

Scan Pairs	Time of per step (s)				Total
	Generate SEI		Rotation	Translation	
	scene	object	Determination	Recovery	
~~24→~~0	0.016555	0.014305	0.380083	0.161735	0.572678
~~48→~~24	0.014442	0.010459	0.379623	0.12837	0.532894
~~72→~~48	0.010925	0.008001	0.378071	0.10508	0.502077
~~96→~~72	0.008365	0.010345	0.380985	0.117875	0.51757
~~120→~~96	0.010086	0.014518	0.378677	0.124043	0.527324
~~144→~~120	0.014838	0.017278	0.379016	0.139924	0.551056
~~168→~~144	0.017402	0.017936	0.379491	0.167985	0.582814
~~192→~~168	0.017956	0.016434	0.377876	0.152093	0.564359
~~216→~~192	0.016213	0.013612	0.380421	0.154821	0.565067
~~240→~~216	0.013664	0.009671	0.379452	0.114244	0.517031
~~264→~~240	0.009968	0.008618	0.378251	0.134984	0.531821
~~288→~~264	0.008254	0.011538	0.377983	0.147195	0.54497
~~312→~~288	0.011369	0.016604	0.383193	0.127056	0.538222
~~336→~~312	0.016370	0.017004	0.378389	0.164699	0.576462
~~0→~~336	0.017192	0.016399	0.378903	0.141931	0.554425

For ICP methods, including plain ICP and ICP based on normals, the *max correspondence distance* is set to be 0.01 meters. And when estimate the normals of points, we search the nearest 30 points rather than set the fixed search radius. The FPFH-based algorithm resamples the scans with leaf size 0.05 meters as keypoints and the support radius of keypoints when compute FPFH is 0.03 meters. As before, the dimension of spherical grids when calculate the SEIs is still 64×64 and the bandwidth of SHT is 32, which proves the robustness to parameters of our SEI-based algorithm.

The performance of the registration algorithms are presented in Table 4.5. It can be seen from Table 4.5 that the local registration methods work well, the normal-based ICP method succeed for all the 15 scans and the plain ICP method succeed for 14 scans. To be honest, our SEI-based algorithm performs better than expected, considering the less overlaps between scans. Our SEI-based method succeeds in all the scan pairs without exception, and gives pretty good results, as the Figure 4.12 depicted. Please note that the results are produced by our SEI-based method without refinement by any local registration techniques. In consideration of the less overlaps between scan pairs, the FPFH-based method works poor for this dataset.

The details of runtime of our SEI-based registration method is proposed in Table 4.6. We could see from Table 4.6 that the runtime of our SEI-based algorithm is quite stable, and the generation of SEIs could be implemented offline and separately before online registration. Furthermore, although only the generation of SEIs of scenes is needed, the runtime for generation SEIs of both objects and scenes are listed in Table 4.6. For instance, in the procedure aligning *dragonStandRight*48 to *dragonStandRight*24, the algorithm only needs to compute the SEI of *dragonStandRight*48, since the SEI of *dragonStandRight*24 has been calculated in the last registration procedure.

Table 4.7 Processing time of four scan registration algorithms applied to *Dragon Stand* (without resample).

Scan Pairs	SEI	Plain ICP	ICP based on Normal	FPFH
~~24→~~0	0.572678	2.596035	1.01024	57.5285
~~48→~~24	0.532894	1.628604	0.708653	33.3644
~~72→~~48	0.502077	1.399241	0.599606	17.6705
~~96→~~72	0.51757	1.907137	1.07388	22.0373
~~120→~~96	0.527324	2.222456	0.888632	41.0801
~~144→~~120	0.551056	3.081652	0.962245	64.5501
~~168→~~144	0.582814	3.472294	1.12889	76.9048
~~192→~~168	0.564359	3.053031	1.02088	73.0521
~~216→~~192	0.565067	2.647928	0.933933	54.7115
~~240→~~216	0.517031	2.239667	0.617719	33.5312
~~264→~~240	0.531821	3.643474	0.835928	21.9377
~~288→~~264	0.54497	3.723	1.35585	25.1341
~~312→~~288	0.538222	3.37609	1.13287	45.2044
~~336→~~312	0.576462	3.354734	1.32149	65.5567
~~0→~~336	0.554425	3.412242	1.2286	69.598

Since both our SEI-based method and ICP methods work well, the comparison of the efficiency seems more important. So the runtime of the four algorithms are listed in Table 4.7 and also depicted in Figure 4.13. We could see that our SEI-based registration algorithm needs the least runtime and gives results in about 0.5 seconds. Admittedly, it is unfair for FPFH-based algorithm to be compared when it is not applied combining the keypoints detection techniques. But the combination with keypoints detection techniques could not meliorate the fluctuation of its runtime. For the ICP methods, the normal-based ICP needs less time than the plain ICP method, which breaks free from the expected.

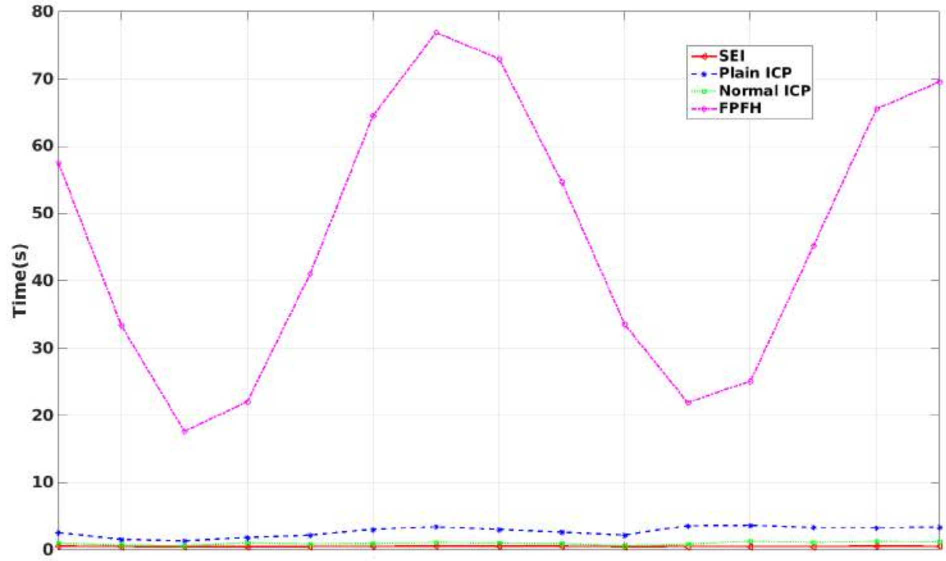


Fig. 4.13 Runtime of four registration algorithms applied to *Dragon Stand* (without resample).

4.6 Experiments about SEI-based feature matching

In this section, the SEI is applied as a local descriptor, and compared with the existing and representative local descriptors based on the public available dataset. The reason why we adopt the public available dataset is that it is convenient for other researchers to compare our algorithm with their own algorithms.

Experiments have been carried on standard personal desktop computers. We implement the two kinds of rotation-invariant SEIs as two separate feature Classes in PCL, and the code has been published online, please refer to Appendix A. The Point Cloud Library (PCL) [45] is utilized for reading and writing point clouds and 3D visualization, and SEI is implemented as a typical PCL feature class which could be combined seamlessly with other classes of PCL. The linear algebra library Eigen [24], a C++ template library for linear algebra, is employed for matrix processing. The SOFT Package [46] is employed to obtain the Spherical Harmonic coefficients of SEIs. The *std::chrono::high_resolution_clock* class in C++ 11 is employed to record the runtime.

In Section 4.6.1, we review three representative local shape descriptors, which are employed to compare with rotation-invariant SEI. And then in Section 4.6.2, we describe the methodology of this experiment, including dataset, procedure and measurement. The key is how to compare local shape descriptors fairly and independent of other techniques. The comparison results are presented in Section 4.6.3 and the superiority of our algorithm is discussed in Section 4.6.3. Due to the space limitation, we cannot present all the results. Indeed, some typical results has been selected for explanation.

4.6.1 Related work

In this part, we recall three typical local shape descriptors, Signature of Histograms of Orientation (SHOT), 3D Shape Context (3DSC) and Fast Point Feature Histograms (FPFH), and all of them are popular and widely used in computer vision/geometry community. Even though FPFH is introduced in Section 3.5.1, we still recall its procedure briefly here for clarity. And we omit famous Spin Image (SI) [66][65] because 3DSC is regarded as a successful inheritor and extension of SI.

Signature of Histograms of Orientations (SHOT)

Signature of Histograms of Orientations (SHOT) is a more recent shape descriptor presented by Computer Vision Laboratory of University of Bologna [150][123]. SHOT superimposes a 3D grid on the support and calculates a set of local histograms over the 3D volumes defined by grids. The local histograms are built on the normals of points, since normals are considered more descriptive than plain 3D coordinates. Finally, all local histograms are grouped together to form the SHOT descriptor. SHOT combines both the robustness of histograms and descriptiveness of points location, and is regarded as a successful local shape descriptor.

Specifically, the calculation of SHOT contains four steps:

1. for each keypoint, search the points in its support and build a repeatable and disambiguated LRF based on the Eigenvalue Decomposition (EVD) of the covariance matrix of the point coordinates within the support. And then transform the global coordinates of points into corresponding local coordinates.

2. divide the support by an isotropic spherical grid that encompasses partitions along the radial, azimuth and elevation axes.

3. build the local histograms of each 3D sub-volume determined by the spherical grids. The local histograms are built by accumulating point clouts into bins according to $cos(\theta_i)$, which could be achieved by $cos(\theta_i) = \vec{n}_u \cdot \vec{n}_{v_i}$ where $\vec{n}_u$ is the normal at the keypoint and $\vec{n}_{v_i}$ is the normal at point v_i within the sub-volumes of support.

4. group together all the local histograms to formulate the final SHOT descriptor.

PCL is a good example implementation SHOT feature. In this experiment, we use the *pcl::SHOTEstimation* class to compute the SHOT features and adopt the *SHOT352* point type.

3D Shape Context (3DSC)

3D shape context (3DSC) proposed in [51][149] is inherited directly from the 2D shape context [11]. 3DSC shares quite similar idea with and is a straightforward extension of 2D shape context. 3DSC divides the support of keypoint and accumulates the weighted count of points located in sub-volumes. SHOT has some similarity with 3DSC, but for each sub-volume SHOT builds a histogram rather than just counts the number of points. And the way of support division is different between SHOT and 3DSC.

Overall, the estimation of 3DSC includes three steps:

1. for each keypoint, build its LRF and transform the coordinates of points in its support to the local coordinates in this LRF.

2. divide the support equally in azimuth and elevation directions and logarithmically along the radial dimension into several bins.

3. for each bin, accumulate the weighted count of points located in it.

In this experiment, we use the *pcl::ShapeContext3DEstimation* to compute the 3DSC features and employ the *pcl::ShapeContext1980* point type.

Fast Point Feature Histograms (FPFH)

FPFH is a simplified version of Point Feature Histograms (PFH) [122], and it reduces the computational complexity but retains most of the discriminative power of PFH [119]. FPFH is calculated based on the oriented points, which means just like SHOT and 3DSC, FPFH is also calculated based on normals of points. But unlike SHOT and 3DSC, FPFH describes the property of the support independent of the point locations, which means FPFH does not build a LRF related to keypoints.

Generally speaking, FPFH calculation includes three steps:

1. for each keypoint p_i in the scan, search its support;
2. for every point pair in support, calculate three features which together to express the mean curvature. The three features are combined and put into the equivalent histogram bin. The histograms are called Simplified Point Feature Histogram (SPFH).
3. After achieving SPFH of every point, for each point p_i redetermine its k neighbours and use the neighbouring SPFH values to weight the final histogram of p_i:

$$\mathbf{FPFH}(p_i) = \mathbf{SPFH}(p_i) + \frac{1}{k}\sum_{j=1}^{k}\frac{1}{\omega_k}\cdot\mathbf{SPFH}(p_j)$$

FPFH is also implemented in PCL. We use the *pcl::FPFHEstimation* class in PCL to compute FPFH in this experiment, and adopt the *pcl::FPFHSignature33* feature type.

4.6.2 Methodology

How to compare local shape descriptors fairly and independent of other techniques is the key to the methodology of this experiment. The local features are usually adopted as one part in object recognition, but the performance of object recognition depends on not only the effectiveness of feature matching but also the repeatability of keypoints detection and efficiency of features cluttering techniques.

In this experiment, we use the "**BoD1**" dataset publicly available on the Internet: *http://vision.deis.unibo.it/research/80-shot*, which is created especially for the comparison of feature descriptors. The dataset is created from six model belonging to the "Stanford 3D Scan Scanning Repository" [133]. Each scene contains a subset of the 6 models randomly chosen and randomly rotated and translated so to create clutter conditions.

We use the same keypoints detector for all the feature descriptors for a fair comparison. Specifically, we select 100 points randomly in each model, then we extract their corresponding points from the scenes, in this way, the performance of feature matching is not affected by error of keypoints detectors. It is worthy to note that the keypoints are selected randomly but they should be the same ones for all the four feature descriptors: SEI, SHOT, 3DSC and FPFH. The keypoints of the six model used in this experiments are shown in Figure 4.14. Limited by the space, we just present the features matching results of selected 5 scenes, the scenes and the corresponding keypoints are depicted in Figure 4.15.

With regard to the evaluation of local shape descriptors, we adopt the frequently used criteria named *Recall vs 1-Precision* curve. It is generated by changing the thresholds for feature matching and computing the feature recall and precision for each threshold. *Precison* and *Recall* are the basic measures used in evaluating search algorithms, but they are naturally to be used to evaluate feature matching strategies. In search problems, there is a set of records in the database which is relevant to the search topic, and the search algorithms will give a set of retrieved records. The retrieved records could be relevant or irrelevant. And *Precison* and *Recall* is defined as following:

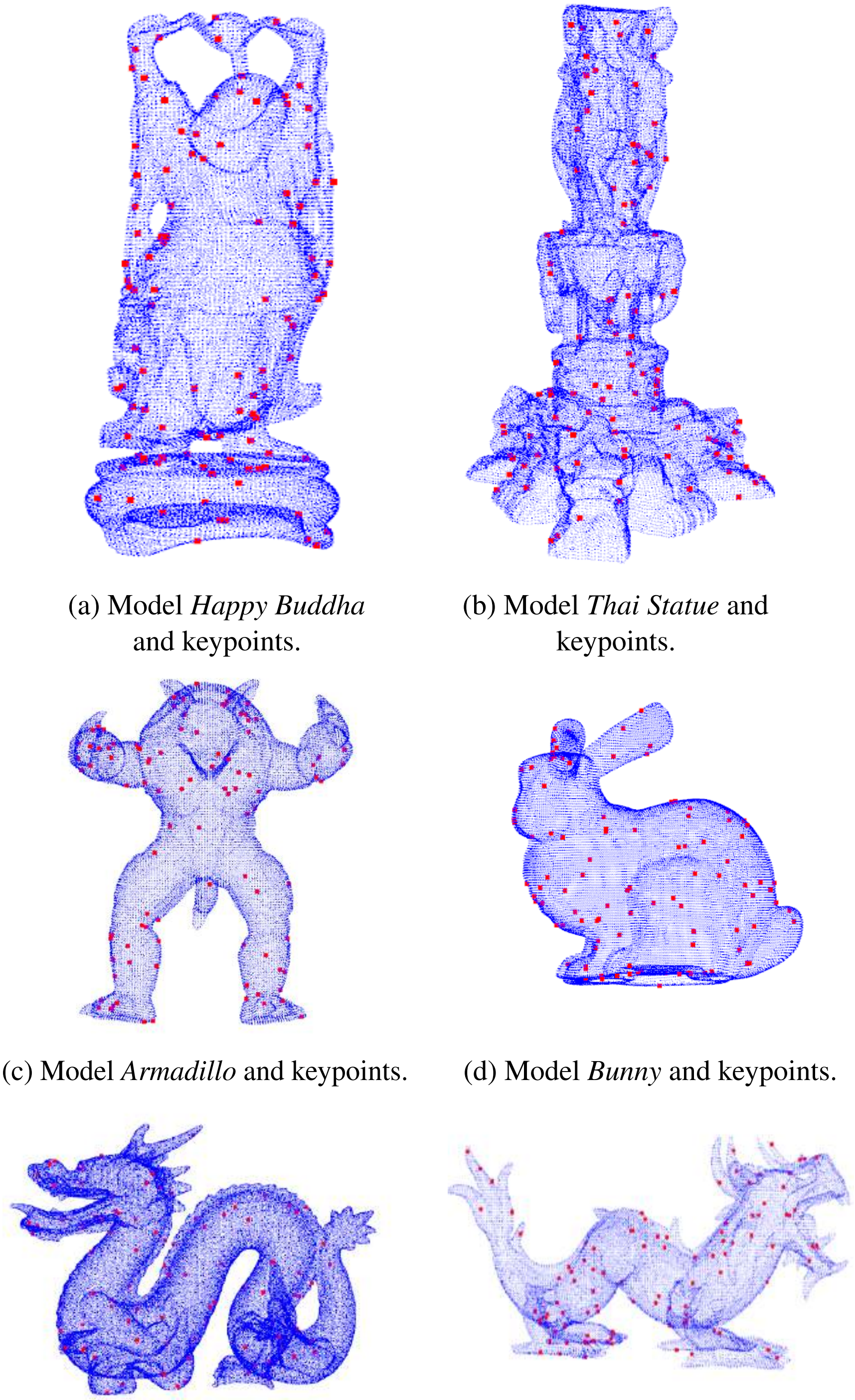
$$Recall = \frac{Relevant\ Records \cap Retrieved\ Records}{Relevant\ Records} \tag{4.14}$$

$$Precision = \frac{Relevant\ Records \cap Retrieved\ Records}{Retrieved\ Records} \tag{4.15}$$

Perhaps, an example could be helpful to understand the precision and recall concepts. Suppose the case for recognizing cups in a scene which contains 35 cups and 20 bottles, and the algorithm recognizes 20 objects which actually includes 15 cups and 5 bottles. For this scenario, the recall of the recognition algorithm is $15/35$ and the precision is $15/20$. Normally, the recall and precision are inversely related, which means when recall increases the precision decreases and vice versa.

For our feature matching experiment, the relevant records are the 600 point correspondences, 100 for each model. And the feature matching strategies would build n correspondences under the specified threshold. Assume the n correspondences contain m correct correspondences, then the recall the the feature matching strategy under this threshold is $m/600$ and precision if m/n.

To evaluate the robustness to noise of feature matching techniques, we add different levels of Gaussian noise to the original datasets. We calculate the mesh resolution of the models, select the maximum mesh resolution (mr) and add Gaussian noise with standard



(a) Model *Happy Buddha* and keypoints.

(b) Model *Thai Statue* and keypoints.

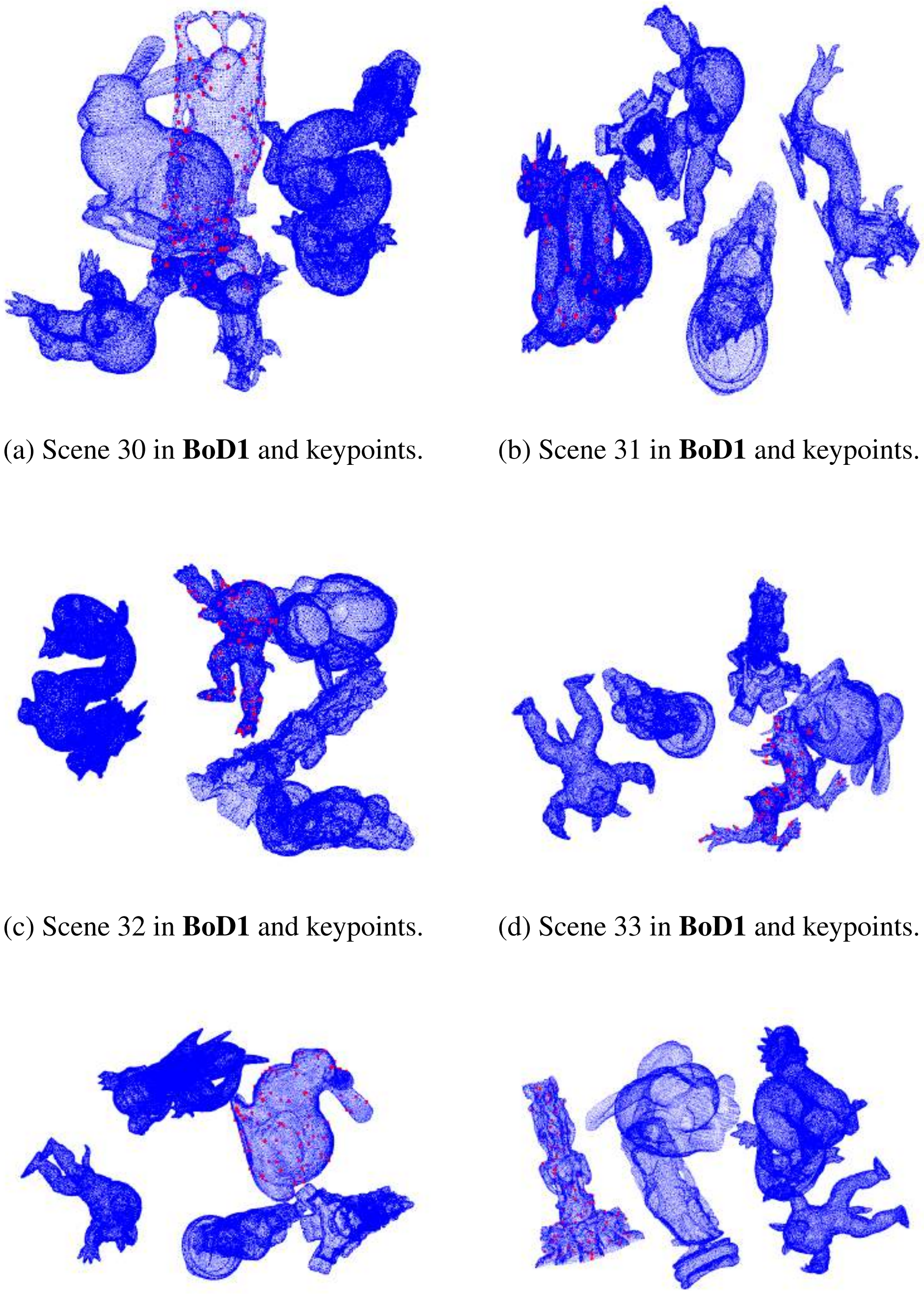
(c) Model *Armadillo* and keypoints.

(d) Model *Bunny* and keypoints.

(e) Model *Dragon* and keypoints.

(f) Model *Asian Dragon* and keypoints.

Fig. 4.14 Models and the keypoints used in the experiments of SEI as a local shape descriptor.



(a) Scene 30 in **BoD1** and keypoints.

(b) Scene 31 in **BoD1** and keypoints.

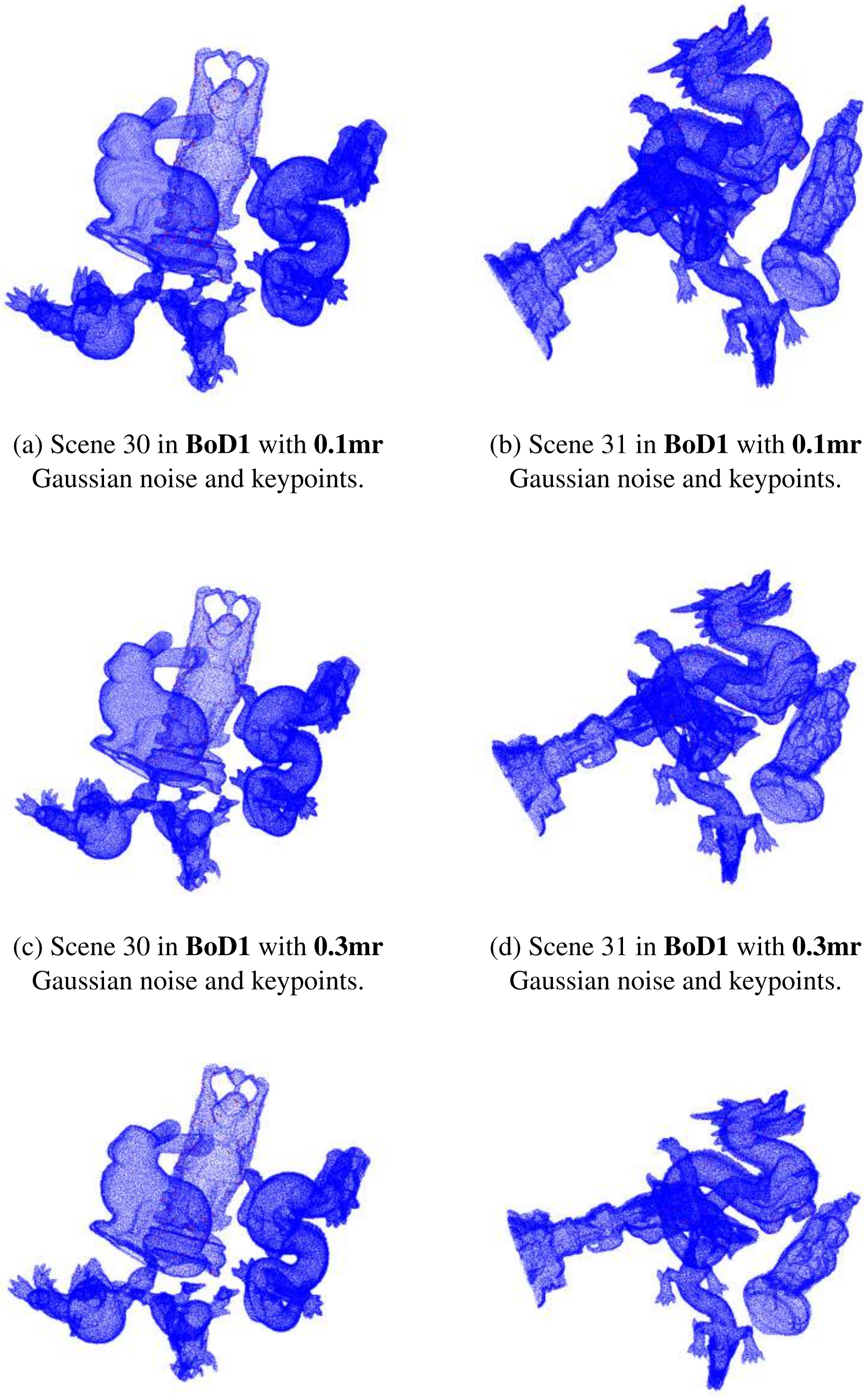
(c) Scene 32 in **BoD1** and keypoints.

(d) Scene 33 in **BoD1** and keypoints.

(e) Scene 35 in **BoD1** and keypoints corresponding with *Bunny*.

(f) Scene 35 in **BoD1** and keypoints corresponding with *Thai Statue*.

Fig. 4.15 Scenes and the keypoints used in the experiments of SEI as a local shape descriptor.



(a) Scene 30 in **BoD1** with **0.1mr** Gaussian noise and keypoints.

(b) Scene 31 in **BoD1** with **0.1mr** Gaussian noise and keypoints.

(c) Scene 30 in **BoD1** with **0.3mr** Gaussian noise and keypoints.

(d) Scene 31 in **BoD1** with **0.3mr** Gaussian noise and keypoints.

(e) Scene 30 in **BoD1** with **0.5mr** Gaussian noise and keypoints.

(f) Scene 31 in **BoD1** with **0.5mr** Gaussian noise and keypoints.

Fig. 4.16 Example scenes with different level of Gaussian noise and keypoints.

deviation $0.1mr$, $0.3mr$, $0.5mr$. The examples of *Scene 30* and *Scene 31* with different levels of Gaussian noise are presented in Figure 4.16.

For the correspondence matching algorithm, we compute the phase correlation between features and find the maximum value for the correlation coefficients by the following equations:

$$C(\boldsymbol{\alpha}) = \frac{\overline{\mathscr{F}_m(\boldsymbol{\alpha})}}{|\mathscr{F}_m(\boldsymbol{\alpha})|} \bullet \frac{\mathscr{F}_s(\boldsymbol{\alpha})}{|\mathscr{F}_s(\boldsymbol{\alpha})|} \tag{4.16}$$

$$c(\boldsymbol{x}) = \mathscr{F}^{-1}\{c(\boldsymbol{\alpha})\} \tag{4.17}$$

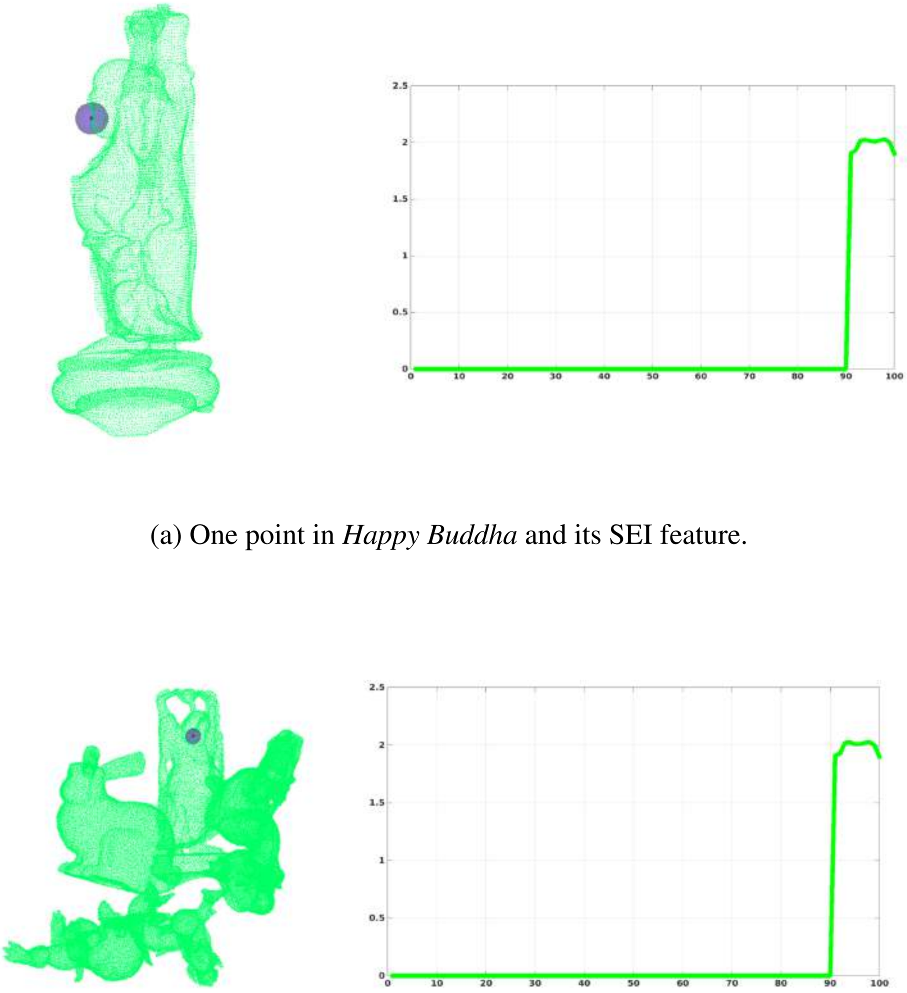
$$d = \max\{c(\boldsymbol{x})\} \tag{4.18}$$

Theoretically, the phase correlation coefficient, which is regarded as the distance between two features for our correspondence matching algorithm, should be 1 between the corresponding features, but in practice, the peak deteriorates because of the noise or keypoints movements.

The SEIs of one example point in *Happy Buddha* and its corresponding point in Scene 30 of **BoD1** with 0.3mr Gaussian noise are depicted in Figure 4.17. Our measured distance between these two SEIs is 0.997807. Further, the superiority of our measurement could be proved by the instance proposed in Figure 4.18. The SEIs of one point in *Dragon* and its corresponding point in Scene 31 of **BoD1** with 0.5mr Gaussian noise are described. It could be seen that the two features are different at the end part, which is pointed out by red line in Figure 4.18b. If the similarity of features is measured by Euclidean Distance, which is the most popular measurement, this point pair could not be considered as the correspondence. But if measured by our phase correlation coefficients, the distance is 0.999752, and the point pair is regarded as a quite well correspondence. And for our measured distance, the threshold could be selected easily, since the maximum of the distances between features is 1, which means the features match perfectly.

4.6.3 Experiment results

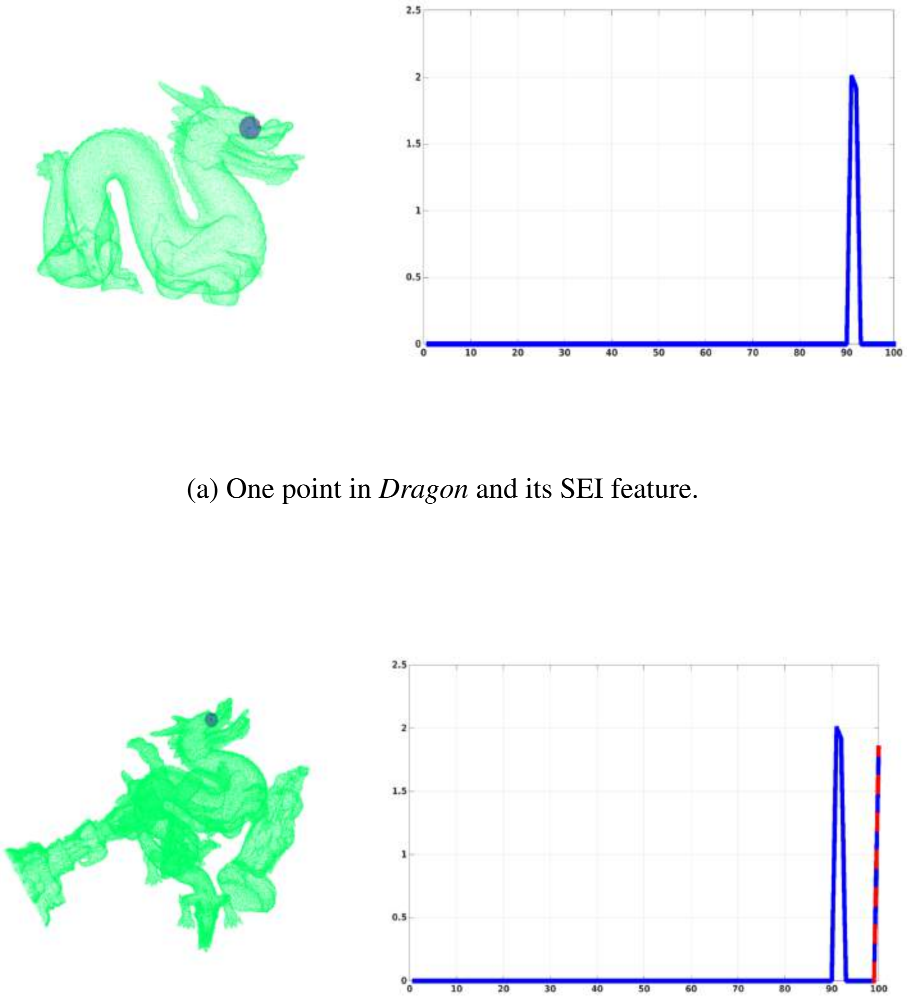
The C++ code to reproduce all the experiment results, including the results of correspondence matching algorithms based on SHOT, 3DSC and FPFH, are published online. In the experiment, we found that the SEI based on repeatable and disambiguated LRF produces better results than SEI based on SHT, so in this part, we only present the experiment result of SEI based on LRF and compare with the state-of-the-art algorithms. However, the C++ code about SEI based on SHT is still available on the Internet, please refer to Appendix A.



(a) One point in *Happy Buddha* and its SEI feature.

(b) Corresponding point in Scene 30 of **BoD1** with 0.3mr Gaussian noise and its SEI feature.

Fig. 4.17 Examples of SEI as local shape descriptor (I).

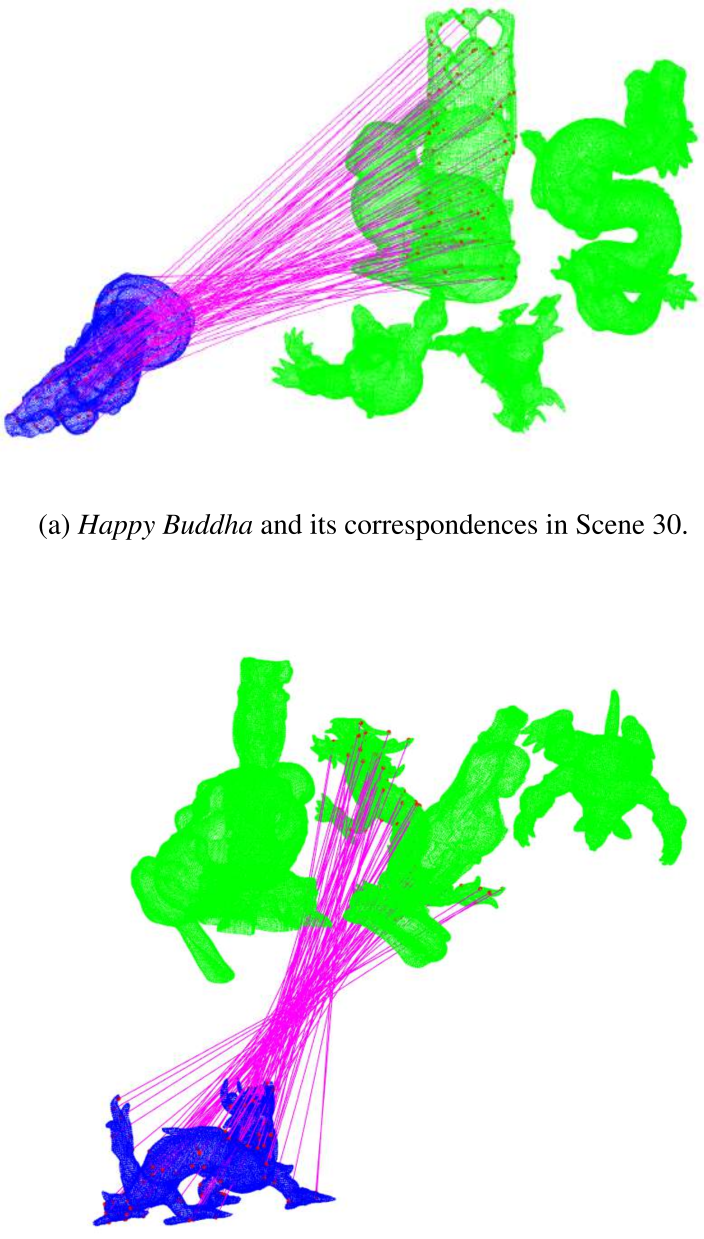


(a) One point in *Dragon* and its SEI feature.

(b) Corresponding point in Scene 31 of **BoD1** with 0.5mr Gaussian noise and its SEI feature.

Fig. 4.18 Examples of SEI as local shape descriptor (II).

Two examples about correspondences built by SEI-based feature matching technique is presented in Figure 4.19. In appearance, our SEI-based feature matching algorithm could search the correspondences precisely and robustly. The quantitative comparison is demonstrated later.



(a) *Happy Buddha* and its correspondences in Scene 30.

(b) *Asian Dragon* and its correspondences in Scene 32.

Fig. 4.19 Example of feature correspondences built by our SEI-based feature matching algorithm.

Table 4.8 Performances of SEI-based feature matching algorithm applied to scenes with 0.1mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	0.0		0.95		0.96		0.97		0.98		0.99		0.999	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	90	99	90	98	90	97	90	92	86	86	84	54	53
Dragon	100	79	97	79	95	77	92	76	86	71	75	68	41	41
Armadillo	100	76	99	76	98	76	96	76	91	76	79	69	62	57
Asian Dragon	100	90	98	90	98	90	98	90	94	89	90	87	64	63
Thai Statue	100	91	99	91	98	90	97	90	93	86	88	82	60	57
Bunny	100	57	100	57	99	56	93	55	78	54	62	47	32	29
SUM	600	483	592	483	586	479	573	477	534	462	480	437	313	330
Recall	**0.8050**		**0.8050**		**0.7983**		**0.7950**		**0.7700**		**0.7283**		**0.5000**	
Precision	**0.8050**		**0.8159**		**0.8174**		**0.8325**		**0.8652**		**0.9104**		**0.9585**	

Table 4.9 Performances of SEI-based feature matching algorithm applied to scenes with 0.3mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	0.0		0.95		0.96		0.97		0.98		0.99		0.999	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	80	98	80	97	80	97	80	92	77	77	68	39	35
Dragon	100	57	97	57	96	57	93	57	76	54	66	50	26	24
Armadillo	100	70	98	69	95	69	86	66	81	63	72	59	42	38
Asian Dragon	100	84	97	84	97	84	96	83	91	82	84	78	46	45
Thai Statue	100	84	99	83	99	83	97	83	93	80	82	75	47	46
Bunny	100	53	100	53	100	53	93	50	80	49	56	41	21	19
SUM	600	428	589	426	584	426	562	419	513	405	437	371	221	207
Recall	**0.7133**		**0.7100**		**0.7100**		**0.6983**		**0.6750**		**0.6183**		**0.3450**	
Precision	**0.7133**		**0.7233**		**0.7295**		**0.7456**		**0.7895**		**0.8490**		**0.9367**	

Table 4.10 Performances of SEI-based feature matching algorithm applied to scenes with 0.5mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	0.0		0.95		0.96		0.97		0.98		0.99		0.999	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	78	98	78	95	77	94	77	89	75	74	70	33	32
Dragon	100	52	96	50	95	50	88	48	79	45	60	39	11	11
Armadillo	100	64	94	61	91	61	88	60	81	55	68	49	34	28
Asian Dragon	100	79	98	79	95	78	92	78	78	68	69	64	26	26
Thai Statue	100	84	98	82	98	82	96	82	91	78	79	69	30	25
Bunny	100	36	100	36	99	36	88	33	71	29	51	27	12	8
SUM	600	393	584	386	573	384	546	378	489	350	401	318	146	130
Recall	**0.6550**		**0.6433**		**0.6400**		**0.6300**		**0.5833**		**0.5300**		**0.2167**	
Precision	**0.6550**		**0.6610**		**0.6702**		**0.6923**		**0.7157**		**0.7930**		**0.8904**	

Table 4.11 Performances of SHOT-based feature matching algorithm applied to scenes with 0.1mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		0.80		0.70		0.50		0.40		0.30		0.20	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	44	91	43	71	36	33	28	27	24	15	15	8	8
Dragon	100	79	94	77	86	75	64	63	61	60	52	51	42	41
Armadillo	100	85	97	83	96	83	83	77	73	55	61	60	46	46
Asian Dragon	100	73	97	71	84	67	62	56	47	68	39	36	17	17
Thai Statue	100	66	96	65	92	63	56	45	26	78	10	9	4	4
Bunny	100	62	88	60	75	56	56	48	48	29	43	41	34	33
SUM	600	409	563	399	504	380	354	317	282	264	220	212	151	149
Recall	**0.6817**		**0.6650**		**0.6333**		**0.5283**		**0.4400**		**0.3533**		**0.2483**	
Precision	**0.6817**		**0.7087**		**0.7540**		**0.8955**		**0.9362**		**0.9636**		**0.9868**	

Table 4.12 Performances of SHOT-based feature matching algorithm applied to scenes with 0.3mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		0.80		0.70		0.50		0.40		0.30		0.20	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	44	91	43	73	40	36	29	26	24	12	12	8	38
Dragon	100	79	94	77	85	76	66	65	61	60	55	54	41	40
Armadillo	100	85	98	83	98	83	79	76	71	69	59	58	47	47
Asian Dragon	100	74	97	72	83	66	62	56	45	41	40	36	17	17
Thai Statue	100	63	96	62	93	60	56	41	27	21	11	10	4	4
Bunny	100	62	88	60	76	55	59	49	47	43	43	40	32	31
SUM	600	407	564	397	508	380	358	316	277	258	220	210	149	147
Recall	**0.6783**		**0.6617**		**0.6333**		**0.5267**		**0.4300**		**0.3500**		**0.2450**	
Precision	**0.6783**		**0.7039**		**0.7480**		**0.8827**		**0.9314**		**0.9545**		**0.9866**	

Table 4.13 Performances of SHOT-based feature matching algorithm applied to scenes with 0.5mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		0.80		0.70		0.50		0.40		0.30		0.20	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	45	89	43	74	38	38	30	25	23	13	13	8	38
Dragon	100	78	95	76	91	76	65	63	62	60	53	51	42	40
Armadillo	100	82	97	80	96	80	77	72	70	65	59	57	45	45
Asian Dragon	100	72	97	71	84	67	65	56	45	40	38	34	16	16
Thai Statue	100	63	99	63	94	61	52	40	24	20	10	9	3	3
Bunny	100	61	89	59	78	54	58	49	49	44	41	38	35	33
SUM	600	401	566	392	517	376	355	310	275	252	214	202	149	145
Recall	**0.6683**		**0.6533**		**0.6267**		**0.5167**		**0.4200**		**0.3367**		**0.2417**	
Precision	**0.6683**		**0.6926**		**0.7273**		**0.8732**		**0.9164**		**0.9439**		**0.9732**	

The feature matching results of our SEI-based algorithm under different thresholds and Gaussian noises are demonstrated detailedly, as Table 4.8, Table 4.9 and Table 4.10 show. SHOT-based feature method performs best in the state-of-the-art methods (compare with FPFH and 3DSC), so we present the results of SHOT-based algorithm in Table 4.11, Table 4.12 and Table 4.13. To keep the compactness, we omit the details of FPFH-based and 3DSC-based feature matching algorithms, but they are available in Appendix C.

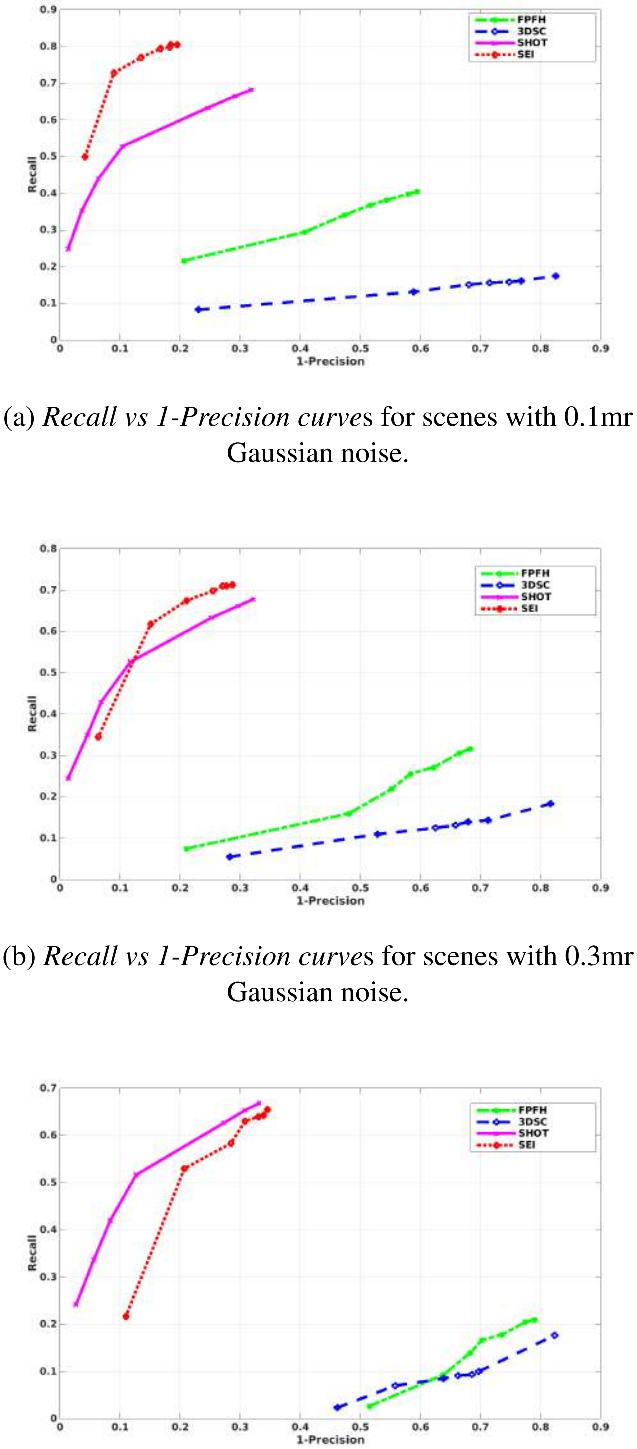
To compare the feature matching results of different algorithm, we depict the *Recall vs 1-Precision* curves in Figure 4.20. We can see that our SEI-based and SHOT-based feature matching algorithms perform orders of magnitude better than FPFH-based and 3DSC-based techniques. As to the comparison between SEI-based and SHOT-based algorithm, our SEI-based algorithm performs better than SHOT-based method for the scenes with 0.1mr and 0.3mr Gaussian noise. But SHOT-based feature matching technique gives better results with regard to scenes with 0.5mr Gaussian noise. And after analysing the results of scenes of 0.5mr Gaussian noise, we found that our SEI-based method gives comparable *recall* results, while the *precision* of SHOT-based method is better than our SEI-based technique. The reason behind this is partially because SHOT-based method employ the more strict Euclidean Distance in searching correspondences procedure.

Table 4.14 Memory requirements and computational efficiency of feature matching algorithms.

	Length	Radius (mr)	Time Normal(ms)	Time Feature(ms)
SEI	64	30	0.000	($\approx$ 24.442)$*$keypints_num
SHOT	352	30	($\approx$ 0.164)$*$scan_size	($\approx$ 32.481)$*$keypints_num
FPFH	33	10	($\approx$ 0.164)$*$scan_size	($\approx$ 109.723)$*$keypints_num
3DSC	1980	30	($\approx$ 0.164)$*$scan_size	($\approx$ 54.838)$*$keypints_num

In addition, we also compare the feature matching methods with respect to their memory requirements and computational efficiency. The experiments are implemented on a laptop with the following details:

- **Memory:** 3.8GiB
- **Processor:** Intel® Core™ i5-2430 CPU @ 2.40GHZ
- **Graphics:** Intel® Sandybridge Mobile
- **System:** 64-bit Ubuntu 14.04 LTS



(a) *Recall vs 1-Precision curve*s for scenes with 0.1mr Gaussian noise.

(b) *Recall vs 1-Precision curve*s for scenes with 0.3mr Gaussian noise.

(c) *Recall vs 1-Precision curve*s for scenes with 0.5mr Gaussian noise.

Fig. 4.20 *Recall vs 1-Precision curve*s of four feature matching algorithms to scenes with difference levels of Gaussian noise.

The memory requirements and computational efficiency of the four feature matching techniques are listed in Table 4.14. The radius of support of feature is set to be 30mr for SEI, SHOT and 3DSC, while the radius is 10mr for FPFH. That is because the FPFH calculation with 30mr support size takes really too much time. As we can see from Table 4.14, FPFH requires the most time even when the support size is 10mr, which is much less than other methods. Our SEI requires much less memory compared with SHOT and 3DSC. And it is worthy to note that all the SHOT, FPFH and 3DSC features require estimating normals of all the points, not only the keypoints. Normally, the normal estimation needs more than 10 seconds in this experiment. While our SEI feature does not require the normal estimation procedure. By the way, the runtime for repeatable LRF estimation of the keypoints is included in the feature estimation time shown in Table 4.14.

4.7 Summary

In this chapter, we propose a novel shape descriptor Spherical Entropy Image (SEI), which inherits the robustness of histogram-based shape descriptors and descriptive power of signature-based shape descriptors. We develop a feature-less registration method using the SEI as a global shape descriptor. This registration method determines the rotation by aligning the SEIs based on the Spherical Harmonic analysis and calculates the translation by the POMF technique, and then integrates the rotation recovery and translation estimation into an iteration framework. Besides, we also show the possibility of using SEI as a local shape descriptor. Several experiments have been carried on to validate the efficiency of SEI. The scan registration experiments involving outdoor scenes and handicrafts are implemented. The experiment results show that the SEI-based registration method outperforms the state-of-the-art algorithms both in successful ratio and processing time. The feature matching experiment employing SEI as a local shape descriptor for public available dataset is also proposed. SEI achieves better performance in terms of *Recall vs 1-Precision* curve and processing time compared with the state-of-the-art shape descriptors.

CHAPTER 5

Conclusion

Study the past, if you would divine the future.

Confucius

Contents

5.1 Conclusion

IN this thesis, we propose two 3D global shape descriptors, Hough Transform Descriptor and Spherical Entropy Image, and develop two novel 3D scan registration methods based on these two shape descriptors respectively.

For the Hough Transform Descriptor-based registration method, we project the 3D scan data into Hough domain, in this way, the 3D rotation of the original data is decoupled from its 3D translation. Our registration method based on Hough Transform Descriptor recovers the rotation and translation between scan pairs *separately*. The rotation recovery is divided into two steps: yaw determination and roll & pitch determination. Overall speaking, we resample the Hough Transform Descriptors to convert yaw/roll/pitch rotation into offsets of resampled Hough Transform Descriptors, and then use phase correlation techniques to determine the offsets. Assume the determined rotation is correct and complete, there is only translation between scan pairs after rerotating them according to the determined rotation matrix.

We apply the Phase Only Matched Filter (POMF), which is based on the Fourier analysis, to estimate the shift between two rerotated scans. The POMF decouples the local signal energy from the signal structure because two shifted signals carry the shift information only within the phase of their Fourier spectrum. Before applying POMF, it is necessary to convert the original 3D surface into volume grids. The general way to achieve volume grids is assigning voxels the value of 1 if they are occupied by the surface, otherwise their values are 0. But we found the volume grids obtained by this method are less informative and the translation recovery usually fails, because the variation of signal is critical to success of Fourier Transform-based correlation techniques. In our novel algorithm, we divide the translation determination into two separate steps: recover the offsets in (x, y) direction and in z direction. We project the scans onto their (x, y) planes and the xy pixel stores the maximum z value of all points whose (x, y) locate in this pixel. The offsets in (x, y) direction could be determined by 2D POMF. Meanwhile, we project the scans onto its z axes to use 1D POMF to calculate the offsets in z direction. Please note that the recovery of translation in (x, y) direction and z direction is independent and could be carried on simultaneously.

We also propose a novel spherical shape descriptor named Spherical Entropy Image (SEI) and apply it in 3D scan registration aided by Spherical Harmonic analysis, which could be regarded as an extension of Fourier transform. We first divide the original scan into several small 3D patches according to the elevation and longitude of point. And for each patch, consider the depth of points as observations of a random variable and compute the entropy of this variable. The final SEI of the 3D point clouds could be achieved by computing the entropy of all patches in a dense manner. The transformation between the original data could be recovered by aligning their corresponding SEIs. The SEI is a spherical function defined on the sphere. The sphere could be regarded as the critical point between 3D space and 2D surface, and the rotation of sphere could be converted to the translation of 2D spherical surface. The generalized convolution theorem based on Spherical Harmonic analysis could be employed to match patterns defined on 2D sphere. The translation recovery in this algorithm adopts the same method employed by the registration method based on Hough Transform Descriptor.

Entropy estimation is translation-invariant, since it is merely related to probability of the depth of points but independent of value of the depth. However, SEI is not translation-invariant because the way to divide scans is translation-dependent. In other words, unlike the registration method based on Hough Transform Descriptor, it is impossible to estimate the rotation and translation separately for the SEI-based registration method. Thus we run into a classical Chicken-and-Egg problem: it is necessary to achieve the rotation to recover the translation, while the translation rectified is helpful to determine the rotation matrix accurately. In our algorithm, we borrow the iteration idea to handle this dilemma, which is one of our major contributions. The experiments show that 2 iterations are usually enough, which ensures our SEI-based registration methods are more efficient than most existing registration methods. Indeed, iteration is needless when there is only rotation between scan pairs, in which situation our SEI-based registration method performs efficiently and precisely. Besides the usage of SEI as a global shape descriptor in scan registration, the possibility of using SEI as a local shape descriptor in feature matching task is also discussed.

Elaborate experiments prove the efficiency of our novel methods. Specifically, we employ both the public available datasets and the dataset captured by our own custom-built platform. We believe the public available datasets and code could help other researchers to compare our algorithms with their own research. The code and datasets to reproduce all the experiment results in this thesis are published on the Internet, please refer to Appendix A. The experiments prove that our novel scan registration algorithms based on global shape descriptors outperform the state of the art registration methods with regard to the employed datasets.

The SEI-based feature-less scan registration strategy has much similarity with the HTD-based scan registration method proposed in Section 3.4.1. Both of these two registration algorithms describe the original scan by a global shape descriptor, and determine the transformation between the original scans by aligning the corresponding shape descriptors. Moreover, these two algorithms both use POMF to recover the translation after rotation matrix is estimated and applied to the original scans.

The difference is the technique how to estimate the rotation matrix between the spherical images. The rotation recovery technique used in HTD-based scan registration algorithm divide the rotation recovery into two steps: yaw angle determination and roll & pitch angle estimation. This method is based on the resampling of the spherical images and is not theoretical complete. But it is effective and efficient in robotic mapping scenarios. The generalized convolution theorem based on SHT is applied to determine the rotation between spherical images in SEI-based scan registration algorithm. It is theoretical complete but play poor for sparse images such as HTD, so it could not be used in HTD-based scan registration. From the view of performance, the HTD-based registration method performs well in robotic mapping, and the existence of planes helps it work well. While the SEI-based registration method is more general and has broad applicable scenarios, but its precision is lower than HTD-based registration method.

Generally speaking, our work has the following major contributions:

- develop a novel Hough Transform Descriptor based registration method, which outperforms the state-of-the-art registration methods.
- propose a new transformation recovery technique based on Phase Only Matched Filter (POMF), the superiority of our translation recovery technique dues to rendering the original 3D point clouds into a plane and a vector separately instead of a single volume. This is a far reaching contribution since it improves the performance of translation recovery of point clouds, which could be used by other registration methods which estimate rotation and translation separately.
- invent a novel shape descriptor named Spherical Entropy Image (SEI), which inherits both the descriptive power of signature-based descriptors and the robustness of histogram-based descriptors. The possibility of using SEI as a local descriptor is validated and the experiments prove that SEI performs better than the state-of-the-art descriptors in the terms of feature matching application.
- a novel feature-less global registration algorithm is proposed based on SEI and Spherical Harmonic analysis techniques. Meanwhile, we propose an iteration framework to solve the **Chicken-and-Egg** problem. This is also a far-ranging contribution, which could be adopted by all the registration methods based on Spherical Harmonic analysis techniques.

5.2 Future research directions

View-based 3D shape descriptors use a set of rendered views to represent a 3D model. Since image processing has been investigated for many decades, the view-based 3D shape descriptors could benefit from existing image processing technologies. View-based 3D shape descriptors attract more and more attention recently because they perform well in content-based 3D object retrieval. Content-based 3D object retrieval is an active research field that has attracted a significant amount of attention in recent years. This is because the problem of searching for existing 3D models become necessary and urgent as the increasing number of 3D objects available in public or proprietary databases.

For the non-rigid 3D object retrieval, the traditional hand-crafted shape descriptors are difficult to describe the shape precisely and effectively. While the features achieved by deep learning techniques could play an important role in non-rigid model processing. In our opinion, view-based 3D shape descriptors employing convolutional neural network (CNN) may be a good option for this goal. CNN is a quite famous machine learning technique, which achieves roaring success in many applications. In the future, we will develop the view-based 3D shape descriptors based on CNN, and try to apply them into 3D shape retrieval application.

References

[1] Adept Mobile Robots (2012). Pioneer 3-AT. http://www.mobilerobots.com/researchrobots/p3at.aspx. Retrieved May 4, 2015.

[2] Almeida, L. (1994). The fractional Fourier transform and time-frequency representations. *IEEE Transactions on Signal Processing*, 42(11):3084–3091.

[3] Andreasson, H. and Lilienthal, A. (2010). 6D scan registration using depth-interpolated local image featues. *Robotics and Autonomous Systems*, 58(2):157–165.

[4] Ankerst, M., Kastenmüller, G., Kriegel, H.-P., and Seidl, T. (1999). 3D shape histograms for similarity search and classification in spatial databases. In *6th International Symposium, SSD'99*, pages 207–226. Springer.

[5] Assfalg, J., Bertini, M., Bimbo, A., and Pala, P. (2007). Content-based retrieval of 3-D objects using Spin Image signatures. *Multimedia, IEEE Transactions on*, 9(3):589–599.

[6] Averbuch, A. and Shkolnisky, Y. (2003). 3D Fourier based discrete Radon transform. *Applied and Computational Harmonic Analysis*, 15:33–69.

[7] Bailey, D. and Swarztrauber, P. (1991). The fractional Fourier transform and applications. *SIAM Review*, 33(3):389–404.

[8] Ballard, D. (1981). Generalizing the Hough Transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111 – 122.

[9] Bay, H., Ess, A., Tuytelaars, T., and Gool, L. V. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3):346 – 359. Similarity Matching in Computer Vision and Multimedia.

[10] Bayramoglu, N. and Alatan, A. (2010). Shape Index SIFT: Range image recognition using local features. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 352–355.

[11] Belongie, S., Malik, J., and Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522.

[12] Besl, P. and McKay, N. (1992). A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Machine Intell.*, 14(2):239–256.

[13] Biasotti, S., Giorgi, D., Marini, S., Spagnuolo, M., and Falcidieno, B. (2006). A comparison framework for 3D object classification methods. In Gunsel, B., Jain, A., Tekalp, A., and Sankur, B., editors, *Multimedia Content Representation, Classification and Security*, volume 4105 of *Lecture Notes in Computer Science*, pages 314–321. Springer Berlin Heidelberg.

[14] Blanco, J.-L., Fernandez-Madrigal, J.-A., and Gonzalez, J. (2008). Efficient probabilistic range-only SLAM. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1017–1022.

[15] Bôcher, M. (1915). *Plane Analytic Geometry: With Introductory Chapters on the Differential Calculus*. H. Holt.

[16] Borrmann, D., Elseberg, J., Lingemann, K., and Nüchter, A. (2011). The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2(2).

[17] Bourke, P. (2015). Geometric data formats. http://paulbourke.net/dataformats/. Retrieval June 1, 2015.

[18] Bro, R., Acar, E., Kolda, T., Bro, R., and Acar, E. (2008a). Resolving the sign ambiguity in the Singular Value Decomposition. *Journal of Chemometrics*, 22(2):135–140.

[19] Bro, R., Acar, E., and Kolda, T. G. (2008b). Resolving the sign ambiguity in the Singular Value Decomposition. *Journal of Chemometrics*, 22(2):135–140.

[20] Bronstein, M. and Kokkinos, I. (2010). Scale-invariant heat kernel signatures for non-rigid shape recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1704–1711.

[21] Bulow, H. and Birk, A. (2013). Spectral 6DOF registration of noisy 3D range data with partial overlap. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(4):954–969.

[22] Bülow, H. and Birk, A. (2011). Spectral registration of noisy sonar data for underwater 3D mapping. *Autonomous Robots*, 30(3):307–331.

[23] C, H. P. V. (1962). Method and means for recognizing complex patterns. Google Patents, http://www.google.com/patents/US3069654. US Patent 3,069,654.

[24] Campbell, R. J. and Flynn, P. J. (2001). A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding*, 81(2):166 – 210.

[25] Censi, A. and Carpin, S. (2009). HSM3D:feature-less global 6DOF scan-mathing in the Hough/Radon domain. In *Proc. IEEE International Conference on Robotics and Automation(ICRA)*, pages 3899–3906, Kobe, Japan.

[26] Chen, D., Tian, X., Shen, Y., and Ouhyoung, M. (2003). On visual similarity based 3D model retrieval. *Comput. Graph. Forum*, 22(3):223–232.

[27] Chen, H. and Bhanu, B. (2004). 3D free-form object recognition in range images using local surface patches. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 136–139 Vol.3.

[28] Chen, H. and Bhanu, B. (2007). 3D free-form object recognition in range images using local surface patches. *Pattern Recogn. Lett.*, 28(10):1252–1262.

[29] Chen, Q.-S., Defrise, M., and Deconinck, F. (1994). Symmetric phase-only matched filtering of Fourier-Mellin transforms for image registration and recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 16(12):1156–1168.

[30] Chen, Y. and Medioni, G. (1992). Object modeling by registration of multiple range images. *Image and Vision Computing*, 10(3):145–155.

[31] Chua, C. and Jarvis, R. (1997a). Point Signatures: A new representation for 3D object recognition. *International Journal of Computer Vision*, 25(1):63–85.

[32] Chua, C. and Jarvis, R. (1997b). Point Signatures: A new representation for 3D object recognition. *International Journal of Computer Vision*, 25(1):63–85.

[33] Coda Octopus Products Ltd. (2013). Echoscope is the world's highest definition real time 3d imging sonar. http://www.codaoctopus.com/products/echoscope. Retrieval June 1, 2015.

[34] Curless, B. and Levoy, M. (1996). A volumetric method for building complex models from range images. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA. ACM.

[35] Daras, P. and Axenopoulos, A. (2010). A 3D shape retrieval framework supporting multimodal queries. *International Journal of Computer Vision*, 89(2-3):229–247.

[36] Darom, T. and Keller, Y. (2012). Scale-invariant features for 3-d mesh models. *Image Processing, IEEE Transactions on*, 21(5):2758–2769.

[37] DAVID Group (2007-2015). DAVID SLS-2. http://www.david-3d.com/en/products/sls-2. Retrieval June 1, 2015.

[38] De Castro, E. and Morandi, C. (1987). Registration of translated and rotated images using Finite Fourier Transforms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-9(5):700–703.

[39] DICOM (2015). DICOM: Digital Imaging and Communications in Medicine. http://dicom.nema.org/. Retrieval June 1, 2015.

[40] D.M.Healy, Jr., D.N.Rockmore, P.J.Kostelec, and S.Moore (2003). FFTs for the 2-sphere-improvements and variations. *The Journal of Fourier Analysis and Applications* , 9(4):341–385.

[41] Dorai, C. and Jain, A. (1997). COSMOS-A representation scheme for 3D free-form objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(10):1115–1130.

[42] Duda, R. O. and Hart, P. E. (1972). Use of the Hough Transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15.

[43] Elseberg, J., Magnenat, S., Siegwart, R., and Nüchter, A. (2012). Comparison of nearest-neighbor-search strategies and implementations for efficient shape registration. *Journal of Software Engineering for Robotics (JOSER)*, 3(1):2–12.

[44] Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395.

[45] Flint, A., Dick, A., and van den Hengel, A. (2007). ThRIFT: Local 3D structure recognition. In *Digital Image Computing Techniques and Applications, 9th Biennial Conference of the Australian Pattern Recognition Society on*, pages 182–188.

[46] Flint, A., Dick, A., and van den Hengel, A. (2008). Local 3D structure recognition in range images. *Computer Vision, IET*, 2(4):208–217.

[47] FLIR (2013). Pan-Tilt Unit-D48E. http://www.flir.com/mcs/view/?id=53670. Retrieved April 22, 2015.

[48] Foroosh, H., B.Zerubia, J., and Berthod, M. (2002). Extension of Phase Correlation to subpixel registration. *IEEE Transactions on Image Processing*, 11(3):188–200.

[49] FOTONIC (2015). SMART 3D-CAMERAS. http://www.fotonic.com/content/Products/fotonic-products-e-series.aspx. Retrieval June 1, 2015.

[50] Frigo, M. and Johnson, S. G. (2005). The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[51] Frome, A., Huber, D., Kolluri, R., Bülow, T., and Malik, J. (2004). Recognizing objects in range data using regional point descriptors. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 224–237.

[52] Furuya, T. and Ohbuchi, R. (2009). Dense sampling and fast encoding for 3D model retrieval using Bag-of-visual features. In *Proceedings of the ACM International Conference on Image and Video Retrieval*, CIVR '09, pages 26:1–26:8, New York, NY, USA. ACM.

[53] Guennebaud, G., Jacob, B., et al. (2010). Eigen v3. http://eigen.tuxfamily.org. Retrieved April 22, 2013.

[54] Guo, Y., Bennamoun, M., Sohel, F., Lu, M., and Wan, J. (2014). 3D object recognition in cluttered scenes with local surface features: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(11):2270–2287.

[55] Hokuyo (2009). UTM-30LX. http://www.hokuyo-aut.jp/02sensor/07scanner/utm_30lx.html. Retrieved April 22, 2015.

[56] Horn, B. K. P. (1984). Extended Gaussian images. *Proceedings of the IEEE*, 72(2):1671–1686.

[57] Horner, J. L. and Gianino, P. D. (1984). Phase-only matched filtering. *Applied Optics*, 23(6):812–816.

[58] Hough, P. (1959). Machine Analysis Of Bubble Chamber Pictures. *Proc. Int. Conf. High Energy Accelerators and Instrumentation*, C590914:554–558.

[59] Hua, J., Lai, Z., Dong, M., Gu, X., and Qin, H. (2008). Geodesic distance-weighted shape vector image diffusion. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1643–1650.

[60] Huhle, B., Magnusson, M., W.Strasser, and Lilienthal, A. (2008). Registration of colored 3D point clouds with a kernel-based extension to the Normal Distributions Transform. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 4025–4030, Pasadena, CA, USA.

[61] Illingworth, J. and Kittler, J. (1987). The adaptive Hough Transform.*Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 9(5):690–698.

[62] Illingworth, J. and Kittler, J. (1988). A survey of the Hough Transform. *Comput. Vision Graph. Image Process.*, 44(1):87–116.

[63] J.Kostelec, P. and N.Rockmore, D. (2008). FFTs on the rotation group. *Journal of Fourier Analysis and Applications*, 14:145–179.

[64] J.Kostelec, P. and N.Rockmore, D. (2014). The SOFT package:FFTs on the rotation group. "http://www.cs.dartmouth.edu/~geelong/soft/". Retrieved April 22, 2015.

[65] Johnson, A. and Hebert, M. (1999). Using spin Images for efficient object recognition in cluttered 3D scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5):433–449.

[66] Johnson, A. E. and Hebert, M. (1998). Surface matching for object recognition in complex 3-D scenes. *Image and Vision Computing*, 16:635–651.

[67] Jolliffe, I. (2002). *Principal Component Analysis*. Springer Series in Statistics. Springer.

[68] Kazhdan, M., Chazelle, B., Dobkin, D., Finkelstein, A., and Funkhouser, T. (2002). A reflective symmetry descriptor. In Heyden, A., Sparr, G., Nielsen, M., and Johansen, P., editors, *Computer Vision — ECCV 2002*, volume 2351 of *Lecture Notes in Computer Science*, pages 642–656. Springer Berlin Heidelberg.

[69] Kazhdan, M., Chazelle, B., Dobkin, D., Funkhouser, T., and Rusinkiewicz, S. (2004). A reflective symmetry descriptor for 3d models. *Algorithmica*, 38(1):201–225.

[70] Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. (2003). Rotation invariant Spherical Harmonics representation of 3D shape descriptors. In*Proc. Eurographics Symposium on Geometry Processing*, Aachen, Germany.

[71] Keller, Y., Averbuch, A., and Shkolnisky, Y. (2005a). Algebraically accurate volume registration using Euler's theorem and 3D pseudo-polar FFT. *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, 24(11).

[72] Keller, Y., Averbuch, A., and Shkolnisky, Y. (2005b). Algebraically accurate volume registration using Euler's theorem and the 3D pseudo-polar FFT. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, pages 795–800.

[73] Keller, Y., Shkolnisky, Y., and Averbuch, A. (2006). Volume registration using the 3-D pseudopolar Fourier Transform. *Signal Processing, IEEE Transactions on*, 54(11):4323–4331.

[74] Kiryati, N., Eldar, Y., and Bruckstein, A. M. (1991). A probabilistic Hough Transform. *Pattern Recogn.*, 24(4):303–316.

[75] Kitware (2015). ParaView. http://www.paraview.org/. Retrieval June 1, 2015.

[76] Knopp, J., Prasad, M., Willems, G., Timofte, R., and Van Gool, L. (2010). Hough Transform and 3D SURF for robust three dimensional classification. In *Proceedings of the 11th European Conference on Computer Vision: Part VI*, ECCV'10, pages 589–602.

[77] Kovnatsky, A., Bronstein, M., Bronstein, A., and Kimmel, R. (2012a). Photometric heat kernel signatures. In Bruckstein, A., ter Haar Romeny, B., Bronstein, A., and Bronstein, M., editors, *Scale Space and Variational Methods in Computer Vision*, volume 6667 of *Lecture Notes in Computer Science*, pages 616–627. Springer Berlin Heidelberg.

[78] Kovnatsky, A., Bronstein, M. M., Bronstein, A. M., Raviv, D., and Kimmel, R. (2012b). Affine-invariant photometric heat kernel signatures. In *Proceedings of the 5th Eurographics Conference on 3D Object Retrieval*, EG 3DOR'12, pages 39–46, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

[79] Lai, K., Bo, L., Ren, X., and Fox, D. (2011). A large-scale hierarchical multi-view RGB-D object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824.

[80] Larobina, M. and Murino, L. (2014). Medical image file formats. *Journal of Digital Imaging*, 27(2):200–206.

[81] Laser Measurement Systems GmbH, R. (2015). RIEGL. http://www.riegl.com/. Retrieval June 1, 2015.

[82] Leavers, V. F. (1992). *Shape Detection in Computer Vision Using the Hough Transform*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[83] Leica and HEXAGON (2015). Leica Geosystems. http://hds.leica-geosystems.com/en/index.htm. Retrieval June 1, 2015.

[84] Li, B., Godil, A., Aono, M., Bai, X., Furuya, T., Li, L., López-Sastre, R., Johan, H., Ohbuchi, R., Redondo-Cabrera, C., Tatsuma, A., Yanagimachi, T., and Zhang, S. (2012). SHREC'12 Track: Generic 3D Shape Retrieval. In *Proceedings of the 5th Eurographics Conference on 3D Object Retrieval*, EG 3DOR'12, pages 119–126, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

[85] Li, X. and Guskov, I. (2005). Multi-scale features for approximate alignment of point-based surfaces. In *Proceedings of the Third Eurographics Symposium on Geometry Processing*, SGP ’05, pages 217:1–217:11, Aire-la-Ville, Switzerland, Switzerland. Eurographics Association.

[86] Liu, K., Skibbe, H., Schmidt, T., Blein, T., Palme, K., Brox, T., and Ronneberger, O. (2013). Rotation-Invariant HOG descriptors using Fourier Analysis in polar and spherical coordinates. *International Journal of Computer Vision*, 106(3):342–364.

[87] LMI Technology Inc. (2015). HDI Advanced 3D Scanner for 3D Modeling in R&D, Lab and Educational Applications. http://www.lmi3d.com/products/hdi-advance. Retrieval June 1, 2015.

[88] Lo, T.-W. R. and Siebert, J. P. (2009). Local feature extraction and matching on range images: 2.5D SIFT. *Computer Vision and Image Understanding*, 113(12):1235 – 1250. Special issue on 3D Representation for Object and Scene Recognition.

[89] Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

[90] Lucchese, L. and Cortelazzo, G. (1997). Noise-robust estimation of planar roto-translations with high precision. In *Image Processing, 1997. Proceedings., International Conference on*, volume 1, pages 699–702.

[91] Lucchese, L. and Cortelazzo, G. (2000). A noise-robust frequency domain technique for estimating planar roto-translations. *Signal Processing, IEEE Transactions on*, 48(6):1769–1786.

[92] Lucchese, L., Cortelazzo, G., and Monti, C. (1996). A frequency domain technique for estimating rigid planar rotations. In *Circuits and Systems, 1996. ISCAS ’96., Connecting the World., 1996 IEEE International Symposium on*, volume 2, pages 774–777.

[93] Lucchese, L., Doretto, G., and Cortelazzo, G. (1997). Frequency domain estimation of 3-D rigid motion based on range and intensity data. In *3-D Digital Imaging and Modeling, 1997. Proceedings., International Conference on Recent Advances in*, pages 107–112.

[94] Lucchese, L., Doretto, G., and Cortelazzo, G. (2002). A frequency domain technique for range data registration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(11):1468–1484.

[95] Magnusson, M., Lilienthal, A., and Duckett, T. (2007). Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10):803–827.

[96] Makadia, A., IV, A. P., and Daniilidis, K. (2006). Fully automatic registration of 3D point clouds. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, pages 1297–1304, New York,USA.

[97] Malassiotis, S. and Strintzis, M. (2007). Snapshots: A novel local surface descriptor and matching algorithm for robust 3D surface alignment. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(7):1285–1290.

[98] Novatnack, J. and Nishino, K. (2008). Scale-Dependent/Invariant Local 3D Shape Descriptors for fully automatic registration of multiple sets of range images. In Forsyth, D., Torr, P., and Zisserman, A., editors, *Computer Vision – ECCV 2008*, volume 5304 of *Lecture Notes in Computer Science*, pages 440–453. Springer Berlin Heidelberg.

[99] Novotni, M. and Klein, R. (2004). Shape retrieval using 3D Zernike descriptors. *Computer Aided Design*, 36(11):1047–1062.

[100] Ohbuchi, R. and Shimizu, T. (2008). Ranking on semantic manifold for shape-based 3D model retrieval. In *Proceedings of the 1st ACM International Conference on Multimedia Information Retrieval*, MIR '08, pages 411–418, New York, NY, USA. ACM.

[101] Osada, R., Funkhouser, T., Chazelle, B., and Dobkin, D. (2002). Shape distributions. *ACM Transactions on Graphics*, 21(4):807–832.

[102] Pandey, G., McBride, J. R., and Eustice, R. M. (2011). Ford campus vision and lidar data set. *International Journal of Robotics Research*, 30(13):1543–1552.

[103] Papadakis, P., Pratikakis, I., Theoharis, T., and Perantonis, S. (2010). PANORAMA: A 3D shape descriptor based on panoramic views for unsupervised 3D object retrieval. *International Journal of Computer Vision*, 89(2-3):177–192.

[104] Parzen, E. (1962). On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076.

[105] Pathak, K., Birk, A., Vaskevicius, N., and Poppinga, J. (2010). Fast registration based on noisy planes with unknown correspondences for 3-D mapping. *IEEE Transactions on Robotics*, 26(3):424–441.

[106] PCL (2015a). Point Cloud Data (PCD). http://pointclouds.org/documentation/tutorials/pcd_file_format.php.Retrieval June 1, 2015.

[107] PCL (2015b). Point Cloud Library. http://www.pointclouds.org. Retrieval April 1, 2015.

[108] Planitz, B., Maeder, A., and Williams, J. (2005). The correspondence framework for 3D surface matching algorithms. *Computer Vision and Image Understanding*, 97(3):347 – 383.

[109] PMD Technologies GmbH (2014). PMD CamCube 3.0. http://www.pmdtec.com/news_media/video/camcube.php. Retrieval June 1, 2015.

[110] Point Grey Research Inc. (2015). Bumblebee XB3 1394b. http://www.tritech.co.uk/product/multibeam-sonar-for-3d-model-view-of-sonar-imagery-eclipse. Retrieval June 1, 2015.

[111] Pomerleau, F., Colas, F., Siegwart, R., and Magnenat, S. (2013). Comparing ICP variants on real-world data sets. *Autonomous Robots*, 34(3):133–148.

[112] Pomerleau, F., Liu, M., Colas, F., and Siegwart, R. (2012). Challenging data sets for point cloud registration algorithms. *The International Journal of Robotics Research*, 31(14):1705–1711.

[113] Rabbani, T. and Heuvel, F. V. D. (2005). Efficient Hough Transform for automatic detection of cylinders in point clouds. In *Pro. of 11th Annual Conference of the Advanced School for Computing and Imaging (ASCI'05)*, pages 60–65, The Netherlands.

[114] Raviv, D., Bronstein, M. M., Bronstein, A. M., and Kimmel, R. (2010). Volumetric heat kernel signatures. In *Proceedings of the ACM Workshop on 3D Object Retrieval*, 3DOR '10, pages 39–44, New York, NY, USA. ACM.

[115] Reddy, B. and Chatterji, B. N. (1996). An FFT-based technique for translation, rotation, and scale-invariant image registration. *Image Processing, IEEE Transactions on*, 5(8):1266–1271.

[116] Roni (2008). DICOM is easy. http://dicomiseasy.blogspot.co.il/p/introduction-to-dicom.html. Retrieval June 1, 2015.

[117] Ruiz-Correa, S., Shapiro, L., and Melia, M. (2001). A new signature-based method for efficient 3-D object recognition. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* , volume 1, pages I–769–I–776 vol.1.

[118] Rusinkiewicz, S. and M.Levoy (2001). Efficient variants of the ICP algorithm. In *Proc. 3rd International Conference on 3D Digital Imaging and Modeling(3DIM)*, Quebec City, Canada.

[119] Rusu, R., Blodow, N., and Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. In *Proc. IEEE International Conference on Robotics and Automation (ICRA)*, pages 3212–3217, Kobe, Japan.

[120] Rusu, R. B., Beetz, M., Marton, Z. C., Blodow, N., and Dolha, M. (2008a). Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems Journal (Special Issue on Semantic Knowledge)*, 56(11):927–941.

[121] Rusu, R. B. and Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In*IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China.

[122] Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., and Beetz, M. (2008b). Towards 3D point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927 – 941.

[123] Salti, S., Tombari, F., and Stefano, L. D. (2014). SHOT: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125(0):251 – 264.

[124] Salvi, J., Matabosch, C., Fofi, D., and Forest, J. (2007). A review of recent range image registration methods with accuracy evaluation. *Image Vision Comput.*, 25(5):578–596.

[125] Sethian, J. (2006). Fast Marching Methods: A boundary value formulation. https://math.berkeley.edu/~sethian/2006/Explanations/fast_marching_explain.html. Retrieval June 23, 2015.

[126] Shah, T. (2006). *Automatic reconstruction of industrial installations using point clouds and images*. Publications on Geodesy. Nederlander Commissie voor Geodesie.

[127] Shilane, P., Min, P., Kazhdan, M., and Funkhouser, T. (2004). The Princeton Shape Benchmark. In *Proceedings of the Shape Modeling International 2004*, SMI '04, pages 167–178, Washington, DC, USA. IEEE Computer Society.

[128] SICK AG (2015). 3D laser scanners. http://www.sick.com/group/EN/home/products/product_portfolio/laser_measurement_systems/Pages/outdoor_laser_measurement_technology.aspx. Retrieval June 1, 2015.

[129] Skibbe, H., Reisert, M., Schmidt, T., Brox, T., and Ronneberger, O. (2012). Fast rotation invariant 3D feature computation utilizing efficient local neighborhood operators. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(8):1563–1575.

[130] Steder, B., Rusu, R. B., Konolige, K., and Burgard, W. (2011). Point feature extraction on 3D range scans taking into account object boundaries. In *Proceeding of IEEE International Conference on Robotics and Automation*, pages 2601–2608, Shanghai, China.

[131] STEREOLABS (2015). ZED 2K Stereo Camera. https://www.stereolabs.com/. Retrieval June 1, 2015.

[132] Stockman, G. and Shapiro, L. G. (2001). *Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition.

[133] S.U.C.G Laboratory (2015). The stanford 3D scanning repository. http://graphics.stanford.edu/data/3Dscanrep/. Retrieval April 1, 2015.

[134] Sun, B., Kong, W., Xiao, J., and Zhang, J. (2014a). A Hough Transform based scan registration strategy for mobile robotic mapping. In *Proceeding of IEEE International Conference and Robotics and Automation*, pages 4612–4619, Hongkong, China.

[135] Sun, B., Kong, W., Zhang, L., and Zhang, J. (2014b). Fourier analysis techniques applied in data registration: A survey. In *Multisensor Fusion and Information Integration for Intelligent Systems (MFI), 2014 International Conference on*, pages 1–5.

[136] Sun, J., Ovsjanikov, M., and Guibas, L. (2009). A concise and provably informative multi-scale signature based on heat diffusion. *Computer Graphics Forum*, 28(5):1383–1392.

[137] Sun, Y. and Abidi, M. (2001). Surface matching by 3D point's fingerprint. In *Computer Vision, 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 263–269.

[138] Sun, Y., Paik, J., Koschan, A., Page, D. L., and Abidi, M. A. (2003). Point Fingerprint: A new 3-D object representation scheme. *Trans. Sys. Man Cyber. Part B*, 33(4):712–717.

[139] Taati, B., Bondy, M., Jasiobedzki, P., and Greenspan, M. (2007). Variable dimensional local shape descriptors for object recognition in range data. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8.

[140] Taati, B. and Greenspan, M. (2011). Local shape descriptor selection for object recognition in range data. *Computer Vision and Image Understanding*, 115(5):681 – 694. Special issue on 3D Imaging and Modelling.

[141] Takeuchi, E. and Tsubouchi, T. (2006). A 3D scan matching using improved 3D Normal Distributions Transform for mobile robotic mapping. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systerms(IROS)*, pages 3068–3073, Beijing, China.

[142] Tang, S., Wang, X., Lv, X., Han, T., Keller, J., He, Z., Skubic, M., and Lao, S. (2013). Histogram of oriented normal vectors for object recognition with a depth sensor. In Lee, K., Matsushita, Y., Rehg, J., and Hu, Z., editors, *Computer Vision – ACCV 2012*, volume 7725 of *Lecture Notes in Computer Science*, pages 525–538. Springer Berlin Heidelberg.

[143] Tangelder, J. and Veltkamp, R. (2008). A survey of content based 3D shape retrieval methods. *Multimedia Tools and Applications*, 39(3):441–471.

[144] Tatsuma, A. and Aono, M. (2009). Multi-Fourier spectra descriptor and augmentation with spectral clustering for 3D shape retrieval. *The Visual Computer*, 25(8):785–804.

[145] Tatsuma, A., Koyanagi, H., and Aono, M. (2012a). A large-scale shape benchmark for 3D object retrieval: Toyohashi shape benchmark. In *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pages 1–10.

[146] Tatsuma, A., Koyanagi, H., and Aono, M. (2012b). A large-scale shape benchmark for 3D object retrieval: Toyohashi shape benchmark. In *Signal Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*, pages 1–10.

[147] The MathWorks, I. (1994). MATLAB: The language of technical computing. http://de.mathworks.com/products/matlab/. Retrieval June 1, 2015.

[148] Tombari, F. and Di Stefano, L. (2010). Object recognition in 3D scenes with occlusions and clutter by Hough voting. In *Image and Video Technology (PSIVT), 2010 Fourth Pacific-Rim Symposium on*, pages 349–355, Singapore.

[149] Tombari, F., Salti, S., and Di Stefano, L. (2010a). Unique Shape Context for 3D data description. In *Proceedings of the ACM Workshop on 3D Object Retrieval*, 3DOR '10, pages 57–62, New York, NY, USA. ACM.

[150] Tombari, F., Salti, S., and Stefano, L. D. (2010b). Unique signatures of histograms for local surface description. In *Proceeding of 11th European Conference on Computer Vision*, ECCV'10, pages 356–369, Berlin, Heidelberg. Springer-Verlag.

[151] Tombari, F. and Stefano, L. D. (2012). Hough voting for 3D object recognition under occlusion and clutter. *IPSJ Transactions on Computer Vision and Applications*, 4(0):20–29.

[152] Tong, C. H., Gingras, D., Larose, K., Barfoot, T. D., and Dupuis, E. (2013). The canadian planetary emulation terrain 3D mapping dataset. *Int. J. Rob. Res.*, 32(4):389–395.

[153] Trimble Navigation Limited (2015). Trimble. Transforming the way the world works. http://www.trimble.com/3d-laser-scanning/index.aspx. Retrieval June 1, 2015.

[154] Tritech International Limited (2015). Eclipse-Multibeam Sonar For 3D Model View of sonar Imagery. http://www.tritech.co.uk/product/multibeam. Retrieval June 3, 2015.

[155] Turk, G. (1994). Polygon File Format. http://www.dcs.ed.ac.uk/teaching/cs4/www/graphics/Web/ply.html. Retrieval June 1, 2015.

[156] van Kaick, O., Zhang, H., Hamarneh, G., and Cohen-Or, D. (2011). A survey on shape correspondence. *Computer Graphics Forum*, 30(6):1681–1707.

[157] Vosselman, G., Dijkman, S., et al. (2001). 3D building model reconstruction from point clouds and ground planes. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(3/W4):37–44.

[158] Vranic, D. (2005). DESIRE: A composite 3d-shape descriptor. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 4 pp.–.

[159] Wahl, E., Hillenbrand, U., and Hirzinger, G. (2003). Surflet-pair-relation histograms: A statistical 3D-shape representation for rapid classification. In*3-D Digital Imaging and Modeling, 2003. 3DIM 2003. Proceedings. Fourth International Conference on*, pages 474–481.

[160] WIKIPEDIA (2013). Graphics file formats. https://en.wikipedia.org/wiki/Category:Graphics_file_formats. Retrieval June 1, 2015.

[161] WIKIPEDIA (2015). Kinect. https://en.wikipedia.org/wiki/Kinect. Retrieval June 1, 2015.

[162] Wolfram Research, I. (2015). *Mathematica*. Wolfram Research, Inc, Champaign, Illinois, version 10.1 edition.

[163] Xiao, J., Adler, B., Zhang, H., and Zhang, J. (2013a). Planar segment based three-dimensional point cloud registration in outdoor environments. *Journal of Field Robotics*, 30(4):552–582.

[164] Xiao, J., Zhang, J., Adler, B., Zhang, H., and Zhang, J. (2013b). Three-dimensional point cloud plane segmentation in both structured and unstructured environments. *Robotics and Autonomous Systems*, 61(12):1641 – 1652.

[165] Yamany, S. and Farag, A. (2002). Surface signatures: an orientation independent free-form surface representation scheme for the purpose of objects registration and matching. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(8):1105–1120.

[166] Yuen, H. K., Princen, J., Illingworth, J., and Kittler, J. (1989). A comparative study of Hough Transform methods for circle finding. In *Proc. 5th Alvey Vision Conf.*, pages 291–296.

[167] Yuen, H. K., Princen, J., Illingworth, J., and Kittler, J. (1990). Comparative study of Hough Transform methods for circle finding. *Image Vision Comput.*, 8(1):71–77.

[168] Zaharescu, A., Boyer, E., and Horaud, R. (2012). Keypoints and local descriptors of scalar functions on 2D manifolds. *International Journal of Computer Vision*, 100(1):78–98.

[169] Zaharescu, A., Boyer, E., Varanasi, K., and Horaud, R. (2009). Surface feature detection and description with applications to mesh matching. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 373–380.

[170] Zarpalas, D., Daras, P., Axenopoulos, A., Tzovaras, D., and Strintzis, M. G. (2007). 3D model search and retrieval using the Spherical Trace Transform. *EURASIP Journal on Advances in Signal Processing*, 2007(1):1–14.

[171] Zhang, X. and Tan, C. (2013). Segmentation-free keyword spotting for handwritten documents based on heat kernel signature. In *Document Analysis and Recognition (ICDAR), 2013 12th International Conference on*, pages 827–831.

[172] Zhong, Y. (2009). Intrinsic shape signatures: A shape descriptor for 3D object recognition. In *Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference on*, pages 689–696.

APPENDIX A

Code and data to reproduce the experimental results

We publish the code and data which could be used to reproduce the experimental results in this thesis online. That is because we believe making the code online at least has the following benefits:

1 We believe the code would tell everything. Every detail of our algorithm is contained in the code without exception.

2 As Eric Raymond's *Linus's Law* says, "**given enough eyeballs, all bugs are shallow**". By opening the code to public, we get higher chances to fixed the potential bugs.

3 In case other researcher interested, they could create their new work based on our code. It is delightful to informed that our algorithms could be used in other area which we have not foreseen.

The code and datasets to reproduce the experimental results could be found on the following URL:

1. **Examples of HTD based on oriented points:** this package is related to Section 3.3.3 of this thesis, and available on https://bitbucket.org/bo_sun/htd_oriented/downloads. Or if **Git** is installed, enter the following command in terminal:
   git clone https://bo_sun@bitbucket.org/bo_sun/htd_oriented.git

2. **Examples of Translation recovery:** this package is related to Section 3.4.2 of this thesis, and available on https://bitbucket.org/bo_sun/translationrecovery/downloads. Or if **Git** is installed, enter the following command in terminal:
   git clone https://bo_sun@bitbucket.org/bo_sun/translationrecovery.git

3. **HTD based scan registration:** this package contains the code and datasets to reproduce the experiment results of Section 3.5 of this thesis, and is available on https://bitbucket.org/bo_sun/htd_registration/downloads. Or if **Git** is installed, enter the following command in terminal:
   git clone https://bo_sun@bitbucket.org/bo_sun/htd_registration.git

4. **SEI based scan registration:** this package is related to Section 4.5 of my thesis, and contains the data and code to reproduce the experiments results about employing SEI as a global shape descriptor in scan registration. It is available on https://bitbucket.org/bo_sun/sei_registration/downloads. Or if **Git** is installed, enter the following command in terminal:
   git clone https://bo_sun@bitbucket.org/bo_sun/sei_registration.git

5. **SEI based feature matching:** this package is related to Section 4.6 of my thesis. It contains the datasets and code to reproduce the experiment results. It is available on https://bitbucket.org/bo_sun/sei_featurematching/downloads. Or if **Git** is installed, enter the following command in terminal:
   git clone https://bo_sun@bitbucket.org/bo_sun/sei_featurematching.git

Other code which are not employed in the experiments, for example the rotation invariant SEI based on SHT and the SEI based scan registration aided by 3D FFT, are available on requirement: *bo.sun.sd@foxmail.com* The code about the state-of-the-art methods used in comparison experiments are also available on requirement.

APPENDIX B

Related concepts

3D rotation representation

There are many ways to represent the rotation in 3D space, but four of them are quite popular and widely used: rotation matrix, rotation vector, Euler angles and rotation quaternions. And naturally, the four representations could be converted into each other.

Rotation Matrix: Any rotations in 3D space could be represented by a 3×3 orthogonal matrix. By the way, the orthogonal matrix means its columns are unit vectors and orthogonal to each other. We could extract 6 equations which limits a matrix to be a orthogonal matrix, which means the rotation matrix only have 3 degrees of freedom.

Rotation Vector (Axis-angle representation): *Euler's rotation theorem* states that any displacement of a rigid object in 3D space could be expressed by a single rotation about a fixed axis. In other words, the 3D rotation could be represented by a four-element vector, of which three elements indicate the direction of the rotation axis and one element means the magnitude of rotation. And the first three elements should be a unit vector, which means they only have two degrees of freedom. Again the rotation vector totally has 3 degrees of freedom.

Euler Angles: A rotation could be split into three simpler constitutive rotations, and each of them could be expressed by an axis of reference system and an Euler angle around the axis. In this way, any rotation could be described by three Euler angles. Unfortunately, the different sequences of axes three Euler angles applied on result in different rotation since the rotations are not commutative. Commonly, the sequence of axes according to Euler angles is one of the following: $x-y-z$, $z-x-z$, $x-y-x$, $y-z-y$, $z-y-z$, $x-z-x$ and

$y-x-y$. For example, the SOFT(2) library [63] adopt the $z-y-z$ Euler angles to describe the rotation.

Rotation Quaternions: It is naturally to introduce rotation quaternions based on the axis-angle representation. Quaternions give a simple way to encode the rotation axis-angle representation by four numbers. For a rotation around axis $v = v_x i + v_y j + v_z k$ through angle θ could be expressed by a normalized quaternion: $\hat{q} = [q_i q_j q_k q_s]^T$, where $q_i = v_x sin(\frac{\theta}{2})$, $q_j = v_y sin(\frac{\theta}{2})$, $q_k = v_z sin(\frac{\theta}{2})$ and $q_s = cos(\frac{\theta}{2})$.

Spherical coordinates nomenclature

A point P in 3D space could be indexed by three numbers as Figure B.1 describes:**radial distance r** measuring distance between the point and origin; **polar angle** θ measuring the angle from positive z-axis to P; **azimuth angle** φ measuring the angle from x-axis to the projection of P onto xy-plane.

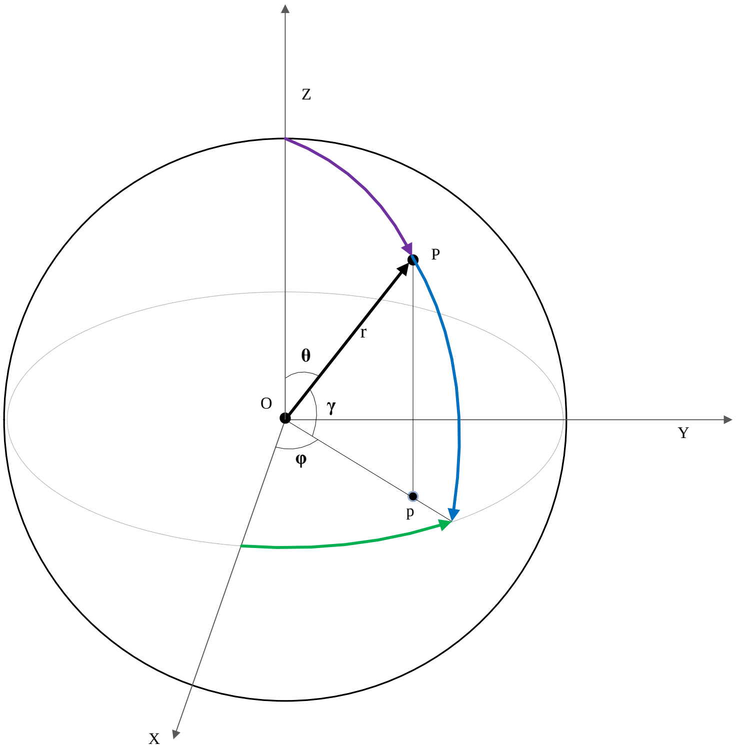


Fig. B.1 Sketch to illustrate Spherical Coordinates.

The problem is that there are many aliases of spherical coordinates, which would confuse the novitiates. θ could be addressed as polar angle, zenith angle, co-latitude. φ could be named as azimuth angle, longitude. The angle γ is also introduced in geography and used by other researchers, and it is called elevation or latitude. Some examples would be helpful to understand the nomenclatures. Take the north polar as an example, its colatitude/polar is 0° and elevation/latitude is 90°. The colatitude/polar of equator is 90°, and colatitude/polar of south pole is 180°. The longitude/azimuth of positive x axis , positive y axis, negative x axis and negative y axis is 0°, 90°, 180° and 270° respectively.

Storage order

There are two different storage orders for matrices when they are stored in memory: column-major order and row-major order. It is not a trivial when obtain the elements of matrix by pointers and pass on matrices between libraries which employ different storage orders. A matrix is stored in row-major order if it is stored row by row; in other words, consecutive elements of rows are contiguous in memory. Likewise, we say a matrix is stored in column-major order if consecutive elements of its columns are contiguous in memory. For a matrix A:

$$A = \begin{pmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \\ a31 & a32 & a33 \end{pmatrix}$$

If A is stored in row-major order, it is stored in memory as:
$a11, a12, a13, a21, a22, a23, a31, a32, a33$
While if A is stored in column-major order, it is stored in memory as:
$a11, a21, a31, a12, a22, a32, a13, a23, a33$

For example, C/C++, Mathematica, Python, FFTW employ the row-major order as their default storage order; while MATLAB, Eigen employ the column-major order. Please note that the default storage order of Eigen is column-major order, but it is possible to set by users.

APPENDIX C

Detailed results of FPFH-based and 3DSC-based feature matching techniques

The feature matching results of our SEI-based algorithm under different thresholds, applied in "**BoD1**" dataset [150] disturbed by different levels of Gaussian noises, are demonstrated detailedly in Table 4.8, Table 4.9 and 4.10. Meanwhile, the results of SHOT-based feature matching algorithm are also listed in Table 4.11, Table 4.12 and Table 4.13.

To keep the compactness, the details of FPFH-based and 3DSC-based feature matching algorithms are omitted in straight matter. So we present the detailed results of FPFH-based feature matching algorithm in Table C.1, Table C.2 and Table C.3. And Table C.4, Table C.5 and Table C.6 describe the detailed results of 3DSC-based feature matching algorithm.

Table C.1 Performances of FPFH-based feature matching algorithm applied to scenes with 0.1mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		1000.0		500.0		400.0		300.0		200.0		100.0	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	29	95	28	85	25	77	22	66	19	46	17	16	9
Dragon	100	59	95	58	83	54	80	53	67	50	52	42	34	31
Armadillo	100	44	99	44	85	44	73	41	59	38	44	35	34	32
Asian Dragon	100	37	94	37	84	37	78	36	67	35	52	31	31	25
Thai Statue	100	22	96	22	92	22	83	22	73	21	57	18	19	10
Bunny	100	52	90	50	73	47	66	47	58	42	48	34	30	23
SUM	600	243	569	239	502	229	457	221	390	205	299	177	164	130
Recall	**0.4050**		**0.3984**		**0.3817**		**0.3683**		**0.3417**		**0.2950**		**0.2167**	
Precision	**0.4050**		**0.4200**		**0.4562**		**0.4836**		**0.5256**		**0.5920**		**0.7927**	

Table C.2 Performances of FPFH-based feature matching algorithm applied to scenes with 0.3mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		1000.0		500.0		400.0		300.0		200.0		100.0	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	23	89	20	73	16	63	16	56	16	28	9	5	3
Dragon	100	38	93	37	70	32	56	30	38	24	20	15	4	3
Armadillo	100	37	95	37	71	34	65	32	49	27	32	21	10	7
Asian Dragon	100	34	93	34	78	32	67	30	57	28	46	26	23	20
Thai Statue	100	19	96	19	85	19	75	19	61	17	40	12	10	8
Bunny	100	39	83	37	55	30	44	27	34	20	19	13	5	4
SUM	600	190	549	184	432	163	370	154	295	132	185	96	57	45
Recall	**0.3167**		**0.3067**		**0.2717**		**0.2567**		**0.2200**		**0.1600**		**0.0750**	
Precision	**0.3167**		**0.3352**		**0.3773**		**0.4162**		**0.4475**		**0.5189**		**0.7895**	

Table C.3 Performances of FPFH-based feature matching algorithm applied to scenes with 0.5mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		1000.0		500.0		400.0		300.0		200.0		100.0	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	18	90	18	75	17	69	17	60	16	35	12	7	4
Dragon	100	19	87	18	43	12	31	12	22	10	11	5	1	1
Armadillo	100	25	98	25	68	22	50	19	33	13	16	7	2	1
Asian Dragon	100	25	94	25	81	25	69	25	60	23	41	18	11	5
Thai Statue	100	18	96	17	87	17	80	17	71	15	47	10	12	5
Bunny	100	21	81	20	51	14	37	10	19	7	5	4	0	0
SUM	600	126	546	123	405	107	336	100	265	84	155	56	33	16
Recall	**0.2100**		**0.2050**		**0.1783**		**0.1667**		**0.1400**		**0.0933**		**0.0267**	
Precision	**0.2100**		**0.2253**		**0.2642**		**0.2976**		**0.3170**		**0.3613**		**0.4848**	

Table C.4 Performances of 3DSC-based feature matching algorithm applied to scenes with 0.1mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		1.50e+08		1.40e+08		1.30e+08		1.20e+08		1.00e+08		0.50e+08	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	11	87	11	78	11	67	11	52	11	27	8	3	3
Dragon	100	11	23	8	16	7	13	7	11	7	5	5	4	4
Armadillo	100	20	85	20	76	20	65	20	58	20	22	14	8	8
Asian Dragon	100	19	86	19	76	19	63	19	48	18	33	17	10	9
Thai Statue	100	17	37	12	31	11	24	10	19	8	13	8	6	6
Bunny	100	27	99	27	99	27	98	27	97	27	92	27	34	20
SUM	600	105	417	97	376	95	330	94	285	91	192	79	65	50
Recall	**0.1750**		**0.1617**		**0.1583**		**0.1567**		**0.1517**		**0.1317**		**0.0833**	
Precision	**0.1750**		**0.2326**		**0.2527**		**0.2848**		**0.3193**		**0.4115**		**0.7692**	

Table C.5 Performances of 3DSC-based feature matching algorithm applied to scenes with 0.3mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		1.50e+08		1.40e+08		1.30e+08		1.20e+08		1.00e+08		0.50e+08	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	9	67	8	54	8	46	8	35	8	13	7	2	2
Dragon	100	14	10	5	7	5	4	3	3	3	3	3	0	0
Armadillo	100	19	50	18	42	18	31	16	20	12	12	9	4	4
Asian Dragon	100	19	51	16	42	15	38	14	32	14	21	13	7	6
Thai Statue	100	20	23	10	19	9	15	9	14	9	6	6	2	2
Bunny	100	29	98	29	98	29	97	29	96	29	85	28	31	19
SUM	600	110	299	86	262	84	231	79	200	75	140	66	46	33
Recall	**0.1833**		**0.1433**		**0.1400**		**0.1317**		**0.1250**		**0.1100**		**0.0550**	
Precision	**0.1833**		**0.2876**		**0.3206**		**0.3420**		**0.3750**		**0.4714**		**0.7174**	

Table C.6 Performances of 3DSC-based feature matching algorithm applied to scenes with 0.5mr Gaussian noise (**R**=number of Retrieved records and **G** = number of Relevant ∩ Retrieved records).

Threshold	+∞		1.50e+08		1.40e+08		1.30e+08		1.20e+08		1.00e+08		0.50e+08	
	R	G	R	G	R	G	R	G	R	G	R	G	R	G
Happy Buddha	100	10	42	9	34	8	28	8	21	8	6	5	1	1
Dragon	100	14	1	1	0	0	0	0	0	0	0	0	0	0
Armadillo	100	20	14	6	12	6	11	6	9	6	9	6	0	0
Asian Dragon	100	19	32	13	29	12	22	11	17	9	7	6	0	0
Thai Statue	100	19	11	8	8	7	7	7	5	5	4	4	1	1
Bunny	100	24	98	23	95	23	95	23	89	23	69	21	24	12
SUM	600	106	198	60	178	56	163	55	141	51	95	42	26	14
Recall	**0.1767**		**0.1000**		**0.0933**		**0.0917**		**0.0850**		**0.0700**		**0.0233**	
Precision	**0.1767**		**0.3030**		**0.3146**		**0.3374**		**0.3617**		**0.4421**		**0.5385**	

Eidesstattliche Versicherung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Dissertationsschrift selbst verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Hamburg, den Unterschrift